PowerDNS Recursor Documentation

PowerDNS.COM BV

Jul 25, 2024

CONTENTS

CHAPTER

ONE

INTRODUCTION

The PowerDNS Recursor is a high-performance DNS recursor with built-in scripting capabilities. It is known to power the resolving needs of over 150 million internet connections.

The documentation is only for the 4.1 and higher series, users of older versions are urged to read *End of life statements* and upgrade!

This documentation is also available as a PDF document.

1.1 Notable features

- Can handle tens of thousands of concurrent questions. A quad Xeon 3GHz has been measured functioning very well at 400000 real life replayed packets per second.
- Relies heavily on Standard C++ Library infrastructure.
- Powered by a highly modern DNS packet parser that should be resistant against many forms of buffer overflows.
- Best spoofing protection that we know about, involving both source port randomisation and spoofing detection.
- Uses 'connected' UDP sockets which allow the recursor to react quickly to unreachable hosts or hosts for which the server is running, but the nameserver is down. This makes the recursor faster to respond in case of misconfigured domains, which are sadly very frequent.
- Special support for *BSD, Linux and Solaris stateful multiplexing (kqueue, epoll, completion ports, /dev/poll).
- Very fast, and contains innovative query-throttling code to save time talking to obsolete or broken nameservers.
- Code is written linearly, sequentially, which means that there are no problems with 'query restart' or anything.
- Does DNSSEC validation
- Is highly scriptable in Lua

1.2 Getting support

PowerDNS is an open source program so you may get help from the PowerDNS users' community or from its authors. You may also help others (please do).

Public support is available via several different channels:

- This documentation
- The mailing list
- `#powerdns` on irc.oftc.net

The Open-Xchange/PowerDNS company can provide help or support you in private as well. Please contact Open-Xchange.

1.2.1 My information is confidential, must I send it to the mailing list, discuss it on IRC, or post it in a GitHub ticket?

Yes, we have a support policy called "Open Source Support: out in the open".

If you desire privacy, please consider entering a support relationship with us, in which case we invite you to contact Open-Xchange.

1.2.2 I have a question!

This happens, we're here to help! Read below on how you can get help

1.2.3 What details should I supply?

Start out with stating what you think should be happening. Quite often, wrong expectations are the actual problem. Furthermore, your operating system, which version of PowerDNS you use and where you got it from (RPM, .DEB, tar.bz2). If you *compiled* it yourself, what were the `./configure` parameters.

If possible, supply the actual name of your domain and the IP address of your server(s).

1.2.4 I found a bug!

As much as we'd like to think we are perfect, bugs happen. If you have found a bug, please file a bug report on GitHub bug report. Please fill in the template and we'll try our best to help you.

1.2.5 I found a security issue!

Please report this in private, see the *PowerDNS Security Policy*.

1.2.6 I have a good idea for a feature!

We like to work on new things! You can file a feature request on GitHub feature request.

1.3 Third party software

We use code from the project listed below.

1.3.1 Protozero

CHAPTER

TWO

GETTING STARTED

PowerDNS Recursor can be installed on any modern unix-like system and is available in the software repositories for all major Linux distributions and BSDs.

2.1 Installation

Recursor is available for many platforms, instructions are provided here for several platforms.

Note: As distribution provided package repositories are not always up-to-date, PowerDNS itself provides repositories for several **Recursor** versions for different operating systems. Checkout the repositories for more information.

2.1.1 Debian-based distributions

On Debian, Ubuntu, Linux Mint and related distributions, running `apt-get install pdns-recursor` as root will install **Recursor**.

2.1.2 Enterprise Linux

On Red Hat, CentOS and related distributions, ensure that EPEL is available. To install **Recursor**, run `yum install pdns-recursor` as root.

2.1.3 FreeBSD

On FreeBSD **Recursor** is available through the FreeBSD ports system. Run `pkg install powerdns-recursor` as root to install.

To compile yourself from ports, run `cd /usr/ports/dns/powerdns-recursor/ && make install clean`.

2.1.4 OpenBSD

On OpenBSD, **Recursor** is available through the OpenBSD ports system. Run `pkg_add powerdns-recursor` as root to install.

2.1.5 macOS

On macOS **Recursor** is available through brew. Run `brew install pdnsrec` to install.

2.1.6 Compiling From Source

See *Compiling PowerDNS Recursor* for instructions on how to build **Recursor** from source.

2.2 Configuring PowerDNS Recursor

The configuration file is called `recursor.conf` and is located in the `SYSCONFDIR` defined at compile-time. This is usually `/etc/powerdns`, `/etc/pdns`, `/etc/pdns-recursor`, `/usr/local/etc` or similar.

Run `pdns_recursor --config=default | grep config-dir` to find this location on your installation. Many packages provide a default configuration file that sets *include-dir*. Consider putting local `.conf` files into this directory, to make it clear which settings were locally modified.

Recursor listens on the local loopback interface by default, this can be changed with the *local-address* setting.

Now access will need to be granted to the **Recursor**. The *allow-from* setting lists the subnets that can communicate with **Recursor**.

An example configuration is shown below. Change this to match the local infrastructure.

```
local-address=192.0.2.25, 2001:DB8::1:25
allow-from=192.0.2.0/24, 2001:DB8::1:/64
```

After a restart of **Recursor**, it will answer queries on 192.0.2.25 and 2001:DB8::1:25, but only for queries with a source address in the 192.0.2.0/24 and 2001:DB8::1:/64 networks.

Recursor is now ready to be used. For more options that can be set in `recursor.conf` see the *PowerDNS Recursor Settings*. Guidance on interaction with **Recursor** is documented in *Operating PowerDNS Recursor*. If dynamic answer generation is needed or policies need to be applied to queries, the *Scripting PowerDNS Recursor* will come in handy.

2.3 Using Ansible

PowerDNS Recursor can also be installed and configured with Ansible. There is a role available from the PowerDNS authors.

CHAPTER

THREE

OPERATING POWERDNS RECURSOR

3.1 Logging

In a production environment, you will want to be able to monitor PowerDNS performance. Furthermore, PowerDNS can perform a configurable amount of operational logging.

On modern Linux distributions, the PowerDNS recursor logs to stderr, which is consumed by `systemd-journald`. This means that looking into the logs that are produced, journalctl can be used:

```
# journalctl -u pdns-recursor -n 100
```

Additionally, the Recursor *can* log to syslog on these systems. Logging to syslog is disabled in the unit file to prevent double logging. To enable this, create an drop in unit file at `/etc/systemd/systemd/pdns-recursor.service.d/use-syslog.conf`:

```
[Service]
ExecStart=
ExecStart=/usr/sbin/pdns_recursor --daemon=no --write-pid=no --enable-syslog
```

3.1.1 Logging to syslog

This chapter assumes familiarity with syslog, the unix logging device. PowerDNS logs messages with different levels. The more urgent the message, the lower the 'priority'.

By default, PowerDNS will only log messages with an urgency of 3 or lower, but this can be changed using the *loglevel* setting in the configuration file. Setting it to 0 will eliminate all logging, 9 will log everything.

By default, logging is performed under the 'DAEMON' facility which is shared with lots of other programs. If you regard nameserving as important, you may want to have it under a dedicated facility so PowerDNS can log to its own files, and not clutter generic files.

For this purpose, syslog knows about 'local' facilities, numbered from LOCAL0 to LOCAL7. To move PowerDNS logging to LOCAL0, add *logging-facility=0* to your configuration.

Furthermore, you may want to have separate files for the differing priorities - preventing lower priority messages from obscuring important ones. A sample `syslog.conf` might be:

```
local0.info                     -/var/log/pdns.info
local0.warn                     -/var/log/pdns.warn
local0.err                      /var/log/pdns.err
```

Where local0.err would store the really important messages. For performance and disk space reasons, it is advised to audit your `syslog.conf` for statements also logging PowerDNS activities. Many `syslog.confs` have a `*.*` statement to `/var/log/syslog`, which you may want to remove.

For performance reasons, be especially certain that no large amounts of synchronous logging take place. Under Linux, this is indicated by file names not starting with a - - indicating a synchronous log, which hurts performance.

Be aware that syslog by default logs messages at the configured priority and higher! To log only info messages, use `local0.=info`

3.2 Cache Management

Sometimes a domain fails to resolve due to an error on the domain owner's end, or records for your own domain have updated and you want your users to immediatly see them without waiting for the TTL to expire. The *rec_control* tool can be used to selectively wipe the cache.

To wipe all records for the exact name 'www.example.com':

```
rec_control wipe-cache www.example.com
```

Whole subtrees can we wiped as well, to wipe all cache entries for 'example.com' and everything below it, suffix the name with a '$':

```
rec_control wipe-cache example.com$
```

Note: When wiping cache entries, matching entries in *all* caches (packet cache, recursor cache, negative cache) are removed.

When debugging resolving issues, it can be advantageous to have a dump of all the cache entries. *rec_control* can write the caches of all threads to a file:

```
rec_control dump-cache /tmp/cache
```

3.3 Tracing Queries

To investigate failures with resolving certain domain names, the PowerDNS **Recursor** features a tracing infrastructure. This infrastructure will log every step the **Recursor** takes to resolve a name and will log all DNSSEC related information as well.

To enable tracing for all queries, enable the *trace* setting. Trace information will be written to the log.

Warning: Enabling tracing for all queries on a system with a high query rate can severely impact performance.

Tracing can also be enabled at runtime, without restarting the **Recursor**, for specific domains. These specific domains can be specified as a regular expression. This can be done using *rec_control trace-regex*:

```
rec_control trace-regex '.*\.example.com\.$'
```

Will enable tracing for any query *in* the example.com domain (but not example.com itself).

Since version 4.9.0 `trace_regex` takes an extra file argument. Trace information will be written to the file and not to the log. If the file argument is a hyphen (-), trace information will be written to the standard output stream. For example:

```
rec_control trace-regex 'example\.com\.$' - | grep asking
```

will show which authoritative servers were consulted.

Do not forget to disable tracing after diagnosis is done:

```
rec_control trace-regex
```

3.4 Logging details of queries and answers

In some cases a tracing provides too much information, and we want to follow what the recursor is doing on a higher level. By setting *logging.quiet* to `true` the recursor will produce a log line for each client query received and answered. Be aware that this causes overhead and should not be used in a high query-per-second production environment:

```
Jul 09 09:08:31 msg="Question" subsystem="syncres" level="0" prio="Info" tid="4"␣
↪ts="1720508911.919" ecs="" mtid="1" proto="udp" qname="www.example.com" qtype="A
↪" remote="127.0.0.1:54573"

Jul 09 09:08:32 msg="Answer" subsystem="syncres" level="0" prio="Info" tid="4" ts=
↪"1720508912.549" additional="1" answer-is-variable="0" answers="1" dotout="0"␣
↪ecs="" into-packetcache="1" maxdepth="3" mtid="1" netms="617.317000" outqueries=
↪"13" proto="udp" qname="www.example.com" qtype="A" rcode="0" rd="1" remote="127.
↪0.0.1:54573" tcpout="0" throttled="0" timeouts="0" totms="627.060000"␣
↪validationState="Secure"
```

When `quiet` is set to `false`, the following keys and values are logged for questions and answers not answered from the packet cache. Refer to *Structured Logging Dictionary* for more details on the common keys used for structured logging messages. Note that depending on record cache content a single client query can result into multiple queries to authoritative servers. If the exact answer is available from the record cache no outgoing queries are needed.

Keys common to Questions and Answers		
Key	**Description**	**Remarks**
`ecs`	Client ECS info	Filled in if enabled
`proto`	Protocol used by client	`udp` or `tcp`
`qname`	Query name	
`qtype`	Query type	
`remote`	Client address	IP:port combination
Keys specific to Answers		
`additional`	Number of additional records in answer	
`answer-is-variable`	Is answer marked variable by recursor?	e.g. ECS dependent answers
`answers`	Number of answer records in answer	
`dotout`	Number of outgoing DoT queries sent to authoritative servers to resolve answer	
`into-packetcache`	Is the answer being stored into the packetcache?	Variable answers (as determined by the recursor or marked as such by Lua code) will not be put into the packet cache
`maxdepth`	Depth of recursion needed to resolve answer	Some queries need resolving multiple targets, e.g. to find the right delegation or answers containing CNAMEs
`netms`	Time spent waiting for answers from authoritative servers	
`outqueries`	Total queries sent to authoritative servers	A single client query can cause multiple queries to authoritative servers, depending on record cache content and the query itself.
`rcode`	Result code	If no rcode is available (e.g. in the case of timeouts) this value can be negative
`rd`	Did the client set the Recursion Desired DNS Header flag?	
`tcpout`	Number of outgoing TCP queries sent to authoritative servers to resolve answer	
`throttled`	Number of potential outgoing queries **not** sent out because the target was marked as unreliable by previous interactions	If a target is throttled, the recursor will try another suitable authoritative server (if available)
`timeouts`	Number of outgoing queries that timed out	
`totms`	Total time spent resolving	
`validation-state`	The DNSSEC status of the answer	

CHAPTER

FOUR

DNSSEC IN THE POWERDNS RECURSOR

As of 4.0.0, the PowerDNS Recursor has support for DNSSEC processing and experimental support for DNSSEC validation.

4.1 DNSSEC settings

The PowerDNS Recursor has 5 different levels of DNSSEC processing, which can be set with the *dnssec* setting in the `recursor.conf`. In order from least to most processing, these are:

4.1.1 `off`

In this mode, **no** DNSSEC processing takes place. The PowerDNS Recursor will not set the DNSSEC OK (DO) bit in the outgoing queries and will ignore the DO and AD bits in queries.

4.1.2 `process-no-validate`

The default mode until PowerDNS Recursor 4.5.0.

In this mode the Recursor acts as a "security aware, non-validating" nameserver, meaning it will set the DO-bit on outgoing queries and will provide DNSSEC related RRsets (NSEC, RRSIG) to clients that ask for them (by means of a DO-bit in the query), except for zones provided through the `auth-zones` setting. It will not do any validation in this mode, not even when requested by the client.

4.1.3 `process`

The default mode since PowerDNS Recursor 4.5.0.

When *dnssec* is set to `process` the behaviour is similar to *process-no-validate*. However, the recursor will try to validate the data if at least one of the DO or AD bits is set in the query; in that case, it will set the AD-bit in the response when the data is validated successfully, or send SERVFAIL when the validation comes up bogus.

4.1.4 `log-fail`

In this mode, the recursor will attempt to validate all data it retrieves from authoritative servers, regardless of the client's DNSSEC desires, and will log the validation result. This mode can be used to determine the extra load and amount of possibly bogus answers before turning on full-blown validation. Responses to client queries are the same as with *process*.

4.1.5 validate

The highest mode of DNSSEC processing. In this mode, all responses will be be validated and queries will be answered with a SERVFAIL in case of bogus data, even if the client did not request validation by setting the AD or DO bit.

Note: the CD-bit is honored for `process`, `log-fail` and `validate`. This mean that even if validation fails, results are returned if the CD-bit is set by the client. For `log-fail`, failures will be logged too.

4.1.6 What, when?

The descriptions above are a bit terse, here's a table describing different scenarios with regards to the `dnssec` mode.

	`off`	`process-no-validate`	`process`	`log-fail`	`validate`
Perform validation	No	No	Only on +AD or +DO from client	Always (logs result)	Always
SERVFAIL on bogus	No	No	Only on +AD or +DO and -CD from client	Only on +AD or +DO and -CD from client	If -CD from client
AD in response on authenticated data	Never	Never	Only on +AD or +DO from client	Only on +AD or +DO from client	Only on +AD or +DO from client
RRSIGs/NSECs in answer on +DO from client	No	Yes	Yes	Yes	Yes

Note: the `dig` tool sets the AD-bit in the query. This might lead to unexpected query results when testing. Set `+noad` on the `dig` commandline when this is the case.

4.2 Trust Anchor Management

In the PowerDNS Recursor, both positive and negative trust anchors can be configured during startup (from a persistent configuration file) and at runtime (which is volatile). However, all trust anchors are configurable.

Current trust anchors can be queried from the recursor by sending a query for "trustanchor.server CH TXT". This query will (if *allow-trust-anchor-query* is enabled) return a TXT record per trust-anchor in the format `"DOMAIN KEYTAG [KEYTAG]..."`.

4.2.1 Trust Anchors

The PowerDNS Recursor ships with the DNSSEC Root key built-in.

Note: it has no support for **RFC 5011** key rollover and does not persist a changed root trust anchor to disk.

Configuring DNSSEC key material must be done in the *lua-config-file*, using `addTA()`. This function takes 2 arguments: the node in the DNS-tree and the data of the corresponding DS record.

To e.g. add a trust anchor for the root and example.com, use the following config in the Lua file:

```
addTA('.', "63149 13 1 a59da3f5c1b97fcd5fa2b3b2b0ac91d38a60d33a") -- This is not␣
↪an ICANN root
addTA('example.com', "44030 8 2␣
↪D4C3D5552B8679FAEEBC317E5F048B614B2E5F607DC57F1553182D49 AB2179F7")
```

For PowerDNS Recursor 4.1.x and below, use the `addDS()` function instead.

Now (re)start the recursor to load these trust anchors.

Reading trust anchors from files

New in version 4.2.0.

It is also possible to read the Trust Anchors from a BIND-style zonefile using the `readTrustAnchorsFromFile()` in the *lua-config-file*. Only the DS and DNSKEY records from this file are read. This file is (by default) re-read every 24 hours for updates. Debian and its derivatives ship the `dns-root-data` package that contains the DNSSEC root trust anchors in `/usr/share/dns/root.key`.

To only use the distribution-provided Trust Anchors, add the following to the *lua-config-file*:

```
clearTA() -- Remove built-in trust-anchors
readTrustAnchorsFromFile("/usr/share/dns/root.key") -- Use these keys
```

Note: When using `readTrustAnchorsFromFile()`, any runtime changes to Trust Anchors (see below) will be overwritten when the file is refreshed. To prevent this, set the `interval` parameter to `0`. This will **disable** automatic reloading of the file.

Runtime Configuration of Trust Anchors

To change or add trust anchors at runtime, use the *rec_control* tool. These runtime settings are not saved to disk. To make them permanent, they should be added to the *lua-config-file* as described above.

Adding a trust anchor is done with the `add-ta` command:

```
$ rec_control add-ta domain.example 63149 13 1␣
↪a59da3f5c1b97fcd5fa2b3b2b0ac91d38a60d33a
Added Trust Anchor for domain.example. with data 63149 13 1␣
↪a59da3f5c1b97fcd5fa2b3b2b0ac91d38a60d33a
```

To view the currently configured trust anchors, run `get-tas`:

```
$ rec_control get-tas
Configured Trust Anchors:
.       63149 13 1 a59da3f5c1b97fcd5fa2b3b2b0ac91d38a60d33a
net.    2574 13 1 a5c5acb889a7ba9b5aa5bef2b0ac9fe1565ddaab
```

To remove a trust anchor, run `clear-ta`:

```
$ rec_control clear-ta domain.example
Removed Trust Anchor for subdomain.example
```

Note: The root trust anchor cannot be removed in this manner.

4.2.2 Negative Trust Anchors

Negative trust anchors (defined in **RFC 7646**) can be used to temporarily disable DNSSEC validation for a part of the DNS-tree. This can be done when e.g. a TLD or high-traffic zone goes bogus. Note that it is good practice to verify that this is indeed the case and not because of malicious actions.

Current negative trust anchors can be queried from the recursor by sending a query for "negativetrustanchor.server CH TXT". This query will (if *allow-trust-anchor-query* is enabled) return a TXT record per negative trust-anchor in the format `"DOMAIN [REASON]"`.

To configure a negative trust anchor, use the `addNTA()` function in the *lua-config-file* and restart the recursor. This function requires the name of the zone and an optional reason:

```
addNTA('example.', "Someone messed up the delegation")
addNTA('powerdns.com') -- No reason given
```

Runtime Configuration of Negative Trust Anchors

The *rec_control* command can be used to manage the negative trust anchors of a running instance. These runtime settings are lost when restarting the recursor, more permanent NTAs should be added to the *lua-config-file* with `addNTA()`.

Adding a negative trust anchor is done with the `add-nta` command (that optionally accepts a reason):

```
$ rec_control add-nta domain.example botched keyroll
Added Negative Trust Anchor for domain.example. with reason 'botched keyroll'
```

To view the currently configured negative trust anchors, run `get-ntas`:

```
$ rec_control get-ntas
Configured Negative Trust Anchors:
subdomain.example.      Operator failed key-roll
otherdomain.example.    DS in parent, no DNSKEY in zone
```

To remove negative trust anchor(s), run `clear-nta`:

```
$ rec_control clear-nta subdomain.example
Removed Negative Trust Anchors for subdomain.example
```

`clear-nta` accepts multiple domain-names and accepts '*' (beware the shell quoting) to remove all negative trust anchors.

CHAPTER

FIVE

POWERDNS RECURSOR SETTINGS

Each setting can appear on the command line, prefixed by `--`, or in the configuration file. The command line overrides the configuration file.

Note: Starting with version 5.0.0, **`Recursor`** supports a new YAML syntax for configuration files. A configuration using the old style syntax can be converted to a YAML configuration using the instructions in *Conversion of old-style settings to YAML format*. In a future release support for the "old-style" settings decribed here will be dropped. See *PowerDNS Recursor New Style (YAML) Settings* for details.

Note: Settings marked as `Boolean` can either be set to an empty value, which means **on**, or to `no` or `off` which means **off**. Anything else means **on**.

For example:

- `serve-rfc1918` on its own means: do serve those zones.
- `serve-rfc1918 = off` or `serve-rfc1918 = no` means: do not serve those zones.
- Anything else means: do serve those zones.

You can use `+=` syntax to set some variables incrementally, but this requires you to have at least one non-incremental setting for the variable to act as base setting. This is mostly useful for *include-dir* directive. An example:

```
forward-zones = foo.example.com=192.168.100.1;
forward-zones += bar.example.com=[1234::abcde]:5353;
```

When a list of **Netmasks** is mentioned, a list of subnets can be specified. A subnet that is not followed by `/` will be interpreted as a `/32` or `/128` subnet (a single address), depending on address family. For most settings, it is possible to exclude ranges by prefixing an item with the negation character `!`. For example:

```
allow-from = 2001:DB8::/32, 128.66.0.0/16, !128.66.1.2
```

In this case the address `128.66.1.2` is excluded from the addresses allowed access.

5.1 The Settings

5.1.1 `aggressive-nsec-cache-size`

New in version 4.5.0.

- Integer
- Default: 100000
- YAML setting: *dnssec.aggressive_nsec_cache_size*

The number of records to cache in the aggressive cache. If set to a value greater than 0, the recursor will cache NSEC and NSEC3 records to generate negative answers, as defined in **RFC 8198**. To use this, DNSSEC processing or validation must be enabled by setting *dnssec* to `process`, `log-fail` or `validate`.

5.1.2 aggressive-cache-min-nsec3-hit-ratio

New in version 4.9.0.

- Integer
- Default: 2000
- YAML setting: *dnssec.aggressive_cache_min_nsec3_hit_ratio*

The limit for which to put NSEC3 records into the aggressive cache. A value of `n` means that an NSEC3 record is only put into the aggressive cache if the estimated probability of a random name hitting the NSEC3 record is higher than `1/n`. A higher `n` will cause more records to be put into the aggressive cache, e.g. a value of 4000 will cause records to be put in the aggressive cache even if the estimated probability of hitting them is twice as low as would be the case for `n=2000`. A value of 0 means no NSEC3 records will be put into the aggressive cache.

For large zones the effectiveness of the NSEC3 cache is reduced since each NSEC3 record only covers a randomly distributed subset of all possible names. This setting avoids doing unnecessary work for such large zones.

5.1.3 allow-from

- Comma separated list of IP addresses or subnets, negation supported
- Default: 127.0.0.0/8, 10.0.0.0/8, 100.64.0.0/10, 169.254.0.0/16, 192.168.0.0/16, 172.16.0.0/12, ::1/128, fc00::/7, fe80::/10
- YAML setting: *incoming.allow_from*

Netmasks (both IPv4 and IPv6) that are allowed to use the server. The default allows access only from **RFC 1918** private IP addresses. An empty value means no checking is done, all clients are allowed. Due to the aggressive nature of the internet these days, it is highly recommended to not open up the recursor for the entire internet. Questions from IP addresses not listed here are ignored and do not get an answer.

When the Proxy Protocol is enabled (see *proxy-protocol-from*), the recursor will check the address of the client IP advertised in the Proxy Protocol header instead of the one of the proxy.

Note that specifying an IP address without a netmask uses an implicit netmask of /32 or /128.

5.1.4 allow-from-file

- String
- Default: (empty)
- YAML setting: *incoming.allow_from_file*

Like *allow-from*, except reading from file. Overrides the *allow-from* setting. To use this feature, supply one netmask per line, with optional comments preceded by a '#'.

5.1.5 allow-notify-for

New in version 4.6.0.

- Comma separated list of strings
- Default: (empty)
- YAML setting: *incoming.allow_notify_for*

Domain names specified in this list are used to permit incoming NOTIFY operations to wipe any cache entries that match the domain name. If this list is empty, all NOTIFY operations will be ignored.

5.1.6 allow-notify-for-file

New in version 4.6.0.

- String
- Default: (empty)
- YAML setting: *incoming.allow_notify_for_file*

Like *allow-notify-for*, except reading from file. To use this feature, supply one domain name per line, with optional comments preceded by a '#'.

NOTIFY-allowed zones can also be specified using *forward-zones-file*.

5.1.7 allow-notify-from

New in version 4.6.0.

- Comma separated list of IP addresses or subnets, negation supported
- Default: (empty)
- YAML setting: *incoming.allow_notify_from*

Netmasks (both IPv4 and IPv6) that are allowed to issue NOTIFY operations to the server. NOTIFY operations from IP addresses not listed here are ignored and do not get an answer.

When the Proxy Protocol is enabled (see *proxy-protocol-from*), the recursor will check the address of the client IP advertised in the Proxy Protocol header instead of the one of the proxy.

Note that specifying an IP address without a netmask uses an implicit netmask of /32 or /128.

NOTIFY operations received from a client listed in one of these netmasks will be accepted and used to wipe any cache entries whose zones match the zone specified in the NOTIFY operation, but only if that zone (or one of its parents) is included in *allow-notify-for*, *allow-notify-for-file*, or *forward-zones-file* with a '^' prefix.

5.1.8 allow-notify-from-file

New in version 4.6.0.

- String
- Default: (empty)
- YAML setting: *incoming.allow_notify_from_file*

Like *allow-notify-from*, except reading from file. To use this feature, supply one netmask per line, with optional comments preceded by a '#'.

5.1.9 allow-no-rd

New in version 5.0.0.

- Boolean
- Default: no
- YAML setting: *incoming.allow_no_rd*

Allow `no recursion desired (RD=0) queries` to query cache contents. If not set (the default), these queries are answered with rcode `Refused`.

5.1.10 any-to-tcp

- Boolean
- Default: no
- YAML setting: *recursor.any_to_tcp*

Answer questions for the ANY type on UDP with a truncated packet that refers the remote server to TCP. Useful for mitigating ANY reflection attacks.

5.1.11 allow-trust-anchor-query

New in version 4.3.0.

- Boolean
- Default: no
- YAML setting: *recursor.allow_trust_anchor_query*

Allow `trustanchor.server CH TXT` and `negativetrustanchor.server CH TXT` queries to view the configured *DNSSEC* (negative) trust anchors.

5.1.12 api-config-dir

New in version 4.0.0.

- String
- Default: (empty)
- YAML setting: *webservice.api_dir*

Directory where the REST API stores its configuration and zones. For configuration updates to work, *include-dir* should have the same value when using old-style settings. When using YAML settings *recursor.include_dir* and *webservice.api_dir* must have a different value.

5.1.13 api-key

New in version 4.0.0.

Changed in version 4.6.0: This setting now accepts a hashed and salted version.

- String
- Default: (empty)
- YAML setting: *webservice.api_key*

Static pre-shared authentication key for access to the REST API. Since 4.6.0 the key can be hashed and salted using `rec_control hash-password` instead of being stored in the configuration in plaintext, but the plaintext version is still supported.

5.1.14 auth-zones

- Comma separated list of 'zonename=filename' pairs
- Default: (empty)
- YAML setting: *recursor.auth_zones*

Zones read from these files (in BIND format) are served authoritatively (but without the AA bit set in responses). DNSSEC is not supported. Example:

```
auth-zones=example.org=/var/zones/example.org, powerdns.com=/var/zones/powerdns.com
```

5.1.15 carbon-interval

- Integer
- Default: 30
- YAML setting: *carbon.interval*

If sending carbon updates, this is the interval between them in seconds. See *Metrics and Statistics*.

5.1.16 carbon-namespace

New in version 4.2.0.

- String
- Default: pdns
- YAML setting: *carbon.ns*

Change the namespace or first string of the metric key. The default is pdns.

5.1.17 carbon-ourname

- String
- Default: (empty)
- YAML setting: *carbon.ourname*

If sending carbon updates, if set, this will override our hostname. Be careful not to include any dots in this setting, unless you know what you are doing. See *Sending metrics to Graphite/Metronome over Carbon*.

5.1.18 carbon-instance

New in version 4.2.0.

- String
- Default: recursor
- YAML setting: *carbon.instance*

Change the instance or third string of the metric key. The default is recursor.

5.1.19 carbon-server

- Comma separated list or IPs of IP:port combinations
- Default: (empty)
- YAML setting: *carbon.server*

If set to an IP or IPv6 address, will send all available metrics to this server via the carbon protocol, which is used by graphite and metronome. Moreover you can specify more than one server using a comma delimited list, ex: carbon-server=10.10.10.10,10.10.10.20. You may specify an alternate port by appending :port, for example: `127.0.0.1:2004`. See *Metrics and Statistics*.

5.1.20 chroot

- String
- Default: (empty)
- YAML setting: *recursor.chroot*

If set, chroot to this directory for more security. This is not recommended; instead, we recommend containing PowerDNS using operating system features. We ship systemd unit files with our packages to make this easy.

Make sure that `/dev/log` is available from within the chroot. Logging will silently fail over time otherwise (on logrotate).

When using `chroot`, all other paths (except for *config-dir*) set in the configuration are relative to the new root.

When running on a system where systemd manages services, `chroot` does not work out of the box, as PowerDNS cannot use the `NOTIFY_SOCKET`. Either do not `chroot` on these systems or set the 'Type' of this service to 'simple' instead of 'notify' (refer to the systemd documentation on how to modify unit-files).

5.1.21 client-tcp-timeout

- Integer
- Default: 2
- YAML setting: *incoming.tcp_timeout*

Time to wait for data from TCP clients.

5.1.22 config-dir

- String
- Default: Determined by distribution
- YAML setting: *recursor.config_dir*

Location of configuration directory (where `recursor.conf` or `recursor.yml` is stored). Usually `/etc/powerdns`, but this depends on `SYSCONFDIR` during compile-time. Use default or set on command line.

5.1.23 config-name

- String
- Default: (empty)
- YAML setting: *recursor.config_name*

When running multiple recursors on the same server, read settings from `recursor-name.conf`, this will also rename the binary image.

5.1.24 cpu-map

- String
- Default: (empty)
- YAML setting: *recursor.cpu_map*

Set CPU affinity for threads, asking the scheduler to run those threads on a single CPU, or a set of CPUs. This parameter accepts a space separated list of thread-id=cpu-id, or thread-id=cpu-id-1,cpu-id-2,...,cpu-id-N. For example, to make the worker thread 0 run on CPU id 0 and the worker thread 1 on CPUs 1 and 2:

```
recursor:
  cpu_map: 0=0 1=1,2
```

The thread handling the control channel, the webserver and other internal stuff has been assigned id 0, the distributor threads if any are assigned id 1 and counting, and the worker threads follow behind. The number of distributor threads is determined by *distributor-threads*, the number of worker threads is determined by the *threads* setting.

This parameter is only available if the OS provides the `pthread_setaffinity_np()` function.

Note that depending on the configuration the Recursor can start more threads. Typically these threads will sleep most of the time. These threads cannot be specified in this setting as their thread-ids are left unspecified.

5.1.25 daemon

Changed in version 4.0.0: Default is now `no`, was `yes` before.

- Boolean
- Default: no
- YAML setting: *recursor.daemon*

Operate in the background.

5.1.26 dont-throttle-names

New in version 4.2.0.

- Comma separated list of strings
- Default: (empty)
- YAML setting: *outgoing.dont_throttle_names*

When an authoritative server does not answer a query or sends a reply the recursor does not like, it is throttled. Any servers' name suffix-matching the supplied names will never be throttled.

> **Warning:** Most servers on the internet do not respond for a good reason (overloaded or unreachable), `dont-throttle-names` could make this load on the upstream server even higher, resulting in further service degradation.

5.1.27 dont-throttle-netmasks

New in version 4.2.0.

- Comma separated list of IP addresses or subnets, negation supported
- Default: (empty)
- YAML setting: *outgoing.dont_throttle_netmasks*

When an authoritative server does not answer a query or sends a reply the recursor does not like, it is throttled. Any servers matching the supplied netmasks will never be throttled.

This can come in handy on lossy networks when forwarding, where the same server is configured multiple times (e.g. with `forward-zones-recurse=example.com=192.0.2.1;192.0.2.1`). By default, the PowerDNS Recursor would throttle the 'first' server on a timeout and hence not retry the 'second' one. In this case, `dont-throttle-netmasks` could be set to `192.0.2.1`.

> **Warning:** Most servers on the internet do not respond for a good reason (overloaded or unreachable), `dont-throttle-netmasks` could make this load on the upstream server even higher, resulting in further service degradation.

5.1.28 `disable-packetcache`

- Boolean
- Default: no
- YAML setting: *packetcache.disable*

Turn off the packet cache. Useful when running with Lua scripts that modify answers in such a way they cannot be cached, though individual answer caching can be controlled from Lua as well.

5.1.29 `disable-syslog`

- Boolean
- Default: no
- YAML setting: *logging.disable_syslog*

Do not log to syslog, only to stderr. Use this setting when running inside a supervisor that handles logging (like systemd). **Note**: do not use this setting in combination with *daemon* as all logging will disappear.

5.1.30 `distribution-load-factor`

New in version 4.1.12.

- Double
- Default: 0.0
- YAML setting: *incoming.distribution_load_factor*

If *pdns-distributes-queries* is set and this setting is set to another value than 0, the distributor thread will use a bounded load-balancing algorithm while distributing queries to worker threads, making sure that no thread is assigned more queries than distribution-load-factor times the average number of queries currently processed by all the workers. For example, with a value of 1.25, no server should get more than 125 % of the average load. This helps making sure that all the workers have roughly the same share of queries, even if the incoming traffic is very skewed, with a larger number of requests asking for the same qname.

5.1.31 `distribution-pipe-buffer-size`

New in version 4.2.0.

- Integer
- Default: 0
- YAML setting: *incoming.distribution_pipe_buffer_size*

Size in bytes of the internal buffer of the pipe used by the distributor to pass incoming queries to a worker thread. Requires support for *F_SETPIPE_SZ* which is present in Linux since 2.6.35. The actual size might be rounded up to a multiple of a page size. 0 means that the OS default size is used. A large buffer might allow the recursor to deal with very short-lived load spikes during which a worker thread gets overloaded, but it will be at the cost of an increased latency.

5.1.32 distributor-threads

New in version 4.2.0.

- Integer
- Default: 1 if *pdns-distributes-queries* is set, 0 otherwise
- YAML setting: *incoming.distributor_threads*

If *pdns-distributes-queries* is set, spawn this number of distributor threads on startup. Distributor threads handle incoming queries and distribute them to other threads based on a hash of the query.

5.1.33 dot-to-auth-names

New in version 4.6.0.

- Comma separated list of strings
- Default: (empty)
- YAML setting: *outgoing.dot_to_auth_names*

Force DoT to the listed authoritative nameservers. For this to work, DoT support has to be compiled in. Currently, the certificate is not checked for validity in any way.

5.1.34 dot-to-port-853

New in version 4.6.0.

- Boolean
- Default: yes
- YAML setting: *outgoing.dot_to_port_853*

Enable DoT to forwarders that specify port 853.

5.1.35 dns64-prefix

New in version 4.4.0.

- String
- Default: (empty)
- YAML setting: *recursor.dns64_prefix*

Enable DNS64 (**RFC 6147**) support using the supplied /96 IPv6 prefix. This will generate 'fake' `AAAA` records for names with only `A` records, as well as 'fake' `PTR` records to make sure that reverse lookup of DNS64-generated IPv6 addresses generate the right name. See *DNS64 support* for more flexible but slower alternatives using Lua.

5.1.36 dnssec

New in version 4.0.0.

Changed in version 4.5.0: The default changed from `process-no-validate` to `process`

- String
- Default: process
- YAML setting: *dnssec.validation*

One of `off`, `process-no-validate`, `process`, `log-fail`, `validate`

Set the mode for DNSSEC processing, as detailed in *DNSSEC in the PowerDNS Recursor*.

`off` No DNSSEC processing whatsoever. Ignore DO-bits in queries, don't request any DNSSEC information from authoritative servers. This behaviour is similar to PowerDNS Recursor pre-4.0.

`process-no-validate` Respond with DNSSEC records to clients that ask for it, set the DO bit on all outgoing queries. Don't do any validation.

`process` Respond with DNSSEC records to clients that ask for it, set the DO bit on all outgoing queries. Do validation for clients that request it (by means of the AD- bit or DO-bit in the query).

`log-fail` Similar behaviour to `process`, but validate RRSIGs on responses and log bogus responses.

`validate` Full blown DNSSEC validation. Send SERVFAIL to clients on bogus responses.

5.1.37 `dnssec-disabled-algorithms`

New in version 4.9.0.

- Comma separated list of strings
- Default: (empty)
- YAML setting: *dnssec.disabled_algorithms*

A list of DNSSEC algorithm numbers that should be considered disabled. These algorithms will not be used to validate DNSSEC signatures. Zones (only) signed with these algorithms will be considered `Insecure`.

If this setting is empty (the default), **`Recursor`** will determine which algorithms to disable automatically. This is done for specific algorithms only, currently algorithms 5 (`RSASHA1`) and 7 (`RSASHA1NSEC3SHA1`).

This is important on systems that have a default strict crypto policy, like RHEL9 derived systems. On such systems not disabling some algorithms (or changing the security policy) will make affected zones to be considered `Bogus` as using these algorithms fails.

5.1.38 `dnssec-log-bogus`

- Boolean
- Default: no
- YAML setting: *dnssec.log_bogus*

Log every DNSSEC validation failure. **Note**: This is not logged per-query but every time records are validated as Bogus.

5.1.39 `dont-query`

- Comma separated list of IP addresses or subnets, negation supported
- Default: 127.0.0.0/8, 10.0.0.0/8, 100.64.0.0/10, 169.254.0.0/16, 192.168.0.0/16, 172.16.0.0/12, ::1/128, fc00::/7, fe80::/10, 0.0.0.0/8, 192.0.0.0/24, 192.0.2.0/24, 198.51.100.0/24, 203.0.113.0/24, 240.0.0.0/4, ::/96, ::ffff:0:0/96, 100::/64, 2001:db8::/32
- YAML setting: *outgoing.dont_query*

The DNS is a public database, but sometimes contains delegations to private IP addresses, like for example 127.0.0.1. This can have odd effects, depending on your network, and may even be a security risk. Therefore, the PowerDNS Recursor by default does not query private space IP addresses. This setting can be used to expand or reduce the limitations.

Queries for names in forward zones and to addresses as configured in any of the settings *forward-zones*, *forward-zones-file* or *forward-zones-recurse* are performed regardless of these limitations.

5.1.40 ecs-add-for

New in version 4.2.0.

- Comma separated list of IP addresses or subnets, negation supported
- Default: 0.0.0.0/0, ::/0, !127.0.0.0/8, !10.0.0.0/8, !100.64.0.0/10, !169.254.0.0/16, !192.168.0.0/16, !172.16.0.0/12, !::1/128, !fc00::/7, !fe80::/10
- YAML setting: *ecs.add_for*

List of requestor netmasks for which the requestor IP Address should be used as the **EDNS Client Subnet** for outgoing queries. Outgoing queries for requestors that do not match this list will use the *ecs-scope-zero-address* instead. Valid incoming ECS values from *use-incoming-edns-subnet* are not replaced.

Regardless of the value of this setting, ECS values are only sent for outgoing queries matching the conditions in the *edns-subnet-allow-list* setting. This setting only controls the actual value being sent.

This defaults to not using the requestor address inside RFC1918 and similar 'private' IP address spaces.

5.1.41 ecs-ipv4-bits

New in version 4.1.0.

- Integer
- Default: 24
- YAML setting: *ecs.ipv4_bits*

Number of bits of client IPv4 address to pass when sending EDNS Client Subnet address information.

5.1.42 ecs-ipv4-cache-bits

New in version 4.1.12.

- Integer
- Default: 24
- YAML setting: *ecs.ipv4_cache_bits*

Maximum number of bits of client IPv4 address used by the authoritative server (as indicated by the EDNS Client Subnet scope in the answer) for an answer to be inserted into the record cache. This condition applies in conjunction with `ecs-cache-limit-ttl`. That is, only if both the limits apply, the record will not be cached. This decision can be overridden by `ecs-ipv4-never-cache` and `ecs-ipv6-never-cache`.

5.1.43 ecs-ipv6-bits

New in version 4.1.0.

- Integer
- Default: 56
- YAML setting: *ecs.ipv6_bits*

Number of bits of client IPv6 address to pass when sending EDNS Client Subnet address information.

5.1.44 ecs-ipv6-cache-bits

New in version 4.1.12.

- Integer
- Default: 56
- YAML setting: *ecs.ipv6_cache_bits*

Maximum number of bits of client IPv6 address used by the authoritative server (as indicated by the EDNS Client Subnet scope in the answer) for an answer to be inserted into the record cache. This condition applies in conjunction with `ecs-cache-limit-ttl`. That is, only if both the limits apply, the record will not be cached. This decision can be overridden by `ecs-ipv4-never-cache` and `ecs-ipv6-never-cache`.

5.1.45 ecs-ipv4-never-cache

New in version 4.5.0.

- Boolean
- Default: no
- YAML setting: *ecs.ipv4_never_cache*

When set, never cache replies carrying EDNS IPv4 Client Subnet scope in the record cache. In this case the decision made by `` `ecs-ipv4-cache-bits `` and `ecs-cache-limit-ttl` is no longer relevant.

5.1.46 ecs-ipv6-never-cache

New in version 4.5.0.

- Boolean
- Default: no
- YAML setting: *ecs.ipv6_never_cache*

When set, never cache replies carrying EDNS IPv6 Client Subnet scope in the record cache. In this case the decision made by `` `ecs-ipv6-cache-bits `` and `ecs-cache-limit-ttl` is no longer relevant.

5.1.47 ecs-minimum-ttl-override

Changed in version 4.5.0: Old versions used default 0.

- Integer
- Default: 1
- YAML setting: *ecs.minimum_ttl_override*

This setting artificially raises the TTLs of records in the ANSWER section of ECS-specific answers to be at least this long. Setting this to a value greater than 1 technically is an RFC violation, but might improve performance a lot. Using a value of 0 impacts performance of TTL 0 records greatly, since it forces the recursor to contact authoritative servers every time a client requests them. Can be set at runtime using `rec_control set-ecs-minimum-ttl 3600`.

5.1.48 ecs-cache-limit-ttl

New in version 4.1.12.

- Integer
- Default: 0

- YAML setting: *ecs.cache_limit_ttl*

The minimum TTL for an ECS-specific answer to be inserted into the record cache. This condition applies in conjunction with `ecs-ipv4-cache-bits` or `ecs-ipv6-cache-bits`. That is, only if both the limits apply, the record will not be cached. This decision can be overridden by `ecs-ipv4-never-cache` and `ecs-ipv6-never-cache`.

5.1.49 ecs-scope-zero-address

New in version 4.1.0.

- String
- Default: (empty)
- YAML setting: *ecs.scope_zero_address*

The IP address sent via EDNS Client Subnet to authoritative servers listed in *edns-subnet-allow-list* when *use-incoming-edns-subnet* is set and the query has an ECS source prefix-length set to 0. The default is to look for the first usable (not an `any` one) address in *query-local-address* (starting with IPv4). If no suitable address is found, the recursor fallbacks to sending 127.0.0.1.

5.1.50 edns-outgoing-bufsize

Changed in version 4.2.0: Before 4.2.0, the default was 1680

- Integer
- Default: 1232
- YAML setting: *outgoing.edns_bufsize*

Note: Why 1232?

1232 is the largest number of payload bytes that can fit in the smallest IPv6 packet. IPv6 has a minimum MTU of 1280 bytes (**RFC 8200, section 5**), minus 40 bytes for the IPv6 header, minus 8 bytes for the UDP header gives 1232, the maximum payload size for the DNS response.

This is the value set for the EDNS0 buffer size in outgoing packets. Lower this if you experience timeouts.

5.1.51 edns-padding-from

New in version 4.5.0.

Changed in version 5.0.5: YAML settings only: previously this was defined as a string instead of a sequence

- Comma separated list of IP addresses or subnets, negation supported
- Default: (empty)
- YAML setting: *incoming.edns_padding_from*

List of netmasks (proxy IP in case of proxy-protocol presence, client IP otherwise) for which EDNS padding will be enabled in responses, provided that *edns-padding-mode* applies.

5.1.52 edns-padding-mode

New in version 4.5.0.

- String
- Default: padded-queries-only

- YAML setting: *incoming.edns_padding_mode*

One of `always`, `padded-queries-only`. Whether to add EDNS padding to all responses (`always`) or only to responses for queries containing the EDNS padding option (`padded-queries-only`, the default). In both modes, padding will only be added to responses for queries coming from *edns-padding-from* sources.

5.1.53 edns-padding-out

New in version 4.8.0.

- Boolean
- Default: yes
- YAML setting: *outgoing.edns_padding*

Whether to add EDNS padding to outgoing DoT queries.

5.1.54 edns-padding-tag

New in version 4.5.0.

- Integer
- Default: 7830
- YAML setting: *incoming.edns_padding_tag*

The packetcache tag to use for padded responses, to prevent a client not allowed by the :ref::*setting-edns-padding-from* list to be served a cached answer generated for an allowed one. This effectively divides the packet cache in two when *edns-padding-from* is used. Note that this will not override a tag set from one of the `Lua` hooks.

5.1.55 edns-subnet-whitelist

Deprecated since version 4.5.0: Use *edns-subnet-allow-list*.

- String
- Default: (empty)
- YAML setting does not exist

5.1.56 edns-subnet-allow-list

New in version 4.5.0.

- Comma separated list of strings
- Default: (empty)
- YAML setting: *outgoing.edns_subnet_allow_list*

List of netmasks and domains that **EDNS Client Subnet** should be enabled for in outgoing queries.

For example, an EDNS Client Subnet option containing the address of the initial requestor (but see *ecs-add-for*) will be added to an outgoing query sent to server 192.0.2.1 for domain X if 192.0.2.1 matches one of the supplied netmasks, or if X matches one of the supplied domains. The initial requestor address will be truncated to 24 bits for IPv4 (see *ecs-ipv4-bits*) and to 56 bits for IPv6 (see *ecs-ipv6-bits*), as recommended in the privacy section of RFC 7871.

Note that this setting describes the destination of outgoing queries, not the sources of incoming queries, nor the subnets described in the EDNS Client Subnet option.

By default, this option is empty, meaning no EDNS Client Subnet information is sent.

5.1.57 entropy-source

Changed in version 4.9.0: This setting is no longer used.

- String
- Default: /dev/urandom
- YAML setting does not exist

PowerDNS can read entropy from a (hardware) source. This is used for generating random numbers which are very hard to predict. Generally on UNIX platforms, this source will be `/dev/urandom`, which will always supply random numbers, even if entropy is lacking. Change to `/dev/random` if PowerDNS should block waiting for enough entropy to arrive.

5.1.58 etc-hosts-file

- String
- Default: /etc/hosts
- YAML setting: *recursor.etc_hosts_file*

The path to the /etc/hosts file, or equivalent. This file can be used to serve data authoritatively using *export-etc-hosts*.

5.1.59 event-trace-enabled

New in version 4.6.0.

- Integer
- Default: 0
- YAML setting: *recursor.event_trace_enabled*

Enable the recording and logging of ref:*event traces*. This is an experimental feature and subject to change. Possible values are 0: (disabled), 1 (add information to protobuf logging messages) and 2 (write to log) and 3 (both).

5.1.60 export-etc-hosts

- Boolean
- Default: no
- YAML setting: *recursor.export_etc_hosts*

If set, this flag will export the host names and IP addresses mentioned in `/etc/hosts`.

5.1.61 export-etc-hosts-search-suffix

- String
- Default: (empty)
- YAML setting: *recursor.export_etc_hosts_search_suffix*

If set, all hostnames in the *export-etc-hosts* file are loaded in canonical form, based on this suffix, unless the name contains a '.', in which case the name is unchanged. So an entry called 'pc' with `export-etc-hosts-search-suffix='home.com'` will lead to the generation of 'pc.home.com' within the recursor. An entry called 'server1.home' will be stored as 'server1.home', regardless of this setting.

5.1.62 extended-resolution-errors

New in version 4.5.0.

Changed in version 5.0.0: Default changed to enabled, previously it was disabled.

- Boolean
- Default: yes
- YAML setting: *recursor.extended_resolution_errors*

If set, the recursor will add an EDNS Extended Error (**RFC 8914**) to responses when resolution failed, like DNSSEC validation errors, explaining the reason it failed. This setting is not needed to allow setting custom error codes from Lua or from a RPZ hit.

5.1.63 forward-zones

- Comma separated list of 'zonename=IP' pairs
- Default: (empty)
- YAML setting: *recursor.forward_zones*

Queries for zones listed here will be forwarded to the IP address listed. i.e.

```
forward-zones=example.org=203.0.113.210, powerdns.com=2001:DB8::BEEF:5
```

Multiple IP addresses can be specified and port numbers other than 53 can be configured:

```
forward-zones=example.org=203.0.113.210:5300;127.0.0.1, powerdns.com=127.0.0.1;198.
↪51.100.10:530;[2001:DB8::1:3]:5300
```

Forwarded queries have the `recursion desired (RD)` bit set to `0`, meaning that this setting is intended to forward queries to authoritative servers. If an `NS` record set for a subzone of the forwarded zone is learned, that record set will be used to determine addresses for name servers of the subzone. This allows e.g. a forward to a local authoritative server holding a copy of the root zone, delegations received from that server will work.

IMPORTANT: When using DNSSEC validation (which is default), forwards to non-delegated (e.g. internal) zones that have a DNSSEC signed parent zone will validate as Bogus. To prevent this, add a Negative Trust Anchor (NTA) for this zone in the *lua-config-file* with `addNTA('your.zone', 'A comment')`. If this forwarded zone is signed, instead of adding NTA, add the DS record to the *lua-config-file*. See the *DNSSEC in the PowerDNS Recursor* information.

5.1.64 forward-zones-file

Changed in version 4.0.0: (Old style settings only) Comments are allowed, everything behind # is ignored.

Changed in version 4.6.0: (Old style settings only) Zones prefixed with a ^ are added to the *allow-notify-for* list. Both prefix characters can be used if desired, in any order.

- String
- Default: (empty)
- YAML setting: *recursor.forward_zones_file*

Same as *forward-zones*, parsed from a file. Only 1 zone is allowed per line, specified as follows:

```
example.org=203.0.113.210, 192.0.2.4:5300
```

Zones prefixed with a + are treated as with *forward-zones-recurse*. Default behaviour without + is as with *forward-zones*.

The DNSSEC notes from *forward-zones* apply here as well.

5.1.65 forward-zones-recurse

- Comma separated list of 'zonename=IP' pairs
- Default: (empty)
- YAML setting: *recursor.forward_zones_recurse*

Like regular *forward-zones*, but forwarded queries have the `recursion desired (RD)` bit set to `1`, meaning that this setting is intended to forward queries to other recursive servers. In contrast to regular forwarding, the rule that delegations of the forwarded subzones are respected is not active. This is because we rely on the forwarder to resolve the query fully.

See *forward-zones* for additional options (such as supplying multiple recursive servers) and an important note about DNSSEC.

5.1.66 gettag-needs-edns-options

New in version 4.1.0.

- Boolean
- Default: no
- YAML setting: *incoming.gettag_needs_edns_options*

If set, EDNS options in incoming queries are extracted and passed to the `gettag()` hook in the `ednsoptions` table.

5.1.67 hint-file

Changed in version 4.6.2: Introduced the value `no` to disable root-hints processing.

Changed in version 4.9.0: Introduced the value `no-refresh` to disable both root-hints processing and periodic refresh of the cached root *NS* records.

- String
- Default: (empty)
- YAML setting: *recursor.hint_file*

If set, the root-hints are read from this file. If empty, the default built-in root hints are used.

In some special cases, processing the root hints is not needed, for example when forwarding all queries to another recursor. For these special cases, it is possible to disable the processing of root hints by setting the value to `no` or `no-refresh`. See *Handling of root hints* for more information on root hints handling.

5.1.68 ignore-unknown-settings

- Comma separated list of strings
- Default: (empty)
- YAML setting: *recursor.ignore_unknown_settings*

Names of settings to be ignored while parsing configuration files, if the setting name is unknown to PowerDNS.

Useful during upgrade testing.

5.1.69 `include-dir`

- String
- Default: (empty)
- YAML setting: *recursor.include_dir*

Directory to scan for additional config files. All files that end with `.conf` are loaded in order using `POSIX` as locale.

5.1.70 `latency-statistic-size`

- Integer
- Default: 10000
- YAML setting: *recursor.latency_statistic_size*

Indication of how many queries will be averaged to get the average latency reported by the 'qa-latency' metric.

5.1.71 `local-address`

- Comma separated list or IPs of IP:port combinations
- Default: 127.0.0.1
- YAML setting: *incoming.listen*

Local IP addresses to which we bind. Each address specified can include a port number; if no port is included then the *local-port* port will be used for that address. If a port number is specified, it must be separated from the address with a ':'; for an IPv6 address the address must be enclosed in square brackets.

Examples:

```
local-address=127.0.0.1 ::1
local-address=0.0.0.0:5353
local-address=[::]:8053
local-address=127.0.0.1:53, [::1]:5353
```

5.1.72 `local-port`

- Integer
- Default: 53
- YAML setting: *incoming.port*

Local port to bind to. If an address in *local-address* does not have an explicit port, this port is used.

5.1.73 `log-timestamp`

- Boolean
- Default: yes
- YAML setting: *logging.timestamp*

5.1.74 non-local-bind

- Boolean
- Default: no
- YAML setting: *incoming.non_local_bind*

Bind to addresses even if one or more of the *local-address*'s do not exist on this server. Setting this option will enable the needed socket options to allow binding to non-local addresses. This feature is intended to facilitate ip-failover setups, but it may also mask configuration issues and for this reason it is disabled by default.

5.1.75 loglevel

Changed in version 5.0.0: Previous version would not allow setting a level below `3 (error)`.

- Integer
- Default: 6
- YAML setting: *logging.loglevel*

Amount of logging. The higher the number, the more lines logged. Corresponds to `syslog` level values (e.g. 0 = `emergency`, 1 = `alert`, 2 = `critical`, 3 = `error`, 4 = `warning`, 5 = `notice`, 6 = `info`, 7 = `debug`). Each level includes itself plus the lower levels before it. Not recommended to set this below 3. If *quiet* is `no/false`, *loglevel* will be minimally set to `6 (info)`.

5.1.76 log-common-errors

- Boolean
- Default: no
- YAML setting: *logging.common_errors*

Some DNS errors occur rather frequently and are no cause for alarm.

5.1.77 log-rpz-changes

New in version 4.1.0.

- Boolean
- Default: no
- YAML setting: *logging.rpz_changes*

Log additions and removals to RPZ zones at Info (6) level instead of Debug (7).

5.1.78 logging-facility

- String
- Default: (empty)
- YAML setting: *logging.facility*

If set to a digit, logging is performed under this LOCAL facility. See *Logging*. Do not pass names like 'local0'!

5.1.79 lowercase-outgoing

- Boolean
- Default: no
- YAML setting: *outgoing.lowercase*

Set to true to lowercase the outgoing queries. When set to 'no' (the default) a query from a client using mixed case in the DNS labels (such as a user entering mixed-case names or draft-vixie-dnsext-dns0x20-00), PowerDNS preserves the case of the query. Broken authoritative servers might give a wrong or broken answer on this encoding. Setting `lowercase-outgoing` to 'yes' makes the PowerDNS Recursor lowercase all the labels in the query to the authoritative servers, but still return the proper case to the client requesting.

5.1.80 lua-config-file

- String
- Default: (empty)
- YAML setting: *recursor.lua_config_file*

If set, and Lua support is compiled in, this will load an additional configuration file for newer features and more complicated setups. See *Advanced Configuration Using Lua* for the options that can be set in this file.

5.1.81 lua-global-include-dir

- String
- Default: (empty)
- YAML setting: *recursor.lua_global_include_dir*

When creating a Lua context, all *.lua files in the directory are loaded to the Lua context.

5.1.82 lua-dns-script

- String
- Default: (empty)
- YAML setting: *recursor.lua_dns_script*

Path to a lua file to manipulate the Recursor's answers. See *Scripting PowerDNS Recursor* for more information.

5.1.83 lua-maintenance-interval

New in version 4.2.0.

- Integer
- Default: 1
- YAML setting: *recursor.lua_maintenance_interval*

The interval between calls to the Lua user defined *maintenance()* function in seconds. See *Maintenance callback*

5.1.84 max-busy-dot-probes

New in version 4.7.0.

- Integer
- Default: 0
- YAML setting: *outgoing.max_busy_dot_probes*

Limit the maximum number of simultaneous DoT probes the Recursor will schedule. The default value 0 means no DoT probes are scheduled.

DoT probes are used to check if an authoritative server's IP address supports DoT. If the probe determines an IP address supports DoT, the Recursor will use DoT to contact it for subsequent queries until a failure occurs. After a failure, the Recursor will stop using DoT for that specific IP address for a while. The results of probes are remembered and can be viewed by the `rec_control dump-dot-probe-map` command. If the maximum number of pending probes is reached, no probes will be scheduled, even if no DoT status is known for an address. If the result of a probe is not yet available, the Recursor will contact the authoritative server in the regular way, unless an authoritative server is configured to be contacted over DoT always using *dot-to-auth-names*. In that case no probe will be scheduled.

Note: DoT probing is an experimental feature. Please test thoroughly to determine if it is suitable in your specific production environment before enabling.

5.1.85 max-cache-bogus-ttl

New in version 4.2.0.

- Integer
- Default: 3600
- YAML setting: *recordcache.max_cache_bogus_ttl*

Maximum number of seconds to cache an item in the DNS cache (negative or positive) if its DNSSEC validation failed, no matter what the original TTL specified, to reduce the impact of a broken domain.

5.1.86 max-cache-entries

- Integer
- Default: 1000000
- YAML setting: *recordcache.max_entries*

Maximum number of DNS record cache entries, shared by all threads since 4.4.0. Each entry associates a name and type with a record set. The size of the negative cache is 10% of this number.

5.1.87 max-cache-ttl

Changed in version 4.1.0: The minimum value of this setting is 15. i.e. setting this to lower than 15 will make this value 15.

- Integer
- Default: 86400
- YAML setting: *recordcache.max_ttl*

Maximum number of seconds to cache an item in the DNS cache, no matter what the original TTL specified. This value also controls the refresh period of cached root data. See *Handling of root hints* for more information on this.

5.1.88 max-concurrent-requests-per-tcp-connection

New in version 4.3.0.

- Integer
- Default: 10
- YAML setting: *incoming.max_concurrent_requests_per_tcp_connection*

Maximum number of incoming requests handled concurrently per tcp connection. This number must be larger than 0 and smaller than 65536 and also smaller than *max-mthreads*.

5.1.89 max-chain-length

New in version 5.1.0.

- Integer
- Default: 0
- YAML setting: *recursor.max_chain_length*

The maximum number of queries that can be attached to an outgoing request chain. Attaching requests to a chain saves on outgoing queries, but the processing of a chain when the reply to the outgoing query comes in might result in a large outgoing traffic spike. Reducing the maximum chain length mitigates this. If this value is zero, no maximum is enforced, though the maximum number of mthreads (*max-mthreads*) also limits the chain length.

5.1.90 max-include-depth

New in version 4.6.0.

- Integer
- Default: 20
- YAML setting: *recursor.max_include_depth*

Maximum number of nested `$INCLUDE` directives while processing a zone file. Zero mean no `$INCLUDE` directives will be accepted.

5.1.91 max-generate-steps

New in version 4.3.0.

- Integer
- Default: 0
- YAML setting: *recursor.max_generate_steps*

Maximum number of steps for a '$GENERATE' directive when parsing a zone file. This is a protection measure to prevent consuming a lot of CPU and memory when untrusted zones are loaded. Default to 0 which means unlimited.

5.1.92 max-mthreads

- Integer
- Default: 2048
- YAML setting: *recursor.max_mthreads*

Maximum number of simultaneous MTasker threads, per worker thread.

5.1.93 max-packetcache-entries

- Integer
- Default: 500000
- YAML setting: *packetcache.max_entries*

Maximum number of Packet Cache entries. Sharded and shared by all threads since 4.9.0.

5.1.94 max-qperq

Changed in version 5.1.0: The default used to be 60, with an extra allowance if qname minimization was enabled. Having better algorithms allows for a lower default limit.

- Integer
- Default: 50
- YAML setting: *outgoing.max_qperq*

The maximum number of outgoing queries that will be sent out during the resolution of a single client query. This is used to avoid cycles resolving names.

5.1.95 max-cnames-followed

New in version 5.1.0.

- Integer
- Default: 10
- YAML setting: *recursor.max_cnames_followed*

Maximum length of a CNAME chain. If a CNAME chain exceeds this length, a `ServFail` answer will be returned. Previously, this limit was fixed at 10.

5.1.96 max-ns-address-qperq

New in version 4.1.16.

New in version 4.2.2.

New in version 4.3.1.

- Integer
- Default: 10
- YAML setting: *outgoing.max_ns_address_qperq*

The maximum number of outgoing queries with empty replies for resolving nameserver names to addresses we allow during the resolution of a single client query. If IPv6 is enabled, an A and a AAAA query for a name counts as 1. If a zone publishes more than this number of NS records, the limit is further reduced for that zone by lowering it by the number of NS records found above the *max-ns-address-qperq* value. The limit wil not be reduced to a number lower than 5.

5.1.97 max-ns-per-resolve

New in version 4.8.0.

New in version 4.7.3.

New in version 4.6.4.

New in version 4.5.11.

- Integer
- Default: 13
- YAML setting: *outgoing.max_ns_per_resolve*

The maximum number of NS records that will be considered to select a nameserver to contact to resolve a name. If a zone has more than *max-ns-per-resolve* NS records, a random sample of this size will be used. If *max-ns-per-resolve* is zero, no limit applies.

5.1.98 `max-negative-ttl`

- Integer
- Default: 3600
- YAML setting: *recordcache.max_negative_ttl*

A query for which there is authoritatively no answer is cached to quickly deny a record's existence later on, without putting a heavy load on the remote server. In practice, caches can become saturated with hundreds of thousands of hosts which are tried only once. This setting, which defaults to 3600 seconds, puts a maximum on the amount of time negative entries are cached.

5.1.99 `max-recursion-depth`

Changed in version 4.1.0: Before 4.1.0, this settings was unlimited.

Changed in version 4.9.0: Before 4.9.0 this setting's default was 40 and the limit on `CNAME` chains (fixed at 16) acted as a bound on he recursion depth.

- Integer
- Default: 16
- YAML setting: *recursor.max_recursion_depth*

Total maximum number of internal recursion calls the server may use to answer a single query. 0 means unlimited. The value of *stack-size* should be increased together with this one to prevent the stack from overflowing. If *qname-minimization* is enabled, the fallback code in case of a failing resolve is allowed an additional *max-recursion-depth/2*.

5.1.100 `max-tcp-clients`

- Integer
- Default: 128
- YAML setting: *incoming.max_tcp_clients*

Maximum number of simultaneous incoming TCP connections allowed.

5.1.101 `max-tcp-per-client`

- Integer
- Default: 0
- YAML setting: *incoming.max_tcp_per_client*

Maximum number of simultaneous incoming TCP connections allowed per client (remote IP address). 0 means unlimited.

5.1.102 max-tcp-queries-per-connection

New in version 4.1.0.

- Integer
- Default: 0
- YAML setting: *incoming.max_tcp_queries_per_connection*

Maximum number of DNS queries in a TCP connection. 0 means unlimited.

5.1.103 max-total-msec

- Integer
- Default: 7000
- YAML setting: *recursor.max_total_msec*

Total maximum number of milliseconds of wallclock time the server may use to answer a single query. 0 means unlimited.

5.1.104 max-udp-queries-per-round

New in version 4.1.4.

- Integer
- Default: 10000
- YAML setting: *incoming.max_udp_queries_per_round*

Under heavy load the recursor might be busy processing incoming UDP queries for a long while before there is no more of these, and might therefore neglect scheduling new `mthreads`, handling responses from authoritative servers or responding to *rec_control* requests. This setting caps the maximum number of incoming UDP DNS queries processed in a single round of looping on `recvmsg()` after being woken up by the multiplexer, before returning back to normal processing and handling other events.

5.1.105 minimum-ttl-override

Changed in version 4.5.0: Old versions used default 0.

- Integer
- Default: 1
- YAML setting: *recursor.minimum_ttl_override*

This setting artificially raises all TTLs to be at least this long. Setting this to a value greater than 1 technically is an RFC violation, but might improve performance a lot. Using a value of 0 impacts performance of TTL 0 records greatly, since it forces the recursor to contact authoritative servers each time a client requests them. Can be set at runtime using `rec_control set-minimum-ttl 3600`.

5.1.106 new-domain-tracking

New in version 4.2.0.

- Boolean
- Default: no
- YAML setting: *nod.tracking*

Whether to track newly observed domains, i.e. never seen before. This is a probabilistic algorithm, using a stable bloom filter to store records of previously seen domains. When enabled for the first time, all domains will appear to be newly observed, so the feature is best left enabled for e.g. a week or longer before using the results. Note that this feature is optional and must be enabled at compile-time, thus it may not be available in all pre-built packages. If protobuf is enabled and configured, then the newly observed domain status will appear as a flag in Response messages.

5.1.107 new-domain-log

New in version 4.2.0.

- Boolean
- Default: yes
- YAML setting: *nod.log*

If a newly observed domain is detected, log that domain in the recursor log file. The log line looks something like:

```
Jul 18 11:31:25 Newly observed domain nod=sdfoijdfio.com
```

5.1.108 new-domain-lookup

New in version 4.2.0.

- String
- Default: (empty)
- YAML setting: *nod.lookup*

If a domain is specified, then each time a newly observed domain is detected, the recursor will perform an A record lookup of '<newly observed domain>.<lookup domain>'. For example if 'new-domain-lookup' is configured as 'nod.powerdns.com', and a new domain 'example.com' is detected, then an A record lookup will be made for 'example.com.nod.powerdns.com'. This feature gives a way to share the newly observed domain with partners, vendors or security teams. The result of the DNS lookup will be ignored by the recursor.

5.1.109 new-domain-db-size

New in version 4.2.0.

- Integer
- Default: 67108864
- YAML setting: *nod.db_size*

The default size of the stable bloom filter used to store previously observed domains is 67108864. To change the number of cells, use this setting. For each cell, the SBF uses 1 bit of memory, and one byte of disk for the persistent file. If there are already persistent files saved to disk, this setting will have no effect unless you remove the existing files.

5.1.110 new-domain-history-dir

New in version 4.2.0.

- String
- Default: Determined by distribution
- YAML setting: *nod.history_dir*

This setting controls which directory is used to store the on-disk cache of previously observed domains.

The default depends on `LOCALSTATEDIR` when building the software. Usually this comes down to `/var/lib/pdns-recursor/nod` or `/usr/local/var/lib/pdns-recursor/nod`).

The newly observed domain feature uses a stable bloom filter to store a history of previously observed domains. The data structure is synchronized to disk every 10 minutes, and is also initialized from disk on startup. This ensures that previously observed domains are preserved across recursor restarts. If you change the new-domain-db-size setting, you must remove any files from this directory.

5.1.111 `new-domain-db-snapshot-interval`

New in version 5.1.0.

- Integer
- Default: 600
- YAML setting: *nod.db_snapshot_interval*

Interval (in seconds) to write the NOD and UDR DB snapshots. Set to zero to disable snapshot writing.',

5.1.112 `new-domain-whitelist`

New in version 4.2.0.

Deprecated since version 4.5.0: Use *new-domain-ignore-list*.

- String
- Default: (empty)
- YAML setting does not exist

5.1.113 `new-domain-ignore-list`

New in version 4.5.0.

- Comma separated list of strings
- Default: (empty)
- YAML setting: *nod.ignore_list*

This setting is a list of all domains (and implicitly all subdomains) that will never be considered a new domain. For example, if the domain 'example.com' is in the list, then 'foo.bar.example.com' will never be considered a new domain. One use-case for the ignore list is to never reveal details of internal subdomains via the new-domain-lookup feature.

5.1.114 `new-domain-ignore-list-file`

New in version 5.1.0.

- String
- Default: (empty)
- YAML setting: *nod.ignore_list_file*

Path to a file with a list of domains. File should have one domain per line, with no extra characters or comments. See *new-domain-ignore-list*.

5.1.115 new-domain-pb-tag

New in version 4.2.0.

- String
- Default: pdns-nod
- YAML setting: *nod.pb_tag*

If protobuf is configured, then this tag will be added to all protobuf response messages when a new domain is observed.

5.1.116 network-timeout

- Integer
- Default: 1500
- YAML setting: *outgoing.network_timeout*

Number of milliseconds to wait for a remote authoritative server to respond. If the number of concurrent requests is high, the :program:Recursor uses a lower value.

5.1.117 non-resolving-ns-max-fails

New in version 4.5.0.

- Integer
- Default: 5
- YAML setting: *outgoing.non_resolving_ns_max_fails*

Number of failed address resolves of a nameserver name to start throttling it, 0 is disabled. Nameservers matching *dont-throttle-names* will not be throttled.

5.1.118 non-resolving-ns-throttle-time

New in version 4.5.0.

- Integer
- Default: 60
- YAML setting: *outgoing.non_resolving_ns_throttle_time*

Number of seconds to throttle a nameserver with a name failing to resolve.

5.1.119 nothing-below-nxdomain

New in version 4.3.0.

- String
- Default: dnssec
- YAML setting: *recursor.nothing_below_nxdomain*
- One of `no`, `dnssec`, `yes`.

The type of **RFC 8020** handling using cached NXDOMAIN responses. This RFC specifies that NXDOMAIN means that the DNS tree under the denied name MUST be empty. When an NXDOMAIN exists in the cache for a shorter name than the qname, no lookup is done and an NXDOMAIN is sent to the client.

For instance, when `foo.example.net` is negatively cached, any query matching `*.foo.example.net` will be answered with NXDOMAIN directly without consulting authoritative servers.

`no` No **RFC 8020** processing is done.

`dnssec` **RFC 8020** processing is only done using cached NXDOMAIN records that are DNSSEC validated.

`yes` **RFC 8020** processing is done using any non-Bogus NXDOMAIN record available in the cache.

5.1.120 `nsec3-max-iterations`

New in version 4.1.0.

Changed in version 4.5.2: Default is now 150, was 2500 before.

Changed in version 5.0.0: Default is now 50, was 150 before.

- Integer
- Default: 50
- YAML setting: *dnssec.nsec3_max_iterations*

Maximum number of iterations allowed for an NSEC3 record. If an answer containing an NSEC3 record with more iterations is received, its DNSSEC validation status is treated as `Insecure`.

5.1.121 `max-rrsigs-per-record`

New in version 5.0.2.

New in version 4.9.3.

New in version 4.8.6.

- Integer
- Default: 2
- YAML setting: *dnssec.max_rrsigs_per_record*

Maximum number of RRSIGs we are willing to cryptographically check when validating a given record. Expired or not yet incepted RRSIGs do not count toward to this limit.

5.1.122 `max-nsec3s-per-record`

New in version 5.0.2.

New in version 4.9.3.

New in version 4.8.6.

- Integer
- Default: 10
- YAML setting: *dnssec.max_nsec3s_per_record*

Maximum number of NSEC3s to consider when validating a given denial of existence.

5.1.123 max-signature-validations-per-query

New in version 5.0.2.

New in version 4.9.3.

New in version 4.8.6.

- Integer
- Default: 30
- YAML setting: *dnssec.max_signature_validations_per_query*

Maximum number of RRSIG signatures we are willing to validate per incoming query.

5.1.124 max-nsec3-hash-computations-per-query

New in version 5.0.2.

New in version 4.9.3.

New in version 4.8.6.

- Integer
- Default: 600
- YAML setting: *dnssec.max_nsec3_hash_computations_per_query*

Maximum number of NSEC3 hashes that we are willing to compute during DNSSEC validation, per incoming query.

5.1.125 aggressive-cache-max-nsec3-hash-cost

New in version 5.0.2.

New in version 4.9.3.

New in version 4.8.6.

- Integer
- Default: 150
- YAML setting: *dnssec.aggressive_cache_max_nsec3_hash_cost*

Maximum estimated NSEC3 cost for a given query to consider aggressive use of the NSEC3 cache. The cost is estimated based on a heuristic taking the zone's NSEC3 salt and iterations parameters into account, as well at the number of labels of the requested name. For example a query for a name like a.b.c.d.e.f.example.com. in an example.com zone. secured with NSEC3 and 10 iterations (NSEC3 iterations count of 9) and an empty salt will have an estimated worst-case cost of 10 (iterations) * 6 (number of labels) = 60. The aggressive NSEC cache is an optimization to reduce the number of queries to authoritative servers, which is especially useful when a zone is under pseudo-random subdomain attack, and we want to skip it the zone parameters make it expensive.

5.1.126 max-ds-per-zone

New in version 5.0.2.

New in version 4.9.3.

New in version 4.8.6.

- Integer
- Default: 8

- YAML setting: *dnssec.max_ds_per_zone*

Maximum number of DS records to consider when validating records inside a zone.

5.1.127 max-dnskeys

New in version 5.0.2.

New in version 4.9.3.

New in version 4.8.6.

- Integer
- Default: 2
- YAML setting: *dnssec.max_dnskeys*

Maximum number of DNSKEYs with the same algorithm and tag to consider when validating a given record. Setting this value to 1 effectively denies DNSKEY tag collisions in a zone.

5.1.128 packetcache-ttl

Changed in version 4.9.0: The default was changed from 3600 (1 hour) to 86400 (24 hours).

- Integer
- Default: 86400
- YAML setting: *packetcache.ttl*

Maximum number of seconds to cache an item in the packet cache, no matter what the original TTL specified.

5.1.129 packetcache-negative-ttl

New in version 4.9.0.

- Integer
- Default: 60
- YAML setting: *packetcache.negative_ttl*

Maximum number of seconds to cache an `NxDomain` or `NoData` answer in the packetcache. This setting's maximum is capped to *packetcache-ttl*. i.e. setting `packetcache-ttl=15` and keeping `packetcache-negative-ttl` at the default will lower `packetcache-negative-ttl` to `15`.

5.1.130 packetcache-servfail-ttl

'versionchanged': ('4.0.0', "This setting's maximum is capped to *packetcache-ttl*. i.e. setting `packetcache-ttl=15` and keeping `packetcache-servfail-ttl` at the default will lower `packetcache-servfail-ttl` to `15`.")

- Integer
- Default: 60
- YAML setting: *packetcache.servfail_ttl*

Maximum number of seconds to cache an answer indicating a failure to resolve in the packet cache. Before version 4.6.0 only `ServFail` answers were considered as such. Starting with 4.6.0, all responses with a code other than `NoError` and `NXDomain`, or without records in the answer and authority sections, are considered as a failure to resolve. Since 4.9.0, negative answers are handled separately from resolving failures.

5.1.131 packetcache-shards

New in version 4.9.0.

- Integer
- Default: 1024
- YAML setting: *packetcache.shards*

Sets the number of shards in the packet cache. If you have high contention as reported by `packetcache-contented/packetcache-acquired`, you can try to enlarge this value or run with fewer threads.

5.1.132 pdns-distributes-queries

Changed in version 4.9.0: Default changed to `no`, previously it was `yes`.

- Boolean
- Default: no
- YAML setting: *incoming.pdns_distributes_queries*

If set, PowerDNS will use distinct threads to listen to client sockets and distribute that work to worker-threads using a hash of the query. This feature should maximize the cache hit ratio on versions before 4.9.0. To use more than one thread set *distributor-threads* in version 4.2.0 or newer. Enabling should improve performance on systems where *reuseport* does not have the effect of balancing the queries evenly over multiple worker threads.

5.1.133 protobuf-use-kernel-timestamp

New in version 4.2.0.

- Boolean
- Default: no
- YAML setting: *logging.protobuf_use_kernel_timestamp*

Whether to compute the latency of responses in protobuf messages using the timestamp set by the kernel when the query packet was received (when available), instead of computing it based on the moment we start processing the query.

5.1.134 proxy-protocol-from

New in version 4.4.0.

Changed in version 5.0.5: YAML settings only: previously this was defined as a string instead of a sequence

- Comma separated list of IP addresses or subnets, negation supported
- Default: (empty)
- YAML setting: *incoming.proxy_protocol_from*

Ranges that are required to send a Proxy Protocol version 2 header in front of UDP and TCP queries, to pass the original source and destination addresses and ports to the recursor, as well as custom values. Queries that are not prefixed with such a header will not be accepted from clients in these ranges. Queries prefixed by headers from clients that are not listed in these ranges will be dropped.

Note that once a Proxy Protocol header has been received, the source address from the proxy header instead of the address of the proxy will be checked against the *allow-from* ACL.

The dnsdist docs have more information about the PROXY protocol.

5.1.135 proxy-protocol-exceptions

New in version 5.1.0.

- Comma separated list or IPs of IP:port combinations
- Default: (empty)
- YAML setting: *incoming.proxy_protocol_exceptions*

If set, clients sending from an address in *proxy-protocol-from* to a address:port listed here are excluded from using the Proxy Protocol. If no port is specified, port 53 is assumed. This is typically used to provide an easy to use address and port to send debug queries to.

5.1.136 proxy-protocol-maximum-size

New in version 4.4.0.

- Integer
- Default: 512
- YAML setting: *incoming.proxy_protocol_maximum_size*

The maximum size, in bytes, of a Proxy Protocol payload (header, addresses and ports, and TLV values). Queries with a larger payload will be dropped.

5.1.137 public-suffix-list-file

New in version 4.2.0.

- String
- Default: (empty)
- YAML setting: *recursor.public_suffix_list_file*

Path to the Public Suffix List file, if any. If set, PowerDNS will try to load the Public Suffix List from this file instead of using the built-in list. The PSL is used to group the queries by relevant domain names when displaying the top queries.

5.1.138 qname-minimization

New in version 4.3.0.

- Boolean
- Default: yes
- YAML setting: *recursor.qname_minimization*

Enable Query Name Minimization. This implements a relaxed form of Query Name Mimimization as described in **RFC 9156**.

5.1.139 qname-max-minimize-count

New in version 5.0.0.

- Integer
- Default: 10
- YAML setting: *recursor.qname_max_minimize_count*

`Max minimize count` parameter, described in **RFC 9156**. This is the maximum number of iterations of the Query Name Minimization Algorithm.

5.1.140 qname-minimize-one-label

New in version 5.0.0.

- Integer
- Default: 4
- YAML setting: *recursor.qname_minimize_one_label*

`Minimize one label` parameter, described in **RFC 9156**. The value for the number of iterations of the Query Name Minimization Algorithm that should only have one label appended. This value has precedence over *qname-max-minimize-count*.

5.1.141 query-local-address

Changed in version 4.4.0: IPv6 addresses can be set with this option as well.

- Comma separated list of IP addresses or subnets, negation supported
- Default: 0.0.0.0
- YAML setting: *outgoing.source_address*

Note: While subnets and their negations are syntactically accepted, the handling of subnets has not been implemented yet. Only individual IP addresses can be listed.

Send out local queries from this address, or addresses. By adding multiple addresses, increased spoofing resilience is achieved. When no address of a certain address family is configured, there are *no* queries sent with that address family. In the default configuration this means that IPv6 is not used for outgoing queries.

5.1.142 quiet

- Boolean
- Default: yes
- YAML setting: *logging.quiet*

Don't log queries.

5.1.143 record-cache-locked-ttl-perc

New in version 4.8.0.

- Integer
- Default: 0
- YAML setting: *recordcache.locked_ttl_perc*

Replace record sets in the record cache only after this percentage of the original TTL has passed. The PowerDNS Recursor already has several mechanisms to protect against spoofing attempts. This adds an extra layer of protection—as it limits the window of time cache updates are accepted—at the cost of a less efficient record cache.

The default value of 0 means no extra locking occurs. When non-zero, record sets received (e.g. in the Additional Section) will not replace existing record sets in the record cache until the given percentage of the original TTL has expired. A value of 100 means only expired record sets will be replaced.

There are a few cases where records will be replaced anyway:

- Record sets that are expired will always be replaced.
- Authoritative record sets will replace unauthoritative record sets unless DNSSEC validation of the new record set failed.
- If the new record set belongs to a DNSSEC-secure zone and successfully passed validation it will replace an existing entry.
- Record sets produced by *refresh-on-ttl-perc* tasks will also replace existing record sets.

5.1.144 `record-cache-shards`

New in version 4.4.0.

- Integer
- Default: 1024
- YAML setting: *recordcache.shards*

Sets the number of shards in the record cache. If you have high contention as reported by `record-cache-contented/record-cache-acquired`, you can try to enlarge this value or run with fewer threads.

5.1.145 `refresh-on-ttl-perc`

New in version 4.5.0.

- Integer
- Default: 0
- YAML setting: *recordcache.refresh_on_ttl_perc*

Sets the 'refresh almost expired' percentage of the record cache. Whenever a record is fetched from the packet or record cache and only `refresh-on-ttl-perc` percent or less of its original TTL is left, a task is queued to refetch the name/type combination to update the record cache. In most cases this causes future queries to always see a non-expired record cache entry. A typical value is 10. If the value is zero, this functionality is disabled.

5.1.146 `reuseport`

Changed in version 4.9.0: The default is changed to `yes`, previously it was `no`. If `SO_REUSEPORT` support is not available, the setting defaults to `no`.

- Boolean
- Default: yes
- YAML setting: *incoming.reuseport*

If `SO_REUSEPORT` support is available, allows multiple threads and processes to open listening sockets for the same port.

Since 4.1.0, when *pdns-distributes-queries* is disabled and *reuseport* is enabled, every worker-thread will open a separate listening socket to let the kernel distribute the incoming queries instead of running a distributor thread (which could otherwise be a bottleneck) and avoiding thundering herd issues, thus leading to much higher performance on multi-core boxes.

5.1.147 rng

Changed in version 4.9.0: This setting is no longer used.

- String
- Default: auto
- YAML setting does not exist
- String
- Default: auto

Specify which random number generator to use. Permissible choices are

- auto - choose automatically
- sodium - Use libsodium `randombytes_uniform`
- openssl - Use libcrypto `RAND_bytes`
- getrandom - Use libc getrandom, falls back to urandom if it does not really work
- arc4random - Use BSD `arc4random_uniform`
- urandom - Use `/dev/urandom`
- kiss - Use simple settable deterministic RNG. **FOR TESTING PURPOSES ONLY!**

5.1.148 root-nx-trust

Changed in version 4.0.0: Default is `yes` now, was `no` before 4.0.0

- Boolean
- Default: yes
- YAML setting: *recursor.root_nx_trust*

If set, an NXDOMAIN from the root-servers will serve as a blanket NXDOMAIN for the entire TLD the query belonged to. The effect of this is far fewer queries to the root-servers.

5.1.149 save-parent-ns-set

New in version 4.7.0.

- Boolean
- Default: yes
- YAML setting: *recursor.save_parent_ns_set*

If set, a parent (non-authoritative) `NS` set is saved if it contains more entries than a newly encountered child (authoritative) `NS` set for the same domain. The saved parent `NS` set is tried if resolution using the child `NS` set fails.

5.1.150 security-poll-suffix

- String
- Default: secpoll.powerdns.com.
- YAML setting: *recursor.security_poll_suffix*

Domain name from which to query security update notifications. Setting this to an empty string disables secpoll.

5.1.151 serve-rfc1918

- Boolean
- Default: yes
- YAML setting: *recursor.serve_rfc1918*

This makes the server authoritatively aware of: `10.in-addr.arpa`, `168.192.in-addr.arpa`, `16-31.172.in-addr.arpa`, which saves load on the AS112 servers. Individual parts of these zones can still be loaded or forwarded.

5.1.152 serve-stale-extensions

New in version 4.8.0.

- Integer
- Default: 0
- YAML setting: *recordcache.serve_stale_extensions*

Maximum number of times an expired record's TTL is extended by 30s when serving stale. Extension only occurs if a record cannot be refreshed. A value of 0 means the `Serve Stale` mechanism is not used. To allow records becoming stale to be served for an hour, use a value of 120. See *Serve Stale* for a description of the Serve Stale mechanism.

5.1.153 server-down-max-fails

- Integer
- Default: 64
- YAML setting: *outgoing.server_down_max_fails*

If a server has not responded in any way this many times in a row, no longer send it any queries for *server-down-throttle-time* seconds. Afterwards, we will try a new packet, and if that also gets no response at all, we again throttle for *server-down-throttle-time* seconds. Even a single response packet will drop the block.

5.1.154 server-down-throttle-time

- Integer
- Default: 60
- YAML setting: *outgoing.server_down_throttle_time*

Throttle a server that has failed to respond *server-down-max-fails* times for this many seconds.

5.1.155 bypass-server-throttling-probability

New in version 5.0.0.

- Integer
- Default: 25
- YAML setting: *outgoing.bypass_server_throttling_probability*

This setting determines the probability of a server marked down to be used anyway. A value of `n` means that the chance of a server marked down still being used after it wins speed selection is is `1/n`. If this setting is zero throttled servers will never be selected to be used anyway.

5.1.156 server-id

- String
- Default: *runtime determined*
- YAML setting: *recursor.server_id*

The reply given by The PowerDNS recursor to a query for 'id.server' with its hostname, useful for in clusters. When a query contains the **NSID EDNS0 Option**, this value is returned in the response as the NSID value.

This setting can be used to override the answer given to these queries. Set to 'disabled' to disable NSID and 'id.server' answers.

Query example (where 192.0.2.14 is your server):

```
dig @192.0.2.14 CHAOS TXT id.server.
dig @192.0.2.14 example.com IN A +nsid
```

5.1.157 setgid

- String
- Default: (empty)
- YAML setting: *recursor.setgid*

PowerDNS can change its user and group id after binding to its socket. Can be used for better *security*.

5.1.158 setuid

- String
- Default: (empty)
- YAML setting: *recursor.setuid*

PowerDNS can change its user and group id after binding to its socket. Can be used for better *security*.

5.1.159 signature-inception-skew

New in version 4.1.5.

Changed in version 4.2.0: Default is now 60, was 0 before.

- Integer
- Default: 60
- YAML setting: *dnssec.signature_inception_skew*

Allow the signature inception to be off by this number of seconds. Negative values are not allowed.

5.1.160 single-socket

- Boolean
- Default: no
- YAML setting: *outgoing.single_socket*

Use only a single socket for outgoing queries.

5.1.161 snmp-agent

New in version 4.1.0.

- Boolean
- Default: no
- YAML setting: *snmp.agent*

If set to true and PowerDNS has been compiled with SNMP support, it will register as an SNMP agent to provide statistics and be able to send traps.

5.1.162 snmp-master-socket

New in version 4.1.0.

Deprecated since version 4.5.0: Use *snmp-daemon-socket*.

- String
- Default: (empty)
- YAML setting does not exist

5.1.163 snmp-daemon-socket

New in version 4.5.0.

- String
- Default: (empty)
- YAML setting: *snmp.daemon_socket*

If not empty and `snmp-agent` is set to true, indicates how PowerDNS should contact the SNMP daemon to register as an SNMP agent.

5.1.164 socket-dir

- String
- Default: (empty)
- YAML setting: *recursor.socket_dir*

Where to store the control socket and pidfile. The default depends on `LOCALSTATEDIR` or the `--with-socketdir` setting when building (usually `/var/run` or `/run`).

When using *chroot* the default becomes `/`. The default value is overruled by the `RUNTIME_DIRECTORY` environment variable when that variable has a value (e.g. under systemd).

5.1.165 socket-group

- String
- Default: (empty)
- YAML setting: *recursor.socket_group*

Group and mode of the controlsocket. Owner and group can be specified by name, mode is in octal.

5.1.166 socket-mode

- String
- Default: (empty)
- YAML setting: *recursor.socket_mode*

Mode of the controlsocket. Owner and group can be specified by name, mode is in octal.

5.1.167 socket-owner

- String
- Default: (empty)
- YAML setting: *recursor.socket_owner*

Owner of the controlsocket. Owner and group can be specified by name, mode is in octal.

5.1.168 spoof-nearmiss-max

Changed in version 4.5.0: Older versions used 20 as the default value.

- Integer
- Default: 1
- YAML setting: *recursor.spoof_nearmiss_max*

If set to non-zero, PowerDNS will assume it is being spoofed after seeing this many answers with the wrong id.

5.1.169 stack-cache-size

New in version 4.9.0.

- Integer
- Default: 100
- YAML setting: *recursor.stack_cache_size*

Maximum number of mthread stacks that can be cached for later reuse, per thread. Caching these stacks reduces the CPU load at the cost of a slightly higher memory usage, each cached stack consuming *stack-size* bytes of memory. It makes no sense to cache more stacks than the value of *max-mthreads*, since there will never be more stacks than that in use at a given time.

5.1.170 stack-size

- Integer
- Default: 200000
- YAML setting: *recursor.stack_size*

Size in bytes of the stack of each mthread.

5.1.171 statistics-interval

New in version 4.1.0.

- Integer
- Default: 1800
- YAML setting: *logging.statistics_interval*

Interval between logging statistical summary on recursor performance. Use 0 to disable.

5.1.172 stats-api-blacklist

New in version 4.2.0.

Deprecated since version 4.5.0: Use *stats-api-disabled-list*.

- Comma separated list of strings
- Default:
- YAML setting does not exist

5.1.173 stats-api-disabled-list

New in version 4.5.0.

- Comma separated list of strings
- Default: cache-bytes, packetcache-bytes, special-memory-usage, ecs-v4-response-bits-*, ecs-v6-response-bits-*
- YAML setting: *recursor.stats_api_disabled_list*

A list of comma-separated statistic names, that are disabled when retrieving the complete list of statistics via the API for performance reasons. These statistics can still be retrieved individually by specifically asking for it.

5.1.174 stats-carbon-blacklist

New in version 4.2.0.

Deprecated since version 4.5.0: Use *stats-carbon-disabled-list*.

- Comma separated list of strings
- Default:
- YAML setting does not exist

5.1.175 stats-carbon-disabled-list

New in version 4.5.0.

- Comma separated list of strings
- Default: cache-bytes, packetcache-bytes, special-memory-usage, ecs-v4-response-bits-*, ecs-v6-response-bits-*, cumul-answers-*, cumul-auth4answers-*, cumul-auth6answers-*
- YAML setting: *recursor.stats_carbon_disabled_list*

A list of comma-separated statistic names, that are prevented from being exported via carbon for performance reasons.

5.1.176 stats-rec-control-blacklist

New in version 4.2.0.

Deprecated since version 4.5.0: Use *stats-rec-control-disabled-list*.

- Comma separated list of strings
- Default:
- YAML setting does not exist

5.1.177 stats-rec-control-disabled-list

New in version 4.5.0.

- Comma separated list of strings
- Default: cache-bytes, packetcache-bytes, special-memory-usage, ecs-v4-response-bits-*, ecs-v6-response-bits-*, cumul-answers-*, cumul-auth4answers-*, cumul-auth6answers-*
- YAML setting: *recursor.stats_rec_control_disabled_list*

A list of comma-separated statistic names, that are disabled when retrieving the complete list of statistics via *rec_control get-all*, for performance reasons. These statistics can still be retrieved individually.

5.1.178 stats-ringbuffer-entries

- Integer
- Default: 10000
- YAML setting: *recursor.stats_ringbuffer_entries*

Number of entries in the remotes ringbuffer, which keeps statistics on who is querying your server. Can be read out using `rec_control top-remotes`.

5.1.179 stats-snmp-blacklist

New in version 4.2.0.

Deprecated since version 4.5.0: Use *stats-snmp-disabled-list*.

- Comma separated list of strings
- Default:
- YAML setting does not exist

5.1.180 stats-snmp-disabled-list

New in version 4.5.0.

- Comma separated list of strings
- Default: cache-bytes, packetcache-bytes, special-memory-usage, ecs-v4-response-bits-*, ecs-v6-response-bits-*
- YAML setting: *recursor.stats_snmp_disabled_list*

A list of comma-separated statistic names, that are prevented from being exported via SNMP, for performance reasons.

5.1.181 structured-logging

New in version 4.6.0.

Changed in version 5.1.0: Disabling structured logging is not supported

- Boolean
- Default: yes
- YAML setting: *logging.structured_logging*

Prefer structured logging when both an old style and a structured log messages is available.

5.1.182 structured-logging-backend

New in version 4.8.0.

Changed in version 5.1.0: The JSON backend was added

- String
- Default: default
- YAML setting: *logging.structured_logging_backend*

The backend used for structured logging output. This setting must be set on the command line (`--structured-logging-backend=...`) to be effective. Available backends are:

- `default`: use the traditional logging system to output structured logging information.
- `systemd-journal`: use systemd-journal. When using this backend, provide `-o verbose` or simular output option to `journalctl` to view the full information.
- `json`: JSON objects are written to the standard error stream.

See *Structured Logging Dictionary* for more details.

5.1.183 tcp-fast-open

New in version 4.1.0.

- Integer
- Default: 0
- YAML setting: *incoming.tcp_fast_open*

Enable TCP Fast Open support, if available, on the listening sockets. The numerical value supplied is used as the queue size, 0 meaning disabled. See *TCP Fast Open Support*.

5.1.184 tcp-fast-open-connect

New in version 4.5.0.

- Boolean
- Default: no
- YAML setting: *outgoing.tcp_fast_open_connect*

Enable TCP Fast Open Connect support, if available, on the outgoing connections to authoritative servers. See *TCP Fast Open Support*.

5.1.185 tcp-out-max-idle-ms

New in version 4.6.0.

- Integer
- Default: 10000
- YAML setting: *outgoing.tcp_max_idle_ms*

Time outgoing TCP/DoT connections are left idle in milliseconds or 0 if no limit. After having been idle for this time, the connection is eligible for closing.

5.1.186 tcp-out-max-idle-per-auth

New in version 4.6.0.

- Integer
- Default: 10
- YAML setting: *outgoing.tcp_max_idle_per_auth*

Maximum number of idle outgoing TCP/DoT connections to a specific IP per thread, 0 means do not keep idle connections open.

5.1.187 tcp-out-max-queries

- Integer
- Default: 0
- YAML setting: *outgoing.tcp_max_queries*

Maximum total number of queries per outgoing TCP/DoT connection, 0 means no limit. After this number of queries, the connection is closed and a new one will be created if needed.

5.1.188 tcp-out-max-idle-per-thread

New in version 4.6.0.

- Integer
- Default: 100
- YAML setting: *outgoing.tcp_max_idle_per_thread*

Maximum number of idle outgoing TCP/DoT connections per thread, 0 means do not keep idle connections open.

5.1.189 threads

- Integer
- Default: 2
- YAML setting: *recursor.threads*

Spawn this number of threads on startup.

5.1.190 tcp-threads

New in version 5.0.0.

- Integer
- Default: 1
- YAML setting: *recursor.tcp_threads*

Spawn this number of TCP processing threads on startup.

5.1.191 trace

- String
- Default: no
- YAML setting: *logging.trace*

One of `no`, `yes` or `fail`. If turned on, output impressive heaps of logging. May destroy performance under load. To log only queries resulting in a `ServFail` answer from the resolving process, this value can be set to `fail`, but note that the performance impact is still large. Also note that queries that do produce a result but with a failing DNSSEC validation are not written to the log

5.1.192 udp-source-port-min

New in version 4.2.0.

- Integer
- Default: 1024
- YAML setting: *outgoing.udp_source_port_min*

This option sets the low limit of UDP port number to bind on.

In combination with *udp-source-port-max* it configures the UDP port range to use. Port numbers are randomized within this range on initialization, and exceptions can be configured with *udp-source-port-avoid*

5.1.193 udp-source-port-max

New in version 4.2.0.

- Integer
- Default: 65535
- YAML setting: *outgoing.udp_source_port_max*

This option sets the maximum limit of UDP port number to bind on.

See *udp-source-port-min*.

5.1.194 udp-source-port-avoid

New in version 4.2.0.

- Comma separated list of strings
- Default: 11211
- YAML setting: *outgoing.udp_source_port_avoid*

A list of comma-separated UDP port numbers to avoid when binding. Ex: *5300,11211*

See *udp-source-port-min*.

5.1.195 udp-truncation-threshold

Changed in version 4.2.0: Before 4.2.0, the default was 1680.

- Integer
- Default: 1232
- YAML setting: *incoming.udp_truncation_threshold*

EDNS0 allows for large UDP response datagrams, which can potentially raise performance. Large responses however also have downsides in terms of reflection attacks. This setting limits the accepted size. Maximum value is 65535, but values above 4096 should probably not be attempted.

To know why 1232, see the note at *edns-outgoing-bufsize*.

5.1.196 unique-response-tracking

New in version 4.2.0.

- Boolean
- Default: no
- YAML setting: *nod.unique_response_tracking*

Whether to track unique DNS responses, i.e. never seen before combinations of the triplet (query name, query type, RR[rrname, rrtype, rrdata]). This can be useful for tracking potentially suspicious domains and behaviour, e.g. DNS fast-flux. If protobuf is enabled and configured, then the Protobuf Response message will contain a flag with udr set to true for each RR that is considered unique, i.e. never seen before. This feature uses a probabilistic data structure (stable bloom filter) to track unique responses, which can have false positives as well as false negatives, thus it is a best-effort feature. Increasing the number of cells in the SBF using the unique-response-db-size setting can reduce FPs and FNs.

5.1.197 unique-response-log

New in version 4.2.0.

- Boolean
- Default: yes
- YAML setting: *nod.unique_response_log*

Whether to log when a unique response is detected. The log line looks something like:

Oct 24 12:11:27 Unique response observed: qname=foo.com qtype=A rrtype=AAAA rrname=foo.com rrcontent=1.2.3.4

5.1.198 unique-response-db-size

New in version 4.2.0.

- Integer
- Default: 67108864
- YAML setting: *nod.unique_response_db_size*

The default size of the stable bloom filter used to store previously observed responses is 67108864. To change the number of cells, use this setting. For each cell, the SBF uses 1 bit of memory, and one byte of disk for the persistent file. If there are already persistent files saved to disk, this setting will have no effect unless you remove the existing files.

5.1.199 `unique-response-history-dir`

New in version 4.2.0.

- String
- Default: Determined by distribution
- YAML setting: *nod.unique_response_history_dir*

This setting controls which directory is used to store the on-disk cache of previously observed responses.

The default depends on `LOCALSTATEDIR` when building the software. Usually this comes down to `/var/lib/pdns-recursor/udr` or `/usr/local/var/lib/pdns-recursor/udr`).

The newly observed domain feature uses a stable bloom filter to store a history of previously observed responses. The data structure is synchronized to disk every 10 minutes, and is also initialized from disk on startup. This ensures that previously observed responses are preserved across recursor restarts. If you change the unique-response-db-size, you must remove any files from this directory.

5.1.200 `unique-response-pb-tag`

New in version 4.2.0.

- String
- Default: pdns-udr
- YAML setting: *nod.unique_response_pb_tag*

If protobuf is configured, then this tag will be added to all protobuf response messages when a unique DNS response is observed.

5.1.201 `unique-response-ignore-list`

New in version 5.1.0.

- Comma separated list of strings
- Default: (empty)
- YAML setting: *nod.unique_response_ignore_list*

This setting is a list of all domains (and implicitly all subdomains) that will never be considered for new unique domain responses. For example, if the domain 'example.com' is in the list, then 'foo.bar.example.com' will never be considered for a new unique domain response.

5.1.202 `unique-response-ignore-list-file`

New in version 5.1.0.

- String
- Default: (empty)
- YAML setting: *nod.unique_response_ignore_list_file*

Path to a file with a list of domains. File should have one domain per line, with no extra characters or comments. See *unique-response-ignore-list*.

5.1.203 use-incoming-edns-subnet

- Boolean
- Default: no
- YAML setting: *incoming.use_incoming_edns_subnet*

Whether to process and pass along a received EDNS Client Subnet to authoritative servers. The ECS information will only be sent for netmasks and domains listed in *edns-subnet-allow-list* and will be truncated if the received scope exceeds *ecs-ipv4-bits* for IPv4 or *ecs-ipv6-bits* for IPv6.

5.1.204 version-string

- String
- Default: *runtime determined*
- YAML setting: *recursor.version_string*

By default, PowerDNS replies to the 'version.bind' query with its version number. Security conscious users may wish to override the reply PowerDNS issues.

5.1.205 webserver

- Boolean
- Default: no
- YAML setting: *webservice.webserver*

Start the webserver (for REST API).

5.1.206 webserver-address

- String
- Default: 127.0.0.1
- YAML setting: *webservice.address*

IP address for the webserver to listen on.

5.1.207 webserver-allow-from

Changed in version 4.1.0: Default is now 127.0.0.1,::1, was 0.0.0.0/0,::/0 before.

- Comma separated list of IP addresses or subnets, negation supported
- Default: 127.0.0.1, ::1
- YAML setting: *webservice.allow_from*

These IPs and subnets are allowed to access the webserver. Note that specifying an IP address without a netmask uses an implicit netmask of /32 or /128.

5.1.208 webserver-hash-plaintext-credentials

New in version 4.6.0.

- Boolean
- Default: no
- YAML setting: *webservice.hash_plaintext_credentials*

Whether passwords and API keys supplied in the configuration as plaintext should be hashed during startup, to prevent the plaintext versions from staying in memory. Doing so increases significantly the cost of verifying credentials and is thus disabled by default. Note that this option only applies to credentials stored in the configuration as plaintext, but hashed credentials are supported without enabling this option.

5.1.209 webserver-loglevel

New in version 4.2.0.

- String
- Default: normal
- YAML setting: *webservice.loglevel*

One of `none`, `normal`, `detailed`. The amount of logging the webserver must do. 'none' means no useful webserver information will be logged. When set to 'normal', the webserver will log a line per request that should be familiar:

```
[webserver] e235780e-a5cf-415e-9326-9d33383e739e 127.0.0.1:55376 'GET /api/v1/
↪servers/localhost/bla HTTP/1.1' 404 196
```

When set to 'detailed', all information about the request and response are logged:

```
[webserver] e235780e-a5cf-415e-9326-9d33383e739e Request Details:
[webserver] e235780e-a5cf-415e-9326-9d33383e739e  Headers:
[webserver] e235780e-a5cf-415e-9326-9d33383e739e   accept: text/html,application/
↪xhtml+xml,application/xml;q=0.9,*/*;q=0.8
[webserver] e235780e-a5cf-415e-9326-9d33383e739e   accept-encoding: gzip, deflate
[webserver] e235780e-a5cf-415e-9326-9d33383e739e   accept-language: en-US,en;q=0.5
[webserver] e235780e-a5cf-415e-9326-9d33383e739e   connection: keep-alive
[webserver] e235780e-a5cf-415e-9326-9d33383e739e   dnt: 1
[webserver] e235780e-a5cf-415e-9326-9d33383e739e   host: 127.0.0.1:8081
[webserver] e235780e-a5cf-415e-9326-9d33383e739e   upgrade-insecure-requests: 1
[webserver] e235780e-a5cf-415e-9326-9d33383e739e   user-agent: Mozilla/5.0 (X11;␣
↪Linux x86_64; rv:64.0) Gecko/20100101 Firefox/64.0
[webserver] e235780e-a5cf-415e-9326-9d33383e739e  No body
[webserver] e235780e-a5cf-415e-9326-9d33383e739e Response details:
[webserver] e235780e-a5cf-415e-9326-9d33383e739e  Headers:
[webserver] e235780e-a5cf-415e-9326-9d33383e739e   Connection: close
[webserver] e235780e-a5cf-415e-9326-9d33383e739e   Content-Length: 49
[webserver] e235780e-a5cf-415e-9326-9d33383e739e   Content-Type: text/html;␣
↪charset=utf-8
[webserver] e235780e-a5cf-415e-9326-9d33383e739e   Server: PowerDNS/0.0.15896.0.
↪gaba8bab3ab
[webserver] e235780e-a5cf-415e-9326-9d33383e739e  Full body:
[webserver] e235780e-a5cf-415e-9326-9d33383e739e   <!html><title>Not Found</title>
↪<h1>Not Found</h1>
[webserver] e235780e-a5cf-415e-9326-9d33383e739e 127.0.0.1:55376 'GET /api/v1/
↪servers/localhost/bla HTTP/1.1' 404 196
```

The value between the hooks is a UUID that is generated for each request. This can be used to find all lines related to a single request.

Note: The webserver logs these line on the NOTICE level. The *loglevel* seting must be 5 or higher for these lines to end up in the log.

5.1.210 webserver-password

Changed in version 4.6.0: This setting now accepts a hashed and salted version.

- String
- Default: (empty)
- YAML setting: *webservice.password*

Password required to access the webserver. Since 4.6.0 the password can be hashed and salted using `rec_control hash-password` instead of being present in the configuration in plaintext, but the plaintext version is still supported.

5.1.211 webserver-port

- Integer
- Default: 8082
- YAML setting: *webservice.port*

TCP port where the webserver should listen on.

5.1.212 write-pid

- Boolean
- Default: yes
- YAML setting: *recursor.write_pid*

If a PID file should be written to *socket-dir*

5.1.213 x-dnssec-names

New in version 4.5.0.

- Comma separated list of strings
- Default: (empty)
- YAML setting: *dnssec.x_dnssec_names*

List of names whose DNSSEC validation metrics will be counted in a separate set of metrics that start with `x-dnssec-result-`. The names are suffix-matched. This can be used to not count known failing (test) name validations in the ordinary DNSSEC metrics.

5.1.214 system-resolver-ttl

New in version 5.1.0.

- Integer
- Default: 0
- YAML setting: *recursor.system_resolver_ttl*

Sets TTL in seconds of the system resolver feature. If not equal to zero names can be used for forwarding targets. The names will be resolved by the system resolver configured in the OS.

The TTL is used as a time to live to see if the names used in forwarding resolve to a different address than before. If the TTL is expired, a re-resolve will be done by the next iteration of the check function; if a change is detected, the recursor performs an equivalent of `rec_control reload-zones`.

Make sure the recursor itself is not used by the system resolver! Default is 0 (not enabled). A suggested value is 60.

5.1.215 system-resolver-interval

New in version 5.1.0.

- Integer
- Default: 0
- YAML setting: *recursor.system_resolver_interval*

Sets the check interval (in seconds) of the system resolver feature. All names known by the system resolver subsystem are periodically checked for changing values.

If the TTL of a name has expired, it is checked by re-resolving it. if a change is detected, the recursor performs an equivalent of `rec_control reload-zones`.

This settings sets the interval between the checks. If set to zero (the default), the value *system-resolver-ttl* is used.

5.1.216 system-resolver-self-resolve-check

New in version 5.1.0.

- Boolean
- Default: yes
- YAML setting: *recursor.system_resolver_self_resolve_check*

Warn on potential self-resolve. If this check draws the wrong conclusion, you can disable it.

CHAPTER

SIX

POWERDNS RECURSOR NEW STYLE (YAML) SETTINGS

Each setting can appear on the command line, prefixed by `--` and using the old style name, or in configuration files. Settings on the command line are processed after the file-based settings are processed.

Note: Starting with version 5.0.0, **Recursor** supports a new YAML syntax for configuration files as described here. If both `recursor.conf` and `recursor.yml` files are found in the configuration directory the YAML file is used. A configuration using the old style syntax can be converted to a YAML configuration using the instructions in *Conversion of old-style settings to YAML format*.

Release 5.0.0 will install a default old-style `recursor.conf` file.

Starting with version 5.1.0, in the absense of a `recursor.yml` file, an existing `recursor.conf` will be processed as YAML, if that fails, it will be processed as old-style configuration. Packages will stop installing a old-style `recursor.conf` file and start installing a default `recursor.conf` file containing YAML syntax.

With the release of 5.2.0, the default will be to expect a YAML configuation file and reading of old-style `recursor.conf` files will have to be enabled specifically by providing a command line option.

In a future release support for the "old-style" `recursor.conf` settings file will be dropped.

Note: Starting with version 5.1.0, the settings originally specified in a Lua config file can also be put in YAML form. The conversion printed by `rec_control show-yaml` will print these settings if a Lua config file is specified in the config file being converted. You have to choose however: either set Lua settings the old way in the Lua config file, or convert all to YAML. If you are using YAML settings of items originally specified in the Lua config file, do not set *recursor.lua_config_file* anymore. The **Recursor** will check that you do not mix both configuration methods.

6.1 YAML settings file

Please refer to e.g. https://docs.ansible.com/ansible/latest/reference_appendices/YAMLSyntax.html for a description of YAML syntax.

A **Recursor** configuration file has several sections. For example, `incoming` for settings related to receiving queries and `dnssec` for settings related to DNSSEC processing.

An example **Recursor** YAML configuration file looks like:

```
dnssec:
  log_bogus: true
incoming:
  listen:
    - 0.0.0.0:5301
    - '[::]:5301'
recursor:
  extended_resolution_errors: true
```

(continues on next page)

(continued from previous page)

```
  forward_zones:
    - zone: example.com
      forwarders:
        - 127.0.0.1:5301
outgoing:
  source_address:
    - 0.0.0.0
    - '::'
logging:
  loglevel: 6
```

Take care when listing IPv6 addresses, as characters used for these are special to YAML. If in doubt, quote any string containing `:`, `!`, `[` or `]` and use (online) tools to check your YAML syntax. Specify an empty sequence using `[]`.

The main setting file is called `recursor.yml` and will be processed first. This settings file might refer to other files via the *recursor.include_dir* setting. The next section will describe how settings specified in multiple files are merged.

6.2 Merging multiple setting files

If *recursor.include_dir* is set, all `.yml` files in it will be processed in alphabetical order, modifying the settings processed so far.

For simple values like an boolean or number setting, a value in the processed file will overwrite an existing setting.

For values of type sequence, the new value will *replace* the existing value if the existing value is equal to the `default` or if the new value is marked with the `!override` tag. Otherwise, the existing value will be *extended* with the new value by appending the new sequence to the existing.

For example, with the above example `recursor.yml` and an include directory containing a file `extra.yml`:

```
dnssec:
  log_bogus: false
recursor:
  forward_zones:
    - zone: example.net
      forwarders:
        - '::1'
outgoing:
   source_address: !override
     - 0.0.0.0
   dont_query: []
```

After merging, `dnssec.log_bogus` will be `false`, the sequence of `recursor.forward_zones` will contain 2 zones and the `outgoing` addresses used will contain one entry, as the `extra.yml` entry has overwritten the existing one.

`outgoing.dont-query` has a non-empty sequence as default value. The main `recursor.yml` did not set it, so before processing `extra.yml` it had the default value. After processing `extra.yml` the value will be set to the empty sequence, as existing default values are overwritten by new values.

Warning: The merging process does not process values deeper than the second level. For example if the main `recursor.yml` specified a forward zone

```
forward_zones:
  - zone: example.net
    forwarders:
      - '::1'
```

and another settings file contains

```
forward_zones:
  - zone: example.net
    forwarders:
      - '::2'
```

The result will *not* be a a single forward with two IP addresses, but two entries for `example.net`. It depends on the specific setting how the sequence is processed and interpreted further.

6.3 Description of YAML syntax for structured types

6.3.1 Socket Address

A socket address is a string containing either an IP address or and IP address:port combination For example:

```
some_key: 127.0.0.1
another_key: '[::1]:8080'
```

6.3.2 Subnet

A subnet is a single IP address or an IP address followed by a slash and a prefix length. If no prefix length is specified, `/32` or `/128` is assumed, indicating a single IP address. Subnets can also be prefixed with a `!`, specifying negation. This can be used to deny addresses from a previously allowed range.

For example, `alow-from` takes a sequence of subnets:

```
allow_from:
  - '2001:DB8::/32'
  - 128.66.0.0/16
  - '!128.66.1.2'
```

In this case the address `128.66.1.2` is excluded from the addresses allowed access.

6.3.3 Forward Zone

A forward zone is defined as:

```
zone: string
forwarders:
  - Socket Address
  - ...
recurse: Boolean, default false
allow_notify:  Boolean, default false
```

An example of a `forward_zones` entry, which consists of a sequence of *Forward Zone* entries:

```
- zone: example1.com
  forwarders:
    - 127.0.0.1
    - 127.0.0.1:5353
    - '[::1]:53'
- zone: example2.com
  forwarders:
    - '::1'
```

(continues on next page)

(continued from previous page)

```
  recurse: true
  notify_allowed: true
```

Starting with version 5.1.0, names can be used if *recursor.system_resolver_ttl* is set. The names will be resolved using the system resolver and an automatic refresh of the forwarding zones will happen if a name starts resolving to a new address. The refresh is done by performing the equivalent of `rec_control reload-zones`.

6.3.4 Auth Zone

An auth zone is defined as:

```
zone: string
file: string
```

An example of a `auth_zones` entry, consisting of a sequence of *Auth Zone*:

```
auth_zones:
  - zone: example.com
    file: zones/example.com.zone
  - zone: example.net
    file: zones/example.net.zone
```

6.4 Description of YAML syntax corresponding to Lua config items

The YAML settings below were introduced in version 5.1.0 and correspond to their respective Lua settings. Refer to *Advanced Configuration Using Lua*.

6.4.1 TrustAnchor

As of version 5.1.0, a trust anchor is defined as

```
name: string
dsrecords: sequence of DS record strings in presentation format
```

An example of a `trustanchors` entry, which is a sequence of *TrustAnchor*:

```
trustanchors:
  - name: example.com
    dsrecords:
    - 10000 8 2 a06d44b80b8f1d39a95c0b0d7c65d08458e880409bbc683457104237c7f8ec8d
```

6.4.2 NegativeTrustAnchor

As of version 5.1.0, a negative trust anchor is defined as

```
name: string
reason: string
```

An example of a `negative_trustanchors` entry, which is a sequence of *NegativeTrustAnchor*:

```
negative_trustanchors:
  - name: example.com
    reason: an example
```

6.4.3 ProtobufServer

As of version 5.1.0, a protobuf server is defined as

```
servers: [] Sequence of strings representing SocketAddress
timeout: 2
maxQueuedEntries: 100
reconnectWaitTime: 1
taggedOnly: false
asyncConnect: false
logQueries: true
logResponses: true
exportTypes: [A, AAAA, CNAME] Sequence of QType names
logMappedFrom: false
```

An example of a `protobuf_servers` entry, which is a sequence of *ProtobufServer*:

```
protobuf_servers:
  - servers: [127.0.0.1:4578]
    exportTypes: [A, AAAA]
  - servers: ['[2001:DB8::1]':7891]
    logQueries: false
    logResponses: true
    exportTypes: [A]
```

6.4.4 DNSTapFrameStreamServers

As of version 5.1.0, a dnstap framestream server is defined as

```
servers: [] Sequence of strings representing SocketAddress or a socket path
logQueries: true
logResponses: true
bufferHint: 0
flushTimeout: 0
inputQueueSize: 0
outputQueueSize: 0
queueNotifyThreshold: 0
reopenInterval: 0
```

An example of a `dnstap_framestream_servers` entry, which is a sequence of *DNSTapFrameStreamServers*:

```
dnstap_framestream_servers:
  - servers: [127.0.0.1:2024]
    logQueries: false
    logResponses: true
```

6.4.5 DNSTapNODFrameStreamServers

As of version 5.1.0, an NOD dnstap framestream server is defined as

```
servers: [] Sequence of strings representing SocketAddress or a socket path
logNODs: true
logUDRs: false
bufferHint: 0
flushTimeout: 0
inputQueueSize: 0
outputQueueSize: 0
```

(continues on next page)

(continued from previous page)

```
queueNotifyThreshold: 0
reopenInterval: 0
```

An example of a `dnstap_nod_framestream_servers` entry, which is a sequence of *DNSTapNODFrameStreamServers*:

```
dnstap_nop_framestream_servers:
  - servers: [127.0.0.1:2024]
    logNODs: false
    logUDRs: true
```

6.4.6 SortList

As of version 5.1.0, a sortlist entry is defined as

```
- key: Subnet
  subnets:
   - subnet: Subnet
     order: number
```

An example of a `sortlists` entry, which is a sequence of *SortList*:

```
sortlists:
  - key: 198.18.0.0/8
    subnets:
     - subnet: 233.252.0.0/24
       order: 10
  - key: 198.18.1.0/8
    subnets:
      - subnet: 198.18.0.0/16
        order: 20
      - subnet: 203.0.113.0/24
        order: 20
```

6.4.7 RPZ

As of version 5.1.0, an RPZ entry is defined as

```
name: name or pathname
addresses: [] Sequence of SocketAddress
defcontent: string
defpol:  Custom, Drop, NXDOMAIN, NODATA Truncate or NoAction
defpolOverrideLocalData: true
defttl: number
extendedErrorCode: number
extendedErrorExtra: string
includeSOA: false
ignoreDuplicates: false
maxTTL: number
policyName: string
tags: Sequence of string
overridesGettag: true
zoneSizeHint: number
tsig:
  name: string
  algo: string
  secret: base64string
refresh: number
```

(continues on next page)

(continued from previous page)

```
maxReceivedMBytes: number
localAddress: IP address
axfrTimeout: number
dumpFile: string
seedFile: string
```

If `addresses` is empty, the `name` field specifies the path name of the RPZ, otherwise the `name` field defines the name of the RPZ.

An example of an `rpzs` entry, which is a sequence of *RPZ*:

```
rpzs:
  - name: 'path/to/a/file'
  - name: 'remote.rpz'
    addresses: ['192.168.178.99']
    policyName: mypolicy
```

6.4.8 ZoneToCache

As of version 5.1.0, a ZoneToCache entry is defined as

```
zone: zonename
method: One of axfr, url, file
sources: [] Sequence of string, representing IP address, URL or path
timeout: 20
tsig:
  name: name of key
  algo: algorithm
  secret: Base64 endcoded secret
refreshPeriod: 86400
retryOnErrorPeriod: 60
maxReceivedMBytes: 0 Zero mean no restrcition
localAddress: local IP address to  bind to.
zonemd: One of ignore, validate, require
dnssec: One of ignore, validate, require
```

An example of an `zonetocaches` entry, which is a sequence of *ZoneToCache*:

```
zonetocaches:
  - zone: .
    method: url
    sources: ['https://www.example.com/path']
  - zone: example.com
    method: file
    sources: ['dir/example.com.zone']
```

6.4.9 AllowedAdditionalQType

As of version 5.1.0, an allowed addtional qtype entry is defined as:

```
qtype: string representing a QType
targets: [] Sequence of string representing QType
mode: One of Ignore, CacheOnly, CacheOnlyRequireAuth, ResolveImmediately,
↪ResolveDeferred, default CacheOnlyRequireAuth
```

An example of an `allowed_additional_qtypes` entry, which is a sequence of *AllowedAdditionalQType*:

```
allowed_additional_qtypes:
- qtype: MX
  targets: [A, AAAA]
- qtype: NAPTR
  targets: [A, AAAA, SRV]
  mode: ResolveDeferred
```

6.4.10 ProxyMapping

As of version 5.1.0, a proxy mapping entry is defined as:

```
subnet: Subnet
address: IPAddress
domains: [] Sequence of string
```

An example of an `proxymappings` entry, which is a sequence of *ProxyMapping*:

```
proxymappings:
  - subnet: 192.168.178.0/24
    address: 128.66.1.2
  - subnet: 192.168.179.0/24
    address: 128.66.1.3
    domains:
      - example.com
      - example.net
```

6.5 The YAML settings

The notation `section.name` means that an entry `name` can appear in the YAML section `section`. So the entry `recordcache.max_ttl` will end up in settings file as follows:

```
recordcache:
  ...
  max_ttl: 3600
  ...
```

6.5.1 `carbon.instance`

New in version 4.2.0.

- String
- Default: `recursor`
- Old style setting: *carbon-instance*

Change the instance or third string of the metric key. The default is recursor.

6.5.2 `carbon.interval`

- Integer
- Default: `30`
- Old style setting: *carbon-interval*

If sending carbon updates, this is the interval between them in seconds. See *Metrics and Statistics*.

6.5.3 carbon.ns

New in version 4.2.0.

- String
- Default: `pdns`
- Old style setting: *carbon-namespace*

Change the namespace or first string of the metric key. The default is pdns.

6.5.4 carbon.ourname

- String
- Default: (empty)
- Old style setting: *carbon-ourname*

If sending carbon updates, if set, this will override our hostname. Be careful not to include any dots in this setting, unless you know what you are doing. See *Sending metrics to Graphite/Metronome over Carbon*.

6.5.5 carbon.server

- Sequence of *Socket Address* (IP or IP:port combinations)
- Default: `[]`
- Old style setting: *carbon-server*

Will send all available metrics to these servers via the carbon protocol, which is used by graphite and metronome. See *Metrics and Statistics*.

6.5.6 dnssec.aggressive_cache_max_nsec3_hash_cost

New in version 5.0.2.

New in version 4.9.3.

New in version 4.8.6.

- Integer
- Default: `150`
- Old style setting: *aggressive-cache-max-nsec3-hash-cost*

Maximum estimated NSEC3 cost for a given query to consider aggressive use of the NSEC3 cache. The cost is estimated based on a heuristic taking the zone's NSEC3 salt and iterations parameters into account, as well at the number of labels of the requested name. For example a query for a name like a.b.c.d.e.f.example.com. in an example.com zone. secured with NSEC3 and 10 iterations (NSEC3 iterations count of 9) and an empty salt will have an estimated worst-case cost of 10 (iterations) * 6 (number of labels) = 60. The aggressive NSEC cache is an optimization to reduce the number of queries to authoritative servers, which is especially useful when a zone is under pseudo-random subdomain attack, and we want to skip it the zone parameters make it expensive.

6.5.7 dnssec.aggressive_cache_min_nsec3_hit_ratio

New in version 4.9.0.

- Integer
- Default: `2000`

- Old style setting: *aggressive-cache-min-nsec3-hit-ratio*

The limit for which to put NSEC3 records into the aggressive cache. A value of `n` means that an NSEC3 record is only put into the aggressive cache if the estimated probability of a random name hitting the NSEC3 record is higher than `1/n`. A higher `n` will cause more records to be put into the aggressive cache, e.g. a value of 4000 will cause records to be put in the aggressive cache even if the estimated probability of hitting them is twice as low as would be the case for `n=2000`. A value of 0 means no NSEC3 records will be put into the aggressive cache.

For large zones the effectiveness of the NSEC3 cache is reduced since each NSEC3 record only covers a randomly distributed subset of all possible names. This setting avoids doing unnecessary work for such large zones.

6.5.8 dnssec.aggressive_nsec_cache_size

New in version 4.5.0.

- Integer
- Default: `100000`
- Old style setting: *aggressive-nsec-cache-size*

The number of records to cache in the aggressive cache. If set to a value greater than 0, the recursor will cache NSEC and NSEC3 records to generate negative answers, as defined in **RFC 8198**. To use this, DNSSEC processing or validation must be enabled by setting *dnssec.validation* to `process`, `log-fail` or `validate`.

6.5.9 dnssec.disabled_algorithms

New in version 4.9.0.

- Sequence of strings
- Default: `[]`
- Old style setting: *dnssec-disabled-algorithms*

A list of DNSSEC algorithm numbers that should be considered disabled. These algorithms will not be used to validate DNSSEC signatures. Zones (only) signed with these algorithms will be considered `Insecure`.

If this setting is empty (the default), **`Recursor`** will determine which algorithms to disable automatically. This is done for specific algorithms only, currently algorithms 5 (`RSASHA1`) and 7 (`RSASHA1NSEC3SHA1`).

This is important on systems that have a default strict crypto policy, like RHEL9 derived systems. On such systems not disabling some algorithms (or changing the security policy) will make affected zones to be considered `Bogus` as using these algorithms fails.

6.5.10 dnssec.log_bogus

- Boolean
- Default: `false`
- Old style setting: *dnssec-log-bogus*

Log every DNSSEC validation failure. **Note**: This is not logged per-query but every time records are validated as Bogus.

6.5.11 dnssec.max_dnskeys

New in version 5.0.2.

New in version 4.9.3.

New in version 4.8.6.

- Integer
- Default: `2`
- Old style setting: *max-dnskeys*

Maximum number of DNSKEYs with the same algorithm and tag to consider when validating a given record. Setting this value to 1 effectively denies DNSKEY tag collisions in a zone.

6.5.12 dnssec.max_ds_per_zone

New in version 5.0.2.

New in version 4.9.3.

New in version 4.8.6.

- Integer
- Default: `8`
- Old style setting: *max-ds-per-zone*

Maximum number of DS records to consider when validating records inside a zone.

6.5.13 dnssec.max_nsec3_hash_computations_per_query

New in version 5.0.2.

New in version 4.9.3.

New in version 4.8.6.

- Integer
- Default: `600`
- Old style setting: *max-nsec3-hash-computations-per-query*

Maximum number of NSEC3 hashes that we are willing to compute during DNSSEC validation, per incoming query.

6.5.14 dnssec.max_nsec3s_per_record

New in version 5.0.2.

New in version 4.9.3.

New in version 4.8.6.

- Integer
- Default: `10`
- Old style setting: *max-nsec3s-per-record*

Maximum number of NSEC3s to consider when validating a given denial of existence.

6.5.15 dnssec.max_rrsigs_per_record

New in version 5.0.2.

New in version 4.9.3.

New in version 4.8.6.

- Integer

- Default: `2`
- Old style setting: *max-rrsigs-per-record*

Maximum number of RRSIGs we are willing to cryptographically check when validating a given record. Expired or not yet incepted RRSIGs do not count toward to this limit.

6.5.16 dnssec.max_signature_validations_per_query

New in version 5.0.2.

New in version 4.9.3.

New in version 4.8.6.

- Integer
- Default: `30`
- Old style setting: *max-signature-validations-per-query*

Maximum number of RRSIG signatures we are willing to validate per incoming query.

6.5.17 dnssec.negative_trustanchors

New in version 5.1.0.

- Sequence of *NegativeTrustAnchor*
- Default: `[]`
- Equivalent Lua config in *Managing DNSSEC Trust Anchors in the Lua Configuration*

Sequence of negative trust anchors.

6.5.18 dnssec.nsec3_max_iterations

New in version 4.1.0.

Changed in version 4.5.2: Default is now 150, was 2500 before.

Changed in version 5.0.0: Default is now 50, was 150 before.

- Integer
- Default: `50`
- Old style setting: *nsec3-max-iterations*

Maximum number of iterations allowed for an NSEC3 record. If an answer containing an NSEC3 record with more iterations is received, its DNSSEC validation status is treated as `Insecure`.

6.5.19 dnssec.signature_inception_skew

New in version 4.1.5.

Changed in version 4.2.0: Default is now 60, was 0 before.

- Integer
- Default: `60`
- Old style setting: *signature-inception-skew*

Allow the signature inception to be off by this number of seconds. Negative values are not allowed.

6.5.20 dnssec.trustanchorfile

New in version 5.1.0.

- String
- Default: (empty)
- Equivalent Lua config in *Managing DNSSEC Trust Anchors in the Lua Configuration*

A path to a zone file to read trust anchors from. This can be used to read distribution provided trust anchors, as for instance `/usr/share/dns/root.key` from Debian's `dns-root-data` package.

6.5.21 dnssec.trustanchorfile_interval

New in version 5.1.0.

- Integer
- Default: `24`
- Equivalent Lua config in *Managing DNSSEC Trust Anchors in the Lua Configuration*

Interval (in hours) to re-read the `trustanchorfile`. Zero disables periodic re-reads.

6.5.22 dnssec.trustanchors

New in version 5.1.0.

- Sequence of *TrustAnchor*
- Default:

```
- name: .
  dsrecords:
  - 20326 8 2 e06d44b80b8f1d39a95c0b0d7c65d08458e880409bbc683457104237c7f8ec8d
```

- Equivalent Lua config in *Managing DNSSEC Trust Anchors in the Lua Configuration*

Sequence of trust anchors. If the sequence contains an entry for the root zone, the default root zone trust anchor is not included. If a zone appears multiple times, the entries in `dsrecords` are merged.

6.5.23 dnssec.validation

New in version 4.0.0.

Changed in version 4.5.0: The default changed from `process-no-validate` to `process`

- String
- Default: `process`
- Old style setting: *dnssec*

One of `off`, `process-no-validate`, `process`, `log-fail`, `validate`

Set the mode for DNSSEC processing, as detailed in *DNSSEC in the PowerDNS Recursor*.

`off` No DNSSEC processing whatsoever. Ignore DO-bits in queries, don't request any DNSSEC information from authoritative servers. This behaviour is similar to PowerDNS Recursor pre-4.0.

`process-no-validate` Respond with DNSSEC records to clients that ask for it, set the DO bit on all outgoing queries. Don't do any validation.

`process` Respond with DNSSEC records to clients that ask for it, set the DO bit on all outgoing queries. Do validation for clients that request it (by means of the AD- bit or DO-bit in the query).

log-fail Similar behaviour to `process`, but validate RRSIGs on responses and log bogus responses.

validate Full blown DNSSEC validation. Send SERVFAIL to clients on bogus responses.

6.5.24 dnssec.x_dnssec_names

New in version 4.5.0.

- Sequence of strings
- Default: `[]`
- Old style setting: *x-dnssec-names*

List of names whose DNSSEC validation metrics will be counted in a separate set of metrics that start with `x-dnssec-result-`. The names are suffix-matched. This can be used to not count known failing (test) name validations in the ordinary DNSSEC metrics.

6.5.25 ecs.add_for

New in version 4.2.0.

- Sequence of *Subnet* (IP addresses or subnets, negation supported)
- Default: `[0.0.0.0/0, '::/0', '!127.0.0.0/8', '!10.0.0.0/8', '!100.64.0.0/10', '!169.254.0.0/16', '!192.168.0.0/16', '!172.16.0.0/12', '!::1/128', '!fc00::/7', '!fe80::/10']`
- Old style setting: *ecs-add-for*

List of requestor netmasks for which the requestor IP Address should be used as the **EDNS Client Subnet** for outgoing queries. Outgoing queries for requestors that do not match this list will use the *ecs.scope_zero_address* instead. Valid incoming ECS values from *incoming.use_incoming_edns_subnet* are not replaced.

Regardless of the value of this setting, ECS values are only sent for outgoing queries matching the conditions in the *outgoing.edns_subnet_allow_list* setting. This setting only controls the actual value being sent.

This defaults to not using the requestor address inside RFC1918 and similar 'private' IP address spaces.

6.5.26 ecs.cache_limit_ttl

New in version 4.1.12.

- Integer
- Default: `0`
- Old style setting: *ecs-cache-limit-ttl*

The minimum TTL for an ECS-specific answer to be inserted into the record cache. This condition applies in conjunction with `ecs-ipv4-cache-bits` or `ecs-ipv6-cache-bits`. That is, only if both the limits apply, the record will not be cached. This decision can be overridden by `ecs-ipv4-never-cache` and `ecs-ipv6-never-cache`.

6.5.27 ecs.ipv4_bits

New in version 4.1.0.

- Integer
- Default: `24`
- Old style setting: *ecs-ipv4-bits*

Number of bits of client IPv4 address to pass when sending EDNS Client Subnet address information.

6.5.28 `ecs.ipv4_cache_bits`

New in version 4.1.12.

- Integer
- Default: `24`
- Old style setting: *ecs-ipv4-cache-bits*

Maximum number of bits of client IPv4 address used by the authoritative server (as indicated by the EDNS Client Subnet scope in the answer) for an answer to be inserted into the record cache. This condition applies in conjunction with `ecs-cache-limit-ttl`. That is, only if both the limits apply, the record will not be cached. This decision can be overridden by `ecs-ipv4-never-cache` and `ecs-ipv6-never-cache`.

6.5.29 `ecs.ipv4_never_cache`

New in version 4.5.0.

- Boolean
- Default: `false`
- Old style setting: *ecs-ipv4-never-cache*

When set, never cache replies carrying EDNS IPv4 Client Subnet scope in the record cache. In this case the decision made by `` `ecs-ipv4-cache-bits `` and `ecs-cache-limit-ttl` is no longer relevant.

6.5.30 `ecs.ipv6_bits`

New in version 4.1.0.

- Integer
- Default: `56`
- Old style setting: *ecs-ipv6-bits*

Number of bits of client IPv6 address to pass when sending EDNS Client Subnet address information.

6.5.31 `ecs.ipv6_cache_bits`

New in version 4.1.12.

- Integer
- Default: `56`
- Old style setting: *ecs-ipv6-cache-bits*

Maximum number of bits of client IPv6 address used by the authoritative server (as indicated by the EDNS Client Subnet scope in the answer) for an answer to be inserted into the record cache. This condition applies in conjunction with `ecs-cache-limit-ttl`. That is, only if both the limits apply, the record will not be cached. This decision can be overridden by `ecs-ipv4-never-cache` and `ecs-ipv6-never-cache`.

6.5.32 ecs.ipv6_never_cache

New in version 4.5.0.

- Boolean
- Default: `false`
- Old style setting: *ecs-ipv6-never-cache*

When set, never cache replies carrying EDNS IPv6 Client Subnet scope in the record cache. In this case the decision made by `ecs-ipv6-cache-bits and `ecs-cache-limit-ttl` is no longer relevant.

6.5.33 ecs.minimum_ttl_override

Changed in version 4.5.0: Old versions used default 0.

- Integer
- Default: `1`
- Old style setting: *ecs-minimum-ttl-override*

This setting artificially raises the TTLs of records in the ANSWER section of ECS-specific answers to be at least this long. Setting this to a value greater than 1 technically is an RFC violation, but might improve performance a lot. Using a value of 0 impacts performance of TTL 0 records greatly, since it forces the recursor to contact authoritative servers every time a client requests them. Can be set at runtime using `rec_control set-ecs-minimum-ttl 3600`.

6.5.34 ecs.scope_zero_address

New in version 4.1.0.

- String
- Default: (empty)
- Old style setting: *ecs-scope-zero-address*

The IP address sent via EDNS Client Subnet to authoritative servers listed in *outgoing.edns_subnet_allow_list* when *incoming.use_incoming_edns_subnet* is set and the query has an ECS source prefix-length set to 0. The default is to look for the first usable (not an `any` one) address in *outgoing.source_address* (starting with IPv4). If no suitable address is found, the recursor fallbacks to sending 127.0.0.1.

6.5.35 incoming.allow_from

- Sequence of *Subnet* (IP addresses or subnets, negation supported)
- Default: `[127.0.0.0/8, 10.0.0.0/8, 100.64.0.0/10, 169.254.0.0/16, 192.168.0.0/16, 172.16.0.0/12, '::1/128', 'fc00::/7', 'fe80::/10']`
- Old style setting: *allow-from*

Netmasks (both IPv4 and IPv6) that are allowed to use the server. The default allows access only from **RFC 1918** private IP addresses. An empty value means no checking is done, all clients are allowed. Due to the aggressive nature of the internet these days, it is highly recommended to not open up the recursor for the entire internet. Questions from IP addresses not listed here are ignored and do not get an answer.

When the Proxy Protocol is enabled (see *incoming.proxy_protocol_from*), the recursor will check the address of the client IP advertised in the Proxy Protocol header instead of the one of the proxy.

Note that specifying an IP address without a netmask uses an implicit netmask of /32 or /128.

6.5.36 incoming.allow_from_file

- String
- Default: (empty)
- Old style setting: *allow-from-file*

Like *incoming.allow_from*, except reading a sequence of *Subnet* from file. Overrides the *incoming.allow_from* setting. Example content of th specified file:

```
- 127.0.01
- ::1
```

6.5.37 incoming.allow_no_rd

New in version 5.0.0.

- Boolean
- Default: `false`
- Old style setting: *allow-no-rd*

Allow `no recursion desired (RD=0) queries` to query cache contents. If not set (the default), these queries are answered with rcode `Refused`.

6.5.38 incoming.allow_notify_for

New in version 4.6.0.

- Sequence of strings
- Default: `[]`
- Old style setting: *allow-notify-for*

Domain names specified in this list are used to permit incoming NOTIFY operations to wipe any cache entries that match the domain name. If this list is empty, all NOTIFY operations will be ignored.

6.5.39 incoming.allow_notify_for_file

New in version 4.6.0.

- String
- Default: (empty)
- Old style setting: *allow-notify-for-file*

Like *incoming.allow_notify_for*, except reading a sequence of names from file. Example contents of specified file:

```
- example.com
- example.org
```

6.5.40 incoming.allow_notify_from

New in version 4.6.0.

- Sequence of *Subnet* (IP addresses or subnets, negation supported)
- Default: `[]`

- Old style setting: *allow-notify-from*

Subnets (both IPv4 and IPv6) that are allowed to issue NOTIFY operations to the server. NOTIFY operations from IP addresses not listed here are ignored and do not get an answer.

When the Proxy Protocol is enabled (see *incoming.proxy_protocol_from*), the recursor will check the address of the client IP advertised in the Proxy Protocol header instead of the one of the proxy.

Note that specifying an IP address without a netmask uses an implicit netmask of /32 or /128.

NOTIFY operations received from a client listed in one of these netmasks will be accepted and used to initiate a freshness check for an RPZ zone or wipe any cache entries whose zones match the zone specified in the NOTIFY operation, but only if that zone (or one of its parents) is included in *incoming.allow_notify_for*, *incoming.allow_notify_for_file*, or *recursor.forward_zones_file* with a `allow_notify` set to `true`.

6.5.41 `incoming.allow_notify_from_file`

New in version 4.6.0.

- String
- Default: (empty)
- Old style setting: *allow-notify-from-file*

Like *incoming.allow_notify_from*, except reading a sequence of *Subnet* from file.

6.5.42 `incoming.distribution_load_factor`

New in version 4.1.12.

- Double
- Default: `0.0`
- Old style setting: *distribution-load-factor*

If *incoming.pdns_distributes_queries* is set and this setting is set to another value than 0, the distributor thread will use a bounded load-balancing algorithm while distributing queries to worker threads, making sure that no thread is assigned more queries than distribution-load-factor times the average number of queries currently processed by all the workers. For example, with a value of 1.25, no server should get more than 125 % of the average load. This helps making sure that all the workers have roughly the same share of queries, even if the incoming traffic is very skewed, with a larger number of requests asking for the same qname.

6.5.43 `incoming.distribution_pipe_buffer_size`

New in version 4.2.0.

- Integer
- Default: `0`
- Old style setting: *distribution-pipe-buffer-size*

Size in bytes of the internal buffer of the pipe used by the distributor to pass incoming queries to a worker thread. Requires support for *F_SETPIPE_SZ* which is present in Linux since 2.6.35. The actual size might be rounded up to a multiple of a page size. 0 means that the OS default size is used. A large buffer might allow the recursor to deal with very short-lived load spikes during which a worker thread gets overloaded, but it will be at the cost of an increased latency.

6.5.44 incoming.distributor_threads

New in version 4.2.0.

- Integer
- Default: 1 if *pdns-distributes-queries* is set, 0 otherwise
- Old style setting: *distributor-threads*

If *incoming.pdns_distributes_queries* is set, spawn this number of distributor threads on startup. Distributor threads handle incoming queries and distribute them to other threads based on a hash of the query.

6.5.45 incoming.edns_padding_from

New in version 4.5.0.

Changed in version 5.0.5: YAML settings only: previously this was defined as a string instead of a sequence

- Sequence of *Subnet* (IP addresses or subnets, negation supported)
- Default: `[]`
- Old style setting: *edns-padding-from*

List of netmasks (proxy IP in case of proxy-protocol presence, client IP otherwise) for which EDNS padding will be enabled in responses, provided that *incoming.edns_padding_mode* applies.

6.5.46 incoming.edns_padding_mode

New in version 4.5.0.

- String
- Default: `padded-queries-only`
- Old style setting: *edns-padding-mode*

One of `always`, `padded-queries-only`. Whether to add EDNS padding to all responses (`always`) or only to responses for queries containing the EDNS padding option (`padded-queries-only`, the default). In both modes, padding will only be added to responses for queries coming from *incoming.edns_padding_from* sources.

6.5.47 incoming.edns_padding_tag

New in version 4.5.0.

- Integer
- Default: `7830`
- Old style setting: *edns-padding-tag*

The packetcache tag to use for padded responses, to prevent a client not allowed by the :ref::*setting-edns-padding-from* list to be served a cached answer generated for an allowed one. This effectively divides the packet cache in two when *incoming.edns_padding_from* is used. Note that this will not override a tag set from one of the `Lua` hooks.

6.5.48 incoming.gettag_needs_edns_options

New in version 4.1.0.

- Boolean
- Default: `false`

- Old style setting: *gettag-needs-edns-options*

If set, EDNS options in incoming queries are extracted and passed to the `gettag()` hook in the `ednsoptions` table.

6.5.49 incoming.listen

- Sequence of *Socket Address* (IP or IP:port combinations)
- Default: `[127.0.0.1]`
- Old style setting: *local-address*

Local IP addresses to which we bind. Each address specified can include a port number; if no port is included then the *incoming.port* port will be used for that address. If a port number is specified, it must be separated from the address with a ':'; for an IPv6 address the address must be enclosed in square brackets.

Example:

```
incoming:
  listen:
    - 127.0.0.1
    - listen: '[::1]:5353'
    - listen: '::'
```

6.5.50 incoming.max_concurrent_requests_per_tcp_connection

New in version 4.3.0.

- Integer
- Default: `10`
- Old style setting: *max-concurrent-requests-per-tcp-connection*

Maximum number of incoming requests handled concurrently per tcp connection. This number must be larger than 0 and smaller than 65536 and also smaller than *max-mthreads*.

6.5.51 incoming.max_tcp_clients

- Integer
- Default: `128`
- Old style setting: *max-tcp-clients*

Maximum number of simultaneous incoming TCP connections allowed.

6.5.52 incoming.max_tcp_per_client

- Integer
- Default: `0`
- Old style setting: *max-tcp-per-client*

Maximum number of simultaneous incoming TCP connections allowed per client (remote IP address). 0 means unlimited.

6.5.53 incoming.max_tcp_queries_per_connection

New in version 4.1.0.

- Integer
- Default: `0`
- Old style setting: *max-tcp-queries-per-connection*

Maximum number of DNS queries in a TCP connection. 0 means unlimited.

6.5.54 incoming.max_udp_queries_per_round

New in version 4.1.4.

- Integer
- Default: `10000`
- Old style setting: *max-udp-queries-per-round*

Under heavy load the recursor might be busy processing incoming UDP queries for a long while before there is no more of these, and might therefore neglect scheduling new `mthreads`, handling responses from authoritative servers or responding to *rec_control* requests. This setting caps the maximum number of incoming UDP DNS queries processed in a single round of looping on `recvmsg()` after being woken up by the multiplexer, before returning back to normal processing and handling other events.

6.5.55 incoming.non_local_bind

- Boolean
- Default: `false`
- Old style setting: *non-local-bind*

Bind to addresses even if one or more of the *incoming.listen*'s do not exist on this server. Setting this option will enable the needed socket options to allow binding to non-local addresses. This feature is intended to facilitate ip-failover setups, but it may also mask configuration issues and for this reason it is disabled by default.

6.5.56 incoming.pdns_distributes_queries

Changed in version 4.9.0: Default changed to `no`, previously it was `yes`.

- Boolean
- Default: `false`
- Old style setting: *pdns-distributes-queries*

If set, PowerDNS will use distinct threads to listen to client sockets and distribute that work to worker-threads using a hash of the query. This feature should maximize the cache hit ratio on versions before 4.9.0. To use more than one thread set *incoming.distributor_threads* in version 4.2.0 or newer. Enabling should improve performance on systems where *incoming.reuseport* does not have the effect of balancing the queries evenly over multiple worker threads.

6.5.57 incoming.port

- Integer
- Default: `53`
- Old style setting: *local-port*

Local port to bind to. If an address in *incoming.listen* does not have an explicit port, this port is used.

6.5.58 incoming.proxy_protocol_exceptions

New in version 5.1.0.

- Sequence of *Socket Address* (IP or IP:port combinations)
- Default: `[]`
- Old style setting: *proxy-protocol-exceptions*

If set, clients sending from an address in *incoming.proxy_protocol_from* to a address:port listed here are excluded from using the Proxy Protocol. If no port is specified, port 53 is assumed. This is typically used to provide an easy to use address and port to send debug queries to.

6.5.59 incoming.proxy_protocol_from

New in version 4.4.0.

Changed in version 5.0.5: YAML settings only: previously this was defined as a string instead of a sequence

- Sequence of *Subnet* (IP addresses or subnets, negation supported)
- Default: `[]`
- Old style setting: *proxy-protocol-from*

Ranges that are required to send a Proxy Protocol version 2 header in front of UDP and TCP queries, to pass the original source and destination addresses and ports to the recursor, as well as custom values. Queries that are not prefixed with such a header will not be accepted from clients in these ranges. Queries prefixed by headers from clients that are not listed in these ranges will be dropped.

Note that once a Proxy Protocol header has been received, the source address from the proxy header instead of the address of the proxy will be checked against the *incoming.allow_from* ACL.

The dnsdist docs have more information about the PROXY protocol.

6.5.60 incoming.proxy_protocol_maximum_size

New in version 4.4.0.

- Integer
- Default: `512`
- Old style setting: *proxy-protocol-maximum-size*

The maximum size, in bytes, of a Proxy Protocol payload (header, addresses and ports, and TLV values). Queries with a larger payload will be dropped.

6.5.61 incoming.proxymappings

New in version 5.1.0.

- Sequence of *ProxyMapping*
- Default: `[]`
- Equivalent Lua config in *Table Based Proxy Mapping*

Sequence of ProxyMapping

6.5.62 incoming.reuseport

Changed in version 4.9.0: The default is changed to `yes`, previously it was `no`. If `SO_REUSEPORT` support is not available, the setting defaults to `no`.

- Boolean
- Default: `true`
- Old style setting: *reuseport*

If `SO_REUSEPORT` support is available, allows multiple threads and processes to open listening sockets for the same port.

Since 4.1.0, when *incoming.pdns_distributes_queries* is disabled and *incoming.reuseport* is enabled, every worker-thread will open a separate listening socket to let the kernel distribute the incoming queries instead of running a distributor thread (which could otherwise be a bottleneck) and avoiding thundering herd issues, thus leading to much higher performance on multi-core boxes.

6.5.63 incoming.tcp_fast_open

New in version 4.1.0.

- Integer
- Default: `0`
- Old style setting: *tcp-fast-open*

Enable TCP Fast Open support, if available, on the listening sockets. The numerical value supplied is used as the queue size, 0 meaning disabled. See *TCP Fast Open Support*.

6.5.64 incoming.tcp_timeout

- Integer
- Default: `2`
- Old style setting: *client-tcp-timeout*

Time to wait for data from TCP clients.

6.5.65 incoming.udp_truncation_threshold

Changed in version 4.2.0: Before 4.2.0, the default was 1680.

- Integer
- Default: `1232`
- Old style setting: *udp-truncation-threshold*

EDNS0 allows for large UDP response datagrams, which can potentially raise performance. Large responses however also have downsides in terms of reflection attacks. This setting limits the accepted size. Maximum value is 65535, but values above 4096 should probably not be attempted.

To know why 1232, see the note at *outgoing.edns_bufsize*.

6.5.66 incoming.use_incoming_edns_subnet

- Boolean
- Default: `false`

- Old style setting: *use-incoming-edns-subnet*

Whether to process and pass along a received EDNS Client Subnet to authoritative servers. The ECS information will only be sent for netmasks and domains listed in *outgoing.edns_subnet_allow_list* and will be truncated if the received scope exceeds *ecs.ipv4_bits* for IPv4 or *ecs.ipv6_bits* for IPv6.

6.5.67 `logging.common_errors`

- Boolean
- Default: `false`
- Old style setting: *log-common-errors*

Some DNS errors occur rather frequently and are no cause for alarm.

6.5.68 `logging.disable_syslog`

- Boolean
- Default: `false`
- Old style setting: *disable-syslog*

Do not log to syslog, only to stderr. Use this setting when running inside a supervisor that handles logging (like systemd). **Note**: do not use this setting in combination with *recursor.daemon* as all logging will disappear.

6.5.69 `logging.dnstap_framestream_servers`

New in version 5.1.0.

- Sequence of *DNSTapFrameStreamServers*
- Default: `[]`
- Equivalent Lua config in *Logging DNS messages with Protocol Buffers*

Sequence of dnstap servers. Currently the maximum size of this list is one.

6.5.70 `logging.dnstap_nod_framestream_servers`

New in version 5.1.0.

- Sequence of *DNSTapNODFrameStreamServers*
- Default: `[]`
- Equivalent Lua config in *Logging DNS messages with Protocol Buffers*

Sequence of NOD dnstap servers. Currently the maximum size of this list is one.

6.5.71 `logging.facility`

- String
- Default: (empty)
- Old style setting: *logging-facility*

If set to a digit, logging is performed under this LOCAL facility. See *Logging*. Do not pass names like 'local0'!

6.5.72 logging.loglevel

Changed in version 5.0.0: Previous version would not allow setting a level below `3 (error)`.

- Integer
- Default: `6`
- Old style setting: *loglevel*

Amount of logging. The higher the number, the more lines logged. Corresponds to `syslog` level values (e.g. 0 = `emergency`, 1 = `alert`, 2 = `critical`, 3 = `error`, 4 = `warning`, 5 = `notice`, 6 = `info`, 7 = `debug`). Each level includes itself plus the lower levels before it. Not recommended to set this below 3. If *logging.quiet* is `no/false`, *logging.loglevel* will be minimally set to `6 (info)`.

6.5.73 logging.outgoing_protobuf_servers

New in version 5.1.0.

- Sequence of *ProtobufServer*
- Default: `[]`
- Equivalent Lua config in *Logging DNS messages with Protocol Buffers*

Sequence of outgoing protobuf servers. Currently the maximum size of this list is one.

6.5.74 logging.protobuf_mask_v4

New in version 5.1.0.

- Integer
- Default: `32`
- Equivalent Lua config in *Logging DNS messages with Protocol Buffers*

Network mask to apply to the client IPv4 addresses, for anonymization purposes. The default of 32 means no anonymization.

6.5.75 logging.protobuf_mask_v6

New in version 5.1.0.

- Integer
- Default: `128`
- Equivalent Lua config in *Logging DNS messages with Protocol Buffers*

Network mask to apply to the client IPv6 addresses, for anonymization purposes. The default of 128 means no anonymization.

6.5.76 logging.protobuf_servers

New in version 5.1.0.

- Sequence of *ProtobufServer*
- Default: `[]`
- Equivalent Lua config in *Logging DNS messages with Protocol Buffers*

Sequence of outgoing protobuf servers. Currently the maximum size of this list is one.

6.5.77 `logging.protobuf_use_kernel_timestamp`

New in version 4.2.0.

- Boolean
- Default: `false`
- Old style setting: *protobuf-use-kernel-timestamp*

Whether to compute the latency of responses in protobuf messages using the timestamp set by the kernel when the query packet was received (when available), instead of computing it based on the moment we start processing the query.

6.5.78 `logging.quiet`

- Boolean
- Default: `true`
- Old style setting: *quiet*

Don't log queries.

6.5.79 `logging.rpz_changes`

New in version 4.1.0.

- Boolean
- Default: `false`
- Old style setting: *log-rpz-changes*

Log additions and removals to RPZ zones at Info (6) level instead of Debug (7).

6.5.80 `logging.statistics_interval`

New in version 4.1.0.

- Integer
- Default: `1800`
- Old style setting: *statistics-interval*

Interval between logging statistical summary on recursor performance. Use 0 to disable.

6.5.81 `logging.structured_logging`

New in version 4.6.0.

Changed in version 5.1.0: Disabling structured logging is not supported

- Boolean
- Default: `true`
- Old style setting: *structured-logging*

Prefer structured logging when both an old style and a structured log messages is available.

6.5.82 logging.structured_logging_backend

New in version 4.8.0.

Changed in version 5.1.0: The JSON backend was added

- String
- Default: `default`
- Old style setting: *structured-logging-backend*

The backend used for structured logging output. This setting must be set on the command line (`--structured-logging-backend=...`) to be effective. Available backends are:

- `default`: use the traditional logging system to output structured logging information.
- `systemd-journal`: use systemd-journal. When using this backend, provide `-o verbose` or simular output option to `journalctl` to view the full information.
- `json`: JSON objects are written to the standard error stream.

See *Structured Logging Dictionary* for more details.

6.5.83 logging.timestamp

- Boolean
- Default: `true`
- Old style setting: *log-timestamp*

6.5.84 logging.trace

- String
- Default: `no`
- Old style setting: *trace*

One of `no`, `yes` or `fail`. If turned on, output impressive heaps of logging. May destroy performance under load. To log only queries resulting in a `ServFail` answer from the resolving process, this value can be set to `fail`, but note that the performance impact is still large. Also note that queries that do produce a result but with a failing DNSSEC validation are not written to the log

6.5.85 nod.db_size

New in version 4.2.0.

- Integer
- Default: `67108864`
- Old style setting: *new-domain-db-size*

The default size of the stable bloom filter used to store previously observed domains is 67108864. To change the number of cells, use this setting. For each cell, the SBF uses 1 bit of memory, and one byte of disk for the persistent file. If there are already persistent files saved to disk, this setting will have no effect unless you remove the existing files.

6.5.86 nod.db_snapshot_interval

New in version 5.1.0.

- Integer
- Default: `600`
- Old style setting: *new-domain-db-snapshot-interval*

Interval (in seconds) to write the NOD and UDR DB snapshots. Set to zero to disable snapshot writing.',

6.5.87 nod.history_dir

New in version 4.2.0.

- String
- Default: Determined by distribution
- Old style setting: *new-domain-history-dir*

This setting controls which directory is used to store the on-disk cache of previously observed domains.

The default depends on `LOCALSTATEDIR` when building the software. Usually this comes down to `/var/lib/pdns-recursor/nod` or `/usr/local/var/lib/pdns-recursor/nod`).

The newly observed domain feature uses a stable bloom filter to store a history of previously observed domains. The data structure is synchronized to disk every 10 minutes, and is also initialized from disk on startup. This ensures that previously observed domains are preserved across recursor restarts. If you change the new-domain-db-size setting, you must remove any files from this directory.

6.5.88 nod.ignore_list

New in version 4.5.0.

- Sequence of strings
- Default: `[]`
- Old style setting: *new-domain-ignore-list*

This setting is a list of all domains (and implicitly all subdomains) that will never be considered a new domain. For example, if the domain 'example.com' is in the list, then 'foo.bar.example.com' will never be considered a new domain. One use-case for the ignore list is to never reveal details of internal subdomains via the new-domain-lookup feature.

6.5.89 nod.ignore_list_file

New in version 5.1.0.

- String
- Default: (empty)
- Old style setting: *new-domain-ignore-list-file*

Path to a file with a list of domains. File should have one domain per line, with no extra characters or comments. See *nod.ignore_list*.

6.5.90 nod.log

New in version 4.2.0.

- Boolean
- Default: `true`
- Old style setting: *new-domain-log*

If a newly observed domain is detected, log that domain in the recursor log file. The log line looks something like:

```
Jul 18 11:31:25 Newly observed domain nod=sdfoijdfio.com
```

6.5.91 nod.lookup

New in version 4.2.0.

- String
- Default: (empty)
- Old style setting: *new-domain-lookup*

If a domain is specified, then each time a newly observed domain is detected, the recursor will perform an A record lookup of '<newly observed domain>.<lookup domain>'. For example if 'new-domain-lookup' is configured as 'nod.powerdns.com', and a new domain 'example.com' is detected, then an A record lookup will be made for 'example.com.nod.powerdns.com'. This feature gives a way to share the newly observed domain with partners, vendors or security teams. The result of the DNS lookup will be ignored by the recursor.

6.5.92 nod.pb_tag

New in version 4.2.0.

- String
- Default: `pdns-nod`
- Old style setting: *new-domain-pb-tag*

If protobuf is configured, then this tag will be added to all protobuf response messages when a new domain is observed.

6.5.93 nod.tracking

New in version 4.2.0.

- Boolean
- Default: `false`
- Old style setting: *new-domain-tracking*

Whether to track newly observed domains, i.e. never seen before. This is a probabilistic algorithm, using a stable bloom filter to store records of previously seen domains. When enabled for the first time, all domains will appear to be newly observed, so the feature is best left enabled for e.g. a week or longer before using the results. Note that this feature is optional and must be enabled at compile-time, thus it may not be available in all pre-built packages. If protobuf is enabled and configured, then the newly observed domain status will appear as a flag in Response messages.

6.5.94 nod.unique_response_db_size

New in version 4.2.0.

- Integer
- Default: `67108864`
- Old style setting: *unique-response-db-size*

The default size of the stable bloom filter used to store previously observed responses is 67108864. To change the number of cells, use this setting. For each cell, the SBF uses 1 bit of memory, and one byte of disk for the persistent file. If there are already persistent files saved to disk, this setting will have no effect unless you remove the existing files.

6.5.95 nod.unique_response_history_dir

New in version 4.2.0.

- String
- Default: Determined by distribution
- Old style setting: *unique-response-history-dir*

This setting controls which directory is used to store the on-disk cache of previously observed responses.

The default depends on `LOCALSTATEDIR` when building the software. Usually this comes down to `/var/lib/pdns-recursor/udr` or `/usr/local/var/lib/pdns-recursor/udr`).

The newly observed domain feature uses a stable bloom filter to store a history of previously observed responses. The data structure is synchronized to disk every 10 minutes, and is also initialized from disk on startup. This ensures that previously observed responses are preserved across recursor restarts. If you change the unique-response-db-size, you must remove any files from this directory.

6.5.96 nod.unique_response_ignore_list

New in version 5.1.0.

- Sequence of strings
- Default: `[]`
- Old style setting: *unique-response-ignore-list*

This setting is a list of all domains (and implicitly all subdomains) that will never be considered for new unique domain responses. For example, if the domain 'example.com' is in the list, then 'foo.bar.example.com' will never be considered for a new unique domain response.

6.5.97 nod.unique_response_ignore_list_file

New in version 5.1.0.

- String
- Default: (empty)
- Old style setting: *unique-response-ignore-list-file*

Path to a file with a list of domains. File should have one domain per line, with no extra characters or comments. See *nod.unique_response_ignore_list*.

6.5.98 nod.unique_response_log

New in version 4.2.0.

- Boolean
- Default: `true`
- Old style setting: *unique-response-log*

Whether to log when a unique response is detected. The log line looks something like:

Oct 24 12:11:27 Unique response observed: qname=foo.com qtype=A rrtype=AAAA rrname=foo.com rrcontent=1.2.3.4

6.5.99 nod.unique_response_pb_tag

New in version 4.2.0.

- String
- Default: `pdns-udr`
- Old style setting: *unique-response-pb-tag*

If protobuf is configured, then this tag will be added to all protobuf response messages when a unique DNS response is observed.

6.5.100 nod.unique_response_tracking

New in version 4.2.0.

- Boolean
- Default: `false`
- Old style setting: *unique-response-tracking*

Whether to track unique DNS responses, i.e. never seen before combinations of the triplet (query name, query type, RR[rrname, rrtype, rrdata]). This can be useful for tracking potentially suspicious domains and behaviour, e.g. DNS fast-flux. If protobuf is enabled and configured, then the Protobuf Response message will contain a flag with udr set to true for each RR that is considered unique, i.e. never seen before. This feature uses a probabilistic data structure (stable bloom filter) to track unique responses, which can have false positives as well as false negatives, thus it is a best-effort feature. Increasing the number of cells in the SBF using the unique-response-db-size setting can reduce FPs and FNs.

6.5.101 outgoing.bypass_server_throttling_probability

New in version 5.0.0.

- Integer
- Default: `25`
- Old style setting: *bypass-server-throttling-probability*

This setting determines the probability of a server marked down to be used anyway. A value of `n` means that the chance of a server marked down still being used after it wins speed selection is is `1/n`. If this setting is zero throttled servers will never be selected to be used anyway.

6.5.102 outgoing.dont_query

- Sequence of *Subnet* (IP addresses or subnets, negation supported)
- Default: `[127.0.0.0/8, 10.0.0.0/8, 100.64.0.0/10, 169.254.0.0/16, 192.168.0.0/16, 172.16.0.0/12, '::1/128', 'fc00::/7', 'fe80::/10', 0.0.0.0/8, 192.0.0.0/24, 192.0.2.0/24, 198.51.100.0/24, 203.0.113.0/24, 240.0.0.0/4, '::/96', '::ffff:0:0/96', '100::/64', '2001:db8::/32']`
- Old style setting: *dont-query*

The DNS is a public database, but sometimes contains delegations to private IP addresses, like for example 127.0.0.1. This can have odd effects, depending on your network, and may even be a security risk. Therefore, the PowerDNS Recursor by default does not query private space IP addresses. This setting can be used to expand or reduce the limitations.

Queries for names in forward zones and to addresses as configured in any of the settings *recursor.forward_zones*, *recursor.forward_zones_file* or *recursor.forward_zones_recurse* are performed regardless of these limitations.

6.5.103 outgoing.dont_throttle_names

New in version 4.2.0.

- Sequence of strings
- Default: `[]`
- Old style setting: *dont-throttle-names*

When an authoritative server does not answer a query or sends a reply the recursor does not like, it is throttled. Any servers' name suffix-matching the supplied names will never be throttled.

> **Warning:** Most servers on the internet do not respond for a good reason (overloaded or unreachable), `dont-throttle-names` could make this load on the upstream server even higher, resulting in further service degradation.

6.5.104 outgoing.dont_throttle_netmasks

New in version 4.2.0.

- Sequence of *Subnet* (IP addresses or subnets, negation supported)
- Default: `[]`
- Old style setting: *dont-throttle-netmasks*

When an authoritative server does not answer a query or sends a reply the recursor does not like, it is throttled. Any servers matching the supplied netmasks will never be throttled.

This can come in handy on lossy networks when forwarding, where the same server is configured multiple times (e.g. with `forward_zones_recurse: [ {zone: example.com, forwarders: [ 192.0.2.1, 192.0.2.1 ] } ]`. By default, the PowerDNS Recursor would throttle the 'first' server on a timeout and hence not retry the 'second' one. In this case, *outgoing.dont_throttle_netmasks* could be set to include `192.0.2.1`.

> **Warning:** Most servers on the internet do not respond for a good reason (overloaded or unreachable), `dont-throttle-netmasks` could make this load on the upstream server even higher, resulting in further service degradation.

6.5.105 `outgoing.dot_to_auth_names`

New in version 4.6.0.

- Sequence of strings
- Default: `[]`
- Old style setting: *dot-to-auth-names*

Force DoT to the listed authoritative nameservers. For this to work, DoT support has to be compiled in. Currently, the certificate is not checked for validity in any way.

6.5.106 `outgoing.dot_to_port_853`

New in version 4.6.0.

- Boolean
- Default: `true`
- Old style setting: *dot-to-port-853*

Enable DoT to forwarders that specify port 853.

6.5.107 `outgoing.edns_bufsize`

Changed in version 4.2.0: Before 4.2.0, the default was 1680

- Integer
- Default: `1232`
- Old style setting: *edns-outgoing-bufsize*

Note: Why 1232?

1232 is the largest number of payload bytes that can fit in the smallest IPv6 packet. IPv6 has a minimum MTU of 1280 bytes (**RFC 8200, section 5**), minus 40 bytes for the IPv6 header, minus 8 bytes for the UDP header gives 1232, the maximum payload size for the DNS response.

This is the value set for the EDNS0 buffer size in outgoing packets. Lower this if you experience timeouts.

6.5.108 `outgoing.edns_padding`

New in version 4.8.0.

- Boolean
- Default: `true`
- Old style setting: *edns-padding-out*

Whether to add EDNS padding to outgoing DoT queries.

6.5.109 `outgoing.edns_subnet_allow_list`

New in version 4.5.0.

- Sequence of strings
- Default: `[]`

- Old style setting: *edns-subnet-allow-list*

List of netmasks and domains that **EDNS Client Subnet** should be enabled for in outgoing queries.

For example, an EDNS Client Subnet option containing the address of the initial requestor (but see *ecs.add_for*) will be added to an outgoing query sent to server 192.0.2.1 for domain X if 192.0.2.1 matches one of the supplied netmasks, or if X matches one of the supplied domains. The initial requestor address will be truncated to 24 bits for IPv4 (see *ecs.ipv4_bits*) and to 56 bits for IPv6 (see *ecs.ipv6_bits*), as recommended in the privacy section of RFC 7871.

Note that this setting describes the destination of outgoing queries, not the sources of incoming queries, nor the subnets described in the EDNS Client Subnet option.

By default, this option is empty, meaning no EDNS Client Subnet information is sent.

6.5.110 `outgoing.lowercase`

- Boolean
- Default: `false`
- Old style setting: *lowercase-outgoing*

Set to true to lowercase the outgoing queries. When set to 'no' (the default) a query from a client using mixed case in the DNS labels (such as a user entering mixed-case names or draft-vixie-dnsext-dns0x20-00), PowerDNS preserves the case of the query. Broken authoritative servers might give a wrong or broken answer on this encoding. Setting `lowercase-outgoing` to 'yes' makes the PowerDNS Recursor lowercase all the labels in the query to the authoritative servers, but still return the proper case to the client requesting.

6.5.111 `outgoing.max_busy_dot_probes`

New in version 4.7.0.

- Integer
- Default: `0`
- Old style setting: *max-busy-dot-probes*

Limit the maximum number of simultaneous DoT probes the Recursor will schedule. The default value 0 means no DoT probes are scheduled.

DoT probes are used to check if an authoritative server's IP address supports DoT. If the probe determines an IP address supports DoT, the Recursor will use DoT to contact it for subsequent queries until a failure occurs. After a failure, the Recursor will stop using DoT for that specific IP address for a while. The results of probes are remembered and can be viewed by the `rec_control dump-dot-probe-map` command. If the maximum number of pending probes is reached, no probes will be scheduled, even if no DoT status is known for an address. If the result of a probe is not yet available, the Recursor will contact the authoritative server in the regular way, unless an authoritative server is configured to be contacted over DoT always using *outgoing.dot_to_auth_names*. In that case no probe will be scheduled.

Note: DoT probing is an experimental feature. Please test thoroughly to determine if it is suitable in your specific production environment before enabling.

6.5.112 `outgoing.max_ns_address_qperq`

New in version 4.1.16.

New in version 4.2.2.

New in version 4.3.1.

- Integer
- Default: `10`
- Old style setting: *max-ns-address-qperq*

The maximum number of outgoing queries with empty replies for resolving nameserver names to addresses we allow during the resolution of a single client query. If IPv6 is enabled, an A and a AAAA query for a name counts as 1. If a zone publishes more than this number of NS records, the limit is further reduced for that zone by lowering it by the number of NS records found above the *outgoing.max_ns_address_qperq* value. The limit wil not be reduced to a number lower than 5.

6.5.113 `outgoing.max_ns_per_resolve`

New in version 4.8.0.

New in version 4.7.3.

New in version 4.6.4.

New in version 4.5.11.

- Integer
- Default: `13`
- Old style setting: *max-ns-per-resolve*

The maximum number of NS records that will be considered to select a nameserver to contact to resolve a name. If a zone has more than *outgoing.max_ns_per_resolve* NS records, a random sample of this size will be used. If *outgoing.max_ns_per_resolve* is zero, no limit applies.

6.5.114 `outgoing.max_qperq`

Changed in version 5.1.0: The default used to be 60, with an extra allowance if qname minimization was enabled. Having better algorithms allows for a lower default limit.

- Integer
- Default: `50`
- Old style setting: *max-qperq*

The maximum number of outgoing queries that will be sent out during the resolution of a single client query. This is used to avoid cycles resolving names.

6.5.115 `outgoing.network_timeout`

- Integer
- Default: `1500`
- Old style setting: *network-timeout*

Number of milliseconds to wait for a remote authoritative server to respond. If the number of concurrent requests is high, the :program:Recursor uses a lower value.

6.5.116 `outgoing.non_resolving_ns_max_fails`

New in version 4.5.0.

- Integer
- Default: `5`

- Old style setting: *non-resolving-ns-max-fails*

Number of failed address resolves of a nameserver name to start throttling it, 0 is disabled. Nameservers matching *outgoing.dont_throttle_names* will not be throttled.

6.5.117 `outgoing.non_resolving_ns_throttle_time`

New in version 4.5.0.

- Integer
- Default: `60`
- Old style setting: *non-resolving-ns-throttle-time*

Number of seconds to throttle a nameserver with a name failing to resolve.

6.5.118 `outgoing.server_down_max_fails`

- Integer
- Default: `64`
- Old style setting: *server-down-max-fails*

If a server has not responded in any way this many times in a row, no longer send it any queries for *outgoing.server_down_throttle_time* seconds. Afterwards, we will try a new packet, and if that also gets no response at all, we again throttle for *outgoing.server_down_throttle_time* seconds. Even a single response packet will drop the block.

6.5.119 `outgoing.server_down_throttle_time`

- Integer
- Default: `60`
- Old style setting: *server-down-throttle-time*

Throttle a server that has failed to respond *outgoing.server_down_max_fails* times for this many seconds.

6.5.120 `outgoing.single_socket`

- Boolean
- Default: `false`
- Old style setting: *single-socket*

Use only a single socket for outgoing queries.

6.5.121 `outgoing.source_address`

Changed in version 4.4.0: IPv6 addresses can be set with this option as well.

- Sequence of *Subnet* (IP addresses or subnets, negation supported)
- Default: `[0.0.0.0]`
- Old style setting: *query-local-address*

Note: While subnets and their negations are syntactically accepted, the handling of subnets has not been implemented yet. Only individual IP addresses can be listed.

Send out local queries from this address, or addresses. By adding multiple addresses, increased spoofing resilience is achieved. When no address of a certain address family is configured, there are *no* queries sent with that address family. In the default configuration this means that IPv6 is not used for outgoing queries.

6.5.122 `outgoing.tcp_fast_open_connect`

New in version 4.5.0.

- Boolean
- Default: `false`
- Old style setting: *tcp-fast-open-connect*

Enable TCP Fast Open Connect support, if available, on the outgoing connections to authoritative servers. See *TCP Fast Open Support*.

6.5.123 `outgoing.tcp_max_idle_ms`

New in version 4.6.0.

- Integer
- Default: `10000`
- Old style setting: *tcp-out-max-idle-ms*

Time outgoing TCP/DoT connections are left idle in milliseconds or 0 if no limit. After having been idle for this time, the connection is eligible for closing.

6.5.124 `outgoing.tcp_max_idle_per_auth`

New in version 4.6.0.

- Integer
- Default: `10`
- Old style setting: *tcp-out-max-idle-per-auth*

Maximum number of idle outgoing TCP/DoT connections to a specific IP per thread, 0 means do not keep idle connections open.

6.5.125 `outgoing.tcp_max_idle_per_thread`

New in version 4.6.0.

- Integer
- Default: `100`
- Old style setting: *tcp-out-max-idle-per-thread*

Maximum number of idle outgoing TCP/DoT connections per thread, 0 means do not keep idle connections open.

6.5.126 outgoing.tcp_max_queries

- Integer
- Default: `0`
- Old style setting: *tcp-out-max-queries*

Maximum total number of queries per outgoing TCP/DoT connection, 0 means no limit. After this number of queries, the connection is closed and a new one will be created if needed.

6.5.127 outgoing.udp_source_port_avoid

New in version 4.2.0.

- Sequence of strings
- Default: `[11211]`
- Old style setting: *udp-source-port-avoid*

A sequence of UDP port numbers to avoid when binding. For example:

```
outgoing:
  udp_source_port_avoid:
  - 5300
  - 11211
```

See *outgoing.udp_source_port_min*.

6.5.128 outgoing.udp_source_port_max

New in version 4.2.0.

- Integer
- Default: `65535`
- Old style setting: *udp-source-port-max*

This option sets the maximum limit of UDP port number to bind on.

See *outgoing.udp_source_port_min*.

6.5.129 outgoing.udp_source_port_min

New in version 4.2.0.

- Integer
- Default: `1024`
- Old style setting: *udp-source-port-min*

This option sets the low limit of UDP port number to bind on.

In combination with *outgoing.udp_source_port_max* it configures the UDP port range to use. Port numbers are randomized within this range on initialization, and exceptions can be configured with *outgoing.udp_source_port_avoid*

6.5.130 packetcache.disable

- Boolean
- Default: `false`
- Old style setting: *disable-packetcache*

Turn off the packet cache. Useful when running with Lua scripts that modify answers in such a way they cannot be cached, though individual answer caching can be controlled from Lua as well.

6.5.131 packetcache.max_entries

- Integer
- Default: `500000`
- Old style setting: *max-packetcache-entries*

Maximum number of Packet Cache entries. Sharded and shared by all threads since 4.9.0.

6.5.132 packetcache.negative_ttl

New in version 4.9.0.

- Integer
- Default: `60`
- Old style setting: *packetcache-negative-ttl*

Maximum number of seconds to cache an `NxDomain` or `NoData` answer in the packetcache. This setting's maximum is capped to *packetcache.ttl*. i.e. setting `packetcache-ttl=15` and keeping `packetcache-negative-ttl` at the default will lower `packetcache-negative-ttl` to `15`.

6.5.133 packetcache.servfail_ttl

'versionchanged': ('4.0.0', "This setting's maximum is capped to *packetcache.ttl*. i.e. setting `packetcache-ttl=15` and keeping `packetcache-servfail-ttl` at the default will lower `packetcache-servfail-ttl` to `15`.")

- Integer
- Default: `60`
- Old style setting: *packetcache-servfail-ttl*

Maximum number of seconds to cache an answer indicating a failure to resolve in the packet cache. Before version 4.6.0 only `ServFail` answers were considered as such. Starting with 4.6.0, all responses with a code other than `NoError` and `NXDomain`, or without records in the answer and authority sections, are considered as a failure to resolve. Since 4.9.0, negative answers are handled separately from resolving failures.

6.5.134 packetcache.shards

New in version 4.9.0.

- Integer
- Default: `1024`
- Old style setting: *packetcache-shards*

Sets the number of shards in the packet cache. If you have high contention as reported by `packetcache-contented/packetcache-acquired`, you can try to enlarge this value or run with fewer threads.

6.5.135 `packetcache.ttl`

Changed in version 4.9.0: The default was changed from 3600 (1 hour) to 86400 (24 hours).

- Integer
- Default: `86400`
- Old style setting: *packetcache-ttl*

Maximum number of seconds to cache an item in the packet cache, no matter what the original TTL specified.

6.5.136 `recordcache.locked_ttl_perc`

New in version 4.8.0.

- Integer
- Default: `0`
- Old style setting: *record-cache-locked-ttl-perc*

Replace record sets in the record cache only after this percentage of the original TTL has passed. The PowerDNS Recursor already has several mechanisms to protect against spoofing attempts. This adds an extra layer of protection—as it limits the window of time cache updates are accepted—at the cost of a less efficient record cache.

The default value of 0 means no extra locking occurs. When non-zero, record sets received (e.g. in the Additional Section) will not replace existing record sets in the record cache until the given percentage of the original TTL has expired. A value of 100 means only expired record sets will be replaced.

There are a few cases where records will be replaced anyway:

- Record sets that are expired will always be replaced.
- Authoritative record sets will replace unauthoritative record sets unless DNSSEC validation of the new record set failed.
- If the new record set belongs to a DNSSEC-secure zone and successfully passed validation it will replace an existing entry.
- Record sets produced by *recordcache.refresh_on_ttl_perc* tasks will also replace existing record sets.

6.5.137 `recordcache.max_cache_bogus_ttl`

New in version 4.2.0.

- Integer
- Default: `3600`
- Old style setting: *max-cache-bogus-ttl*

Maximum number of seconds to cache an item in the DNS cache (negative or positive) if its DNSSEC validation failed, no matter what the original TTL specified, to reduce the impact of a broken domain.

6.5.138 recordcache.max_entries

- Integer
- Default: `1000000`
- Old style setting: *max-cache-entries*

Maximum number of DNS record cache entries, shared by all threads since 4.4.0. Each entry associates a name and type with a record set. The size of the negative cache is 10% of this number.

6.5.139 recordcache.max_negative_ttl

- Integer
- Default: `3600`
- Old style setting: *max-negative-ttl*

A query for which there is authoritatively no answer is cached to quickly deny a record's existence later on, without putting a heavy load on the remote server. In practice, caches can become saturated with hundreds of thousands of hosts which are tried only once. This setting, which defaults to 3600 seconds, puts a maximum on the amount of time negative entries are cached.

6.5.140 recordcache.max_ttl

Changed in version 4.1.0: The minimum value of this setting is 15. i.e. setting this to lower than 15 will make this value 15.

- Integer
- Default: `86400`
- Old style setting: *max-cache-ttl*

Maximum number of seconds to cache an item in the DNS cache, no matter what the original TTL specified. This value also controls the refresh period of cached root data. See *Handling of root hints* for more information on this.

6.5.141 recordcache.refresh_on_ttl_perc

New in version 4.5.0.

- Integer
- Default: `0`
- Old style setting: *refresh-on-ttl-perc*

Sets the 'refresh almost expired' percentage of the record cache. Whenever a record is fetched from the packet or record cache and only `refresh-on-ttl-perc` percent or less of its original TTL is left, a task is queued to refetch the name/type combination to update the record cache. In most cases this causes future queries to always see a non-expired record cache entry. A typical value is 10. If the value is zero, this functionality is disabled.

6.5.142 recordcache.serve_stale_extensions

New in version 4.8.0.

- Integer
- Default: `0`
- Old style setting: *serve-stale-extensions*

Maximum number of times an expired record's TTL is extended by 30s when serving stale. Extension only occurs if a record cannot be refreshed. A value of 0 means the `Serve Stale` mechanism is not used. To allow records becoming stale to be served for an hour, use a value of 120. See *Serve Stale* for a description of the Serve Stale mechanism.

6.5.143 `recordcache.shards`

New in version 4.4.0.

- Integer
- Default: `1024`
- Old style setting: *record-cache-shards*

Sets the number of shards in the record cache. If you have high contention as reported by `record-cache-contented/record-cache-acquired`, you can try to enlarge this value or run with fewer threads.

6.5.144 `recordcache.zonetocaches`

New in version 5.1.0.

- Sequence of *ZoneToCache*
- Default: `[]`
- Equivalent Lua config in *Zone to Cache*

Sequence of ZoneToCache entries

6.5.145 `recursor.allow_trust_anchor_query`

New in version 4.3.0.

- Boolean
- Default: `false`
- Old style setting: *allow-trust-anchor-query*

Allow `trustanchor.server CH TXT` and `negativetrustanchor.server CH TXT` queries to view the configured *DNSSEC* (negative) trust anchors.

6.5.146 `recursor.allowed_additional_qtypes`

New in version 5.1.0.

- Sequence of *AllowedAdditionalQType*
- Default: `[]`
- Equivalent Lua config in *Adding Additional Records to Results*

Sequence of AllowedAdditionalQType

6.5.147 `recursor.any_to_tcp`

- Boolean
- Default: `false`
- Old style setting: *any-to-tcp*

Answer questions for the ANY type on UDP with a truncated packet that refers the remote server to TCP. Useful for mitigating ANY reflection attacks.

6.5.148 recursor.auth_zones

- Sequence of *Auth Zone*
- Default: `[]`
- Old style setting: *auth-zones*

Zones read from these files (in BIND format) are served authoritatively (but without the AA bit set in responses). DNSSEC is not supported. Example:

```
recursor:
   auth-zones:
   - zone: example.org
     file: /var/zones/example.org
   - zone: powerdns.com
     file: /var/zones/powerdns.com
```

6.5.149 recursor.chroot

- String
- Default: (empty)
- Old style setting: *chroot*

If set, chroot to this directory for more security. This is not recommended; instead, we recommend containing PowerDNS using operating system features. We ship systemd unit files with our packages to make this easy.

Make sure that `/dev/log` is available from within the chroot. Logging will silently fail over time otherwise (on logrotate).

When using `chroot`, all other paths (except for *recursor.config_dir*) set in the configuration are relative to the new root.

When running on a system where systemd manages services, `chroot` does not work out of the box, as PowerDNS cannot use the `NOTIFY_SOCKET`. Either do not `chroot` on these systems or set the 'Type' of this service to 'simple' instead of 'notify' (refer to the systemd documentation on how to modify unit-files).

6.5.150 recursor.config_dir

- String
- Default: Determined by distribution
- Old style setting: *config-dir*

Location of configuration directory (where `recursor.conf` or `recursor.yml` is stored). Usually `/etc/powerdns`, but this depends on `SYSCONFDIR` during compile-time. Use default or set on command line.

6.5.151 recursor.config_name

- String
- Default: (empty)
- Old style setting: *config-name*

When running multiple recursors on the same server, read settings from `recursor-name.conf`, this will also rename the binary image.

6.5.152 recursor.cpu_map

- String
- Default: (empty)
- Old style setting: *cpu-map*

Set CPU affinity for threads, asking the scheduler to run those threads on a single CPU, or a set of CPUs. This parameter accepts a space separated list of thread-id=cpu-id, or thread-id=cpu-id-1,cpu-id-2,...,cpu-id-N. For example, to make the worker thread 0 run on CPU id 0 and the worker thread 1 on CPUs 1 and 2:

```
recursor:
  cpu_map: 0=0 1=1,2
```

The thread handling the control channel, the webserver and other internal stuff has been assigned id 0, the distributor threads if any are assigned id 1 and counting, and the worker threads follow behind. The number of distributor threads is determined by *incoming.distributor_threads*, the number of worker threads is determined by the *recursor.threads* setting.

This parameter is only available if the OS provides the `pthread_setaffinity_np()` function.

Note that depending on the configuration the Recursor can start more threads. Typically these threads will sleep most of the time. These threads cannot be specified in this setting as their thread-ids are left unspecified.

6.5.153 recursor.daemon

Changed in version 4.0.0: Default is now `no`, was `yes` before.

- Boolean
- Default: `false`
- Old style setting: *daemon*

Operate in the background.

6.5.154 recursor.dns64_prefix

New in version 4.4.0.

- String
- Default: (empty)
- Old style setting: *dns64-prefix*

Enable DNS64 (**RFC 6147**) support using the supplied /96 IPv6 prefix. This will generate 'fake' `AAAA` records for names with only `A` records, as well as 'fake' `PTR` records to make sure that reverse lookup of DNS64-generated IPv6 addresses generate the right name. See *DNS64 support* for more flexible but slower alternatives using Lua.

6.5.155 recursor.etc_hosts_file

- String
- Default: `/etc/hosts`
- Old style setting: *etc-hosts-file*

The path to the /etc/hosts file, or equivalent. This file can be used to serve data authoritatively using *recursor.export_etc_hosts*.

6.5.156 recursor.event_trace_enabled

New in version 4.6.0.

- Integer
- Default: `0`
- Old style setting: *event-trace-enabled*

Enable the recording and logging of ref:*event traces*. This is an experimental feature and subject to change. Possible values are 0: (disabled), 1 (add information to protobuf logging messages) and 2 (write to log) and 3 (both).

6.5.157 recursor.export_etc_hosts

- Boolean
- Default: `false`
- Old style setting: *export-etc-hosts*

If set, this flag will export the host names and IP addresses mentioned in `/etc/hosts`.

6.5.158 recursor.export_etc_hosts_search_suffix

- String
- Default: (empty)
- Old style setting: *export-etc-hosts-search-suffix*

If set, all hostnames in the *recursor.export_etc_hosts* file are loaded in canonical form, based on this suffix, unless the name contains a '.', in which case the name is unchanged. So an entry called 'pc' with `export-etc-hosts-search-suffix='home.com'` will lead to the generation of 'pc.home.com' within the recursor. An entry called 'server1.home' will be stored as 'server1.home', regardless of this setting.

6.5.159 recursor.extended_resolution_errors

New in version 4.5.0.

Changed in version 5.0.0: Default changed to enabled, previously it was disabled.

- Boolean
- Default: `true`
- Old style setting: *extended-resolution-errors*

If set, the recursor will add an EDNS Extended Error (**RFC 8914**) to responses when resolution failed, like DNSSEC validation errors, explaining the reason it failed. This setting is not needed to allow setting custom error codes from Lua or from a RPZ hit.

6.5.160 recursor.forward_zones

- Sequence of *Forward Zone*
- Default: `[]`
- Old style setting: *forward-zones*

Queries for zones listed here will be forwarded to the IP address listed. i.e.

```
recursor:
   forward-zones:
     - zone: example.org
       forwarders:
       - 203.0.113.210
     - zone: powerdns.com
       forwarders:
       - 2001:DB8::BEEF:5
```

Multiple IP addresses can be specified and port numbers other than 53 can be configured:

```
recursor:
  forward-zones:
  - zone: example.org
    forwarders:
    - 203.0.113.210:5300
    - 127.0.0.1
  - zone: powerdns.com
    forwarders:
    - 127.0.0.1
    - 198.51.100.10:530
    - '[2001:DB8::1:3]:5300'
```

Forwarded queries have the `recursion desired (RD)` bit set to `0`, meaning that this setting is intended to forward queries to authoritative servers. If an `NS` record set for a subzone of the forwarded zone is learned, that record set will be used to determine addresses for name servers of the subzone. This allows e.g. a forward to a local authoritative server holding a copy of the root zone, delegations received from that server will work.

IMPORTANT: When using DNSSEC validation (which is default), forwards to non-delegated (e.g. internal) zones that have a DNSSEC signed parent zone will validate as Bogus. To prevent this, add a Negative Trust Anchor (NTA) for this zone in the *recursor.lua_config_file* with `addNTA('your.zone', 'A comment')`. If this forwarded zone is signed, instead of adding NTA, add the DS record to the *recursor.lua_config_file*. See the *DNSSEC in the PowerDNS Recursor* information.

6.5.161 recursor.forward_zones_file

Changed in version 4.0.0: (Old style settings only) Comments are allowed, everything behind # is ignored.

Changed in version 4.6.0: (Old style settings only) Zones prefixed with a ^ are added to the *allow-notify-for* list. Both prefix characters can be used if desired, in any order.

- String
- Default: (empty)
- Old style setting: *forward-zones-file*

Same as *recursor.forward_zones*, parsed from a file as a sequence of *ZoneForward*.

```
- zone: example1.com
  forwarders:
  - 127.0.0.1
  - 127.0.0.1:5353
  - '[::1]:53'
- zone: example2.com
  forwarders:
  - ::1
  recurse: true
  notify_allowed: true
```

The DNSSEC notes from *recursor.forward_zones* apply here as well.

6.5.162 recursor.forward_zones_recurse

- Sequence of *Forward Zone*
- Default: `[]`
- Old style setting: *forward-zones-recurse*

Like regular *recursor.forward_zones*, but forwarded queries have the `recursion desired (RD)` bit set to `1`, meaning that this setting is intended to forward queries to other recursive servers. In contrast to regular forwarding, the rule that delegations of the forwarded subzones are respected is not active. This is because we rely on the forwarder to resolve the query fully.

See *recursor.forward_zones* for additional options (such as supplying multiple recursive servers) and an important note about DNSSEC.

6.5.163 recursor.hint_file

Changed in version 4.6.2: Introduced the value `no` to disable root-hints processing.

Changed in version 4.9.0: Introduced the value `no-refresh` to disable both root-hints processing and periodic refresh of the cached root *NS* records.

- String
- Default: (empty)
- Old style setting: *hint-file*

If set, the root-hints are read from this file. If empty, the default built-in root hints are used.

In some special cases, processing the root hints is not needed, for example when forwarding all queries to another recursor. For these special cases, it is possible to disable the processing of root hints by setting the value to `no` or `no-refresh`. See *Handling of root hints* for more information on root hints handling.

6.5.164 recursor.ignore_unknown_settings

- Sequence of strings
- Default: `[]`
- Old style setting: *ignore-unknown-settings*

Names of settings to be ignored while parsing configuration files, if the setting name is unknown to PowerDNS.

Useful during upgrade testing.

6.5.165 recursor.include_dir

- String
- Default: (empty)
- Old style setting: *include-dir*

Directory to scan for additional config files. All files that end with `.yml` are loaded in order using `POSIX` as locale.

6.5.166 recursor.latency_statistic_size

- Integer
- Default: `10000`

- Old style setting: *latency-statistic-size*

Indication of how many queries will be averaged to get the average latency reported by the 'qa-latency' metric.

6.5.167 recursor.lua_config_file

- String
- Default: (empty)
- Old style setting: *lua-config-file*

If set, and Lua support is compiled in, this will load an additional configuration file for newer features and more complicated setups. See *Advanced Configuration Using Lua* for the options that can be set in this file.

6.5.168 recursor.lua_dns_script

- String
- Default: (empty)
- Old style setting: *lua-dns-script*

Path to a lua file to manipulate the Recursor's answers. See *Scripting PowerDNS Recursor* for more information.

6.5.169 recursor.lua_global_include_dir

- String
- Default: (empty)
- Old style setting: *lua-global-include-dir*

When creating a Lua context, all *.lua files in the directory are loaded to the Lua context.

6.5.170 recursor.lua_maintenance_interval

New in version 4.2.0.

- Integer
- Default: `1`
- Old style setting: *lua-maintenance-interval*

The interval between calls to the Lua user defined *maintenance()* function in seconds. See *Maintenance callback*

6.5.171 recursor.max_chain_length

New in version 5.1.0.

- Integer
- Default: `0`
- Old style setting: *max-chain-length*

The maximum number of queries that can be attached to an outgoing request chain. Attaching requests to a chain saves on outgoing queries, but the processing of a chain when the reply to the outgoing query comes in might result in a large outgoing traffic spike. Reducing the maximum chain length mitigates this. If this value is zero, no maximum is enforced, though the maximum number of mthreads (*recursor.max_mthreads*) also limits the chain length.

6.5.172 recursor.max_cnames_followed

New in version 5.1.0.

- Integer
- Default: `10`
- Old style setting: *max-cnames-followed*

Maximum length of a CNAME chain. If a CNAME chain exceeds this length, a `ServFail` answer will be returned. Previously, this limit was fixed at 10.

6.5.173 recursor.max_generate_steps

New in version 4.3.0.

- Integer
- Default: `0`
- Old style setting: *max-generate-steps*

Maximum number of steps for a '$GENERATE' directive when parsing a zone file. This is a protection measure to prevent consuming a lot of CPU and memory when untrusted zones are loaded. Default to 0 which means unlimited.

6.5.174 recursor.max_include_depth

New in version 4.6.0.

- Integer
- Default: `20`
- Old style setting: *max-include-depth*

Maximum number of nested `$INCLUDE` directives while processing a zone file. Zero mean no `$INCLUDE` directives will be accepted.

6.5.175 recursor.max_mthreads

- Integer
- Default: `2048`
- Old style setting: *max-mthreads*

Maximum number of simultaneous MTasker threads, per worker thread.

6.5.176 recursor.max_recursion_depth

Changed in version 4.1.0: Before 4.1.0, this settings was unlimited.

Changed in version 4.9.0: Before 4.9.0 this setting's default was 40 and the limit on `CNAME` chains (fixed at 16) acted as a bound on he recursion depth.

- Integer
- Default: `16`
- Old style setting: *max-recursion-depth*

Total maximum number of internal recursion calls the server may use to answer a single query. 0 means unlimited. The value of *recursor.stack_size* should be increased together with this one to prevent the stack from overflowing. If *recursor.qname_minimization* is enabled, the fallback code in case of a failing resolve is allowed an additional *max-recursion-depth/2*.

6.5.177 `recursor.max_total_msec`

- Integer
- Default: `7000`
- Old style setting: *max-total-msec*

Total maximum number of milliseconds of wallclock time the server may use to answer a single query. 0 means unlimited.

6.5.178 `recursor.minimum_ttl_override`

Changed in version 4.5.0: Old versions used default 0.

- Integer
- Default: `1`
- Old style setting: *minimum-ttl-override*

This setting artificially raises all TTLs to be at least this long. Setting this to a value greater than 1 technically is an RFC violation, but might improve performance a lot. Using a value of 0 impacts performance of TTL 0 records greatly, since it forces the recursor to contact authoritative servers each time a client requests them. Can be set at runtime using `rec_control set-minimum-ttl 3600`.

6.5.179 `recursor.nothing_below_nxdomain`

New in version 4.3.0.

- String
- Default: `dnssec`
- Old style setting: *nothing-below-nxdomain*
- One of `no`, `dnssec`, `yes`.

The type of **RFC 8020** handling using cached NXDOMAIN responses. This RFC specifies that NXDOMAIN means that the DNS tree under the denied name MUST be empty. When an NXDOMAIN exists in the cache for a shorter name than the qname, no lookup is done and an NXDOMAIN is sent to the client.

For instance, when `foo.example.net` is negatively cached, any query matching `*.foo.example.net` will be answered with NXDOMAIN directly without consulting authoritative servers.

`no` No **RFC 8020** processing is done.

`dnssec` **RFC 8020** processing is only done using cached NXDOMAIN records that are DNSSEC validated.

`yes` **RFC 8020** processing is done using any non-Bogus NXDOMAIN record available in the cache.

6.5.180 `recursor.public_suffix_list_file`

New in version 4.2.0.

- String
- Default: (empty)

- Old style setting: *public-suffix-list-file*

Path to the Public Suffix List file, if any. If set, PowerDNS will try to load the Public Suffix List from this file instead of using the built-in list. The PSL is used to group the queries by relevant domain names when displaying the top queries.

6.5.181 `recursor.qname_max_minimize_count`

New in version 5.0.0.

- Integer
- Default: `10`
- Old style setting: *qname-max-minimize-count*

`Max minimize count` parameter, described in **RFC 9156**. This is the maximum number of iterations of the Query Name Minimization Algorithm.

6.5.182 `recursor.qname_minimization`

New in version 4.3.0.

- Boolean
- Default: `true`
- Old style setting: *qname-minimization*

Enable Query Name Minimization. This implements a relaxed form of Query Name Mimimization as described in **RFC 9156**.

6.5.183 `recursor.qname_minimize_one_label`

New in version 5.0.0.

- Integer
- Default: `4`
- Old style setting: *qname-minimize-one-label*

`Minimize one label` parameter, described in **RFC 9156**. The value for the number of iterations of the Query Name Minimization Algorithm that should only have one label appended. This value has precedence over *recursor.qname_max_minimize_count*.

6.5.184 `recursor.root_nx_trust`

Changed in version 4.0.0: Default is `yes` now, was `no` before 4.0.0

- Boolean
- Default: `true`
- Old style setting: *root-nx-trust*

If set, an NXDOMAIN from the root-servers will serve as a blanket NXDOMAIN for the entire TLD the query belonged to. The effect of this is far fewer queries to the root-servers.

6.5.185 recursor.rpzs

New in version 5.1.0.

- Sequence of *RPZ*
- Default: `[]`
- Equivalent Lua config in *Response Policy Zones (RPZ)*

Sequence of RPZ entries.

6.5.186 recursor.save_parent_ns_set

New in version 4.7.0.

- Boolean
- Default: `true`
- Old style setting: *save-parent-ns-set*

If set, a parent (non-authoritative) `NS` set is saved if it contains more entries than a newly encountered child (authoritative) `NS` set for the same domain. The saved parent `NS` set is tried if resolution using the child `NS` set fails.

6.5.187 recursor.security_poll_suffix

- String
- Default: `secpoll.powerdns.com.`
- Old style setting: *security-poll-suffix*

Domain name from which to query security update notifications. Setting this to an empty string disables secpoll.

6.5.188 recursor.serve_rfc1918

- Boolean
- Default: `true`
- Old style setting: *serve-rfc1918*

This makes the server authoritatively aware of: `10.in-addr.arpa`, `168.192.in-addr.arpa`, `16-31.172.in-addr.arpa`, which saves load on the AS112 servers. Individual parts of these zones can still be loaded or forwarded.

6.5.189 recursor.server_id

- String
- Default: `*runtime determined*`
- Old style setting: *server-id*

The reply given by The PowerDNS recursor to a query for 'id.server' with its hostname, useful for in clusters. When a query contains the **NSID EDNS0 Option**, this value is returned in the response as the NSID value.

This setting can be used to override the answer given to these queries. Set to 'disabled' to disable NSID and 'id.server' answers.

Query example (where 192.0.2.14 is your server):

```
dig @192.0.2.14 CHAOS TXT id.server.
dig @192.0.2.14 example.com IN A +nsid
```

6.5.190 recursor.setgid

- String
- Default: (empty)
- Old style setting: *setgid*

PowerDNS can change its user and group id after binding to its socket. Can be used for better *security*.

6.5.191 recursor.setuid

- String
- Default: (empty)
- Old style setting: *setuid*

PowerDNS can change its user and group id after binding to its socket. Can be used for better *security*.

6.5.192 recursor.socket_dir

- String
- Default: (empty)
- Old style setting: *socket-dir*

Where to store the control socket and pidfile. The default depends on `LOCALSTATEDIR` or the `--with-socketdir` setting when building (usually `/var/run` or `/run`).

When using *recursor.chroot* the default becomes `/`. The default value is overruled by the `RUNTIME_DIRECTORY` environment variable when that variable has a value (e.g. under systemd).

6.5.193 recursor.socket_group

- String
- Default: (empty)
- Old style setting: *socket-group*

Group and mode of the controlsocket. Owner and group can be specified by name, mode is in octal.

6.5.194 recursor.socket_mode

- String
- Default: (empty)
- Old style setting: *socket-mode*

Mode of the controlsocket. Owner and group can be specified by name, mode is in octal.

6.5.195 recursor.socket_owner

- String
- Default: (empty)
- Old style setting: *socket-owner*

Owner of the controlsocket. Owner and group can be specified by name, mode is in octal.

6.5.196 recursor.sortlists

New in version 5.1.0.

- Sequence of *Sortlist*
- Default: `[]`
- Equivalent Lua config in *Using Sortlist*

Sequence of sort lists.

6.5.197 recursor.spoof_nearmiss_max

Changed in version 4.5.0: Older versions used 20 as the default value.

- Integer
- Default: `1`
- Old style setting: *spoof-nearmiss-max*

If set to non-zero, PowerDNS will assume it is being spoofed after seeing this many answers with the wrong id.

6.5.198 recursor.stack_cache_size

New in version 4.9.0.

- Integer
- Default: `100`
- Old style setting: *stack-cache-size*

Maximum number of mthread stacks that can be cached for later reuse, per thread. Caching these stacks reduces the CPU load at the cost of a slightly higher memory usage, each cached stack consuming *stack-size* bytes of memory. It makes no sense to cache more stacks than the value of *max-mthreads*, since there will never be more stacks than that in use at a given time.

6.5.199 recursor.stack_size

- Integer
- Default: `200000`
- Old style setting: *stack-size*

Size in bytes of the stack of each mthread.

6.5.200 recursor.stats_api_disabled_list

New in version 4.5.0.

- Sequence of strings
- Default: cache-bytes, packetcache-bytes, special-memory-usage, ecs-v4-response-bits-*, ecs-v6-response-bits-*
- Old style setting: *stats-api-disabled-list*

A sequence of statistic names, that are disabled when retrieving the complete list of statistics via the API for performance reasons. These statistics can still be retrieved individually by specifically asking for it.

6.5.201 recursor.stats_carbon_disabled_list

New in version 4.5.0.

- Sequence of strings
- Default: cache-bytes, packetcache-bytes, special-memory-usage, ecs-v4-response-bits-*, ecs-v6-response-bits-*, cumul-answers-*, cumul-auth4answers-*, cumul-auth6answers-*
- Old style setting: *stats-carbon-disabled-list*

A sequence of statistic names, that are prevented from being exported via carbon for performance reasons.

6.5.202 recursor.stats_rec_control_disabled_list

New in version 4.5.0.

- Sequence of strings
- Default: cache-bytes, packetcache-bytes, special-memory-usage, ecs-v4-response-bits-*, ecs-v6-response-bits-*, cumul-answers-*, cumul-auth4answers-*, cumul-auth6answers-*
- Old style setting: *stats-rec-control-disabled-list*

A sequence of statistic names, that are disabled when retrieving the complete list of statistics via *rec_control get-all*, for performance reasons. These statistics can still be retrieved individually.

6.5.203 recursor.stats_ringbuffer_entries

- Integer
- Default: `10000`
- Old style setting: *stats-ringbuffer-entries*

Number of entries in the remotes ringbuffer, which keeps statistics on who is querying your server. Can be read out using `rec_control top-remotes`.

6.5.204 recursor.stats_snmp_disabled_list

New in version 4.5.0.

- Sequence of strings
- Default: cache-bytes, packetcache-bytes, special-memory-usage, ecs-v4-response-bits-*, ecs-v6-response-bits-*
- Old style setting: *stats-snmp-disabled-list*

A sequence of statistic names, that are prevented from being exported via SNMP, for performance reasons.

6.5.205 recursor.system_resolver_interval

New in version 5.1.0.

- Integer
- Default: `0`
- Old style setting: *system-resolver-interval*

Sets the check interval (in seconds) of the system resolver feature. All names known by the system resolver subsystem are periodically checked for changing values.

If the TTL of a name has expired, it is checked by re-resolving it. if a change is detected, the recursor performs an equivalent of `rec_control reload-zones`.

This settings sets the interval between the checks. If set to zero (the default), the value *recursor.system_resolver_ttl* is used.

6.5.206 recursor.system_resolver_self_resolve_check

New in version 5.1.0.

- Boolean
- Default: `true`
- Old style setting: *system-resolver-self-resolve-check*

Warn on potential self-resolve. If this check draws the wrong conclusion, you can disable it.

6.5.207 recursor.system_resolver_ttl

New in version 5.1.0.

- Integer
- Default: `0`
- Old style setting: *system-resolver-ttl*

Sets TTL in seconds of the system resolver feature. If not equal to zero names can be used for forwarding targets. The names will be resolved by the system resolver configured in the OS.

The TTL is used as a time to live to see if the names used in forwarding resolve to a different address than before. If the TTL is expired, a re-resolve will be done by the next iteration of the check function; if a change is detected, the recursor performs an equivalent of `rec_control reload-zones`.

Make sure the recursor itself is not used by the system resolver! Default is 0 (not enabled). A suggested value is 60.

6.5.208 recursor.tcp_threads

New in version 5.0.0.

- Integer
- Default: `1`
- Old style setting: *tcp-threads*

Spawn this number of TCP processing threads on startup.

6.5.209 recursor.threads

- Integer
- Default: `2`
- Old style setting: *threads*

Spawn this number of threads on startup.

6.5.210 recursor.version_string

- String
- Default: `*runtime determined*`
- Old style setting: *version-string*

By default, PowerDNS replies to the 'version.bind' query with its version number. Security conscious users may wish to override the reply PowerDNS issues.

6.5.211 recursor.write_pid

- Boolean
- Default: `true`
- Old style setting: *write-pid*

If a PID file should be written to *recursor.socket_dir*

6.5.212 snmp.agent

New in version 4.1.0.

- Boolean
- Default: `false`
- Old style setting: *snmp-agent*

If set to true and PowerDNS has been compiled with SNMP support, it will register as an SNMP agent to provide statistics and be able to send traps.

6.5.213 snmp.daemon_socket

New in version 4.5.0.

- String
- Default: (empty)
- Old style setting: *snmp-daemon-socket*

If not empty and `snmp-agent` is set to true, indicates how PowerDNS should contact the SNMP daemon to register as an SNMP agent.

6.5.214 webservice.address

- String
- Default: `127.0.0.1`
- Old style setting: *webserver-address*

IP address for the webserver to listen on.

6.5.215 webservice.allow_from

Changed in version 4.1.0: Default is now 127.0.0.1,::1, was 0.0.0.0/0,::/0 before.

- Sequence of *Subnet* (IP addresses or subnets, negation supported)
- Default: `[127.0.0.1, '::1']`
- Old style setting: *webserver-allow-from*

These IPs and subnets are allowed to access the webserver. Note that specifying an IP address without a netmask uses an implicit netmask of /32 or /128.

6.5.216 webservice.api_dir

New in version 4.0.0.

- String
- Default: (empty)
- Old style setting: *api-config-dir*

Directory where the REST API stores its configuration and zones. For configuration updates to work, *recursor.include_dir* should have the same value when using old-style settings. When using YAML settings *recursor.include_dir* and *webservice.api_dir* must have a different value.

6.5.217 webservice.api_key

New in version 4.0.0.

Changed in version 4.6.0: This setting now accepts a hashed and salted version.

- String
- Default: (empty)
- Old style setting: *api-key*

Static pre-shared authentication key for access to the REST API. Since 4.6.0 the key can be hashed and salted using `rec_control hash-password` instead of being stored in the configuration in plaintext, but the plaintext version is still supported.

6.5.218 webservice.hash_plaintext_credentials

New in version 4.6.0.

- Boolean
- Default: `false`
- Old style setting: *webserver-hash-plaintext-credentials*

Whether passwords and API keys supplied in the configuration as plaintext should be hashed during startup, to prevent the plaintext versions from staying in memory. Doing so increases significantly the cost of verifying credentials and is thus disabled by default. Note that this option only applies to credentials stored in the configuration as plaintext, but hashed credentials are supported without enabling this option.

6.5.219 webservice.loglevel

New in version 4.2.0.

- String
- Default: `normal`
- Old style setting: *webserver-loglevel*

One of `none`, `normal`, `detailed`. The amount of logging the webserver must do. 'none' means no useful webserver information will be logged. When set to 'normal', the webserver will log a line per request that should be familiar:

```
[webserver] e235780e-a5cf-415e-9326-9d33383e739e 127.0.0.1:55376 'GET /api/v1/
↪servers/localhost/bla HTTP/1.1' 404 196
```

When set to 'detailed', all information about the request and response are logged:

```
[webserver] e235780e-a5cf-415e-9326-9d33383e739e Request Details:
[webserver] e235780e-a5cf-415e-9326-9d33383e739e  Headers:
[webserver] e235780e-a5cf-415e-9326-9d33383e739e   accept: text/html,application/
↪xhtml+xml,application/xml;q=0.9,*/*;q=0.8
[webserver] e235780e-a5cf-415e-9326-9d33383e739e   accept-encoding: gzip, deflate
[webserver] e235780e-a5cf-415e-9326-9d33383e739e   accept-language: en-US,en;q=0.5
[webserver] e235780e-a5cf-415e-9326-9d33383e739e   connection: keep-alive
[webserver] e235780e-a5cf-415e-9326-9d33383e739e   dnt: 1
[webserver] e235780e-a5cf-415e-9326-9d33383e739e   host: 127.0.0.1:8081
[webserver] e235780e-a5cf-415e-9326-9d33383e739e   upgrade-insecure-requests: 1
[webserver] e235780e-a5cf-415e-9326-9d33383e739e   user-agent: Mozilla/5.0 (X11;␣
↪Linux x86_64; rv:64.0) Gecko/20100101 Firefox/64.0
[webserver] e235780e-a5cf-415e-9326-9d33383e739e  No body
[webserver] e235780e-a5cf-415e-9326-9d33383e739e Response details:
[webserver] e235780e-a5cf-415e-9326-9d33383e739e  Headers:
[webserver] e235780e-a5cf-415e-9326-9d33383e739e   Connection: close
[webserver] e235780e-a5cf-415e-9326-9d33383e739e   Content-Length: 49
[webserver] e235780e-a5cf-415e-9326-9d33383e739e   Content-Type: text/html;␣
↪charset=utf-8
[webserver] e235780e-a5cf-415e-9326-9d33383e739e   Server: PowerDNS/0.0.15896.0.
↪gaba8bab3ab
[webserver] e235780e-a5cf-415e-9326-9d33383e739e  Full body:
[webserver] e235780e-a5cf-415e-9326-9d33383e739e   <!html><title>Not Found</title>
↪<h1>Not Found</h1>
[webserver] e235780e-a5cf-415e-9326-9d33383e739e 127.0.0.1:55376 'GET /api/v1/
↪servers/localhost/bla HTTP/1.1' 404 196
```

The value between the hooks is a UUID that is generated for each request. This can be used to find all lines related to a single request.

Note: The webserver logs these line on the NOTICE level. The *logging.loglevel* seting must be 5 or higher for these lines to end up in the log.

6.5.220 webservice.password

Changed in version 4.6.0: This setting now accepts a hashed and salted version.

- String
- Default: (empty)
- Old style setting: *webserver-password*

Password required to access the webserver. Since 4.6.0 the password can be hashed and salted using `rec_control hash-password` instead of being present in the configuration in plaintext, but the plaintext version is still supported.

6.5.221 `webservice.port`

- Integer
- Default: `8082`
- Old style setting: *webserver-port*

TCP port where the webserver should listen on.

6.5.222 `webservice.webserver`

- Boolean
- Default: `false`
- Old style setting: *webserver*

Start the webserver (for REST API).

CHAPTER

SEVEN

ADVANCED CONFIGURATION USING LUA

Since version 4.0.0, the PowerDNS Recursor supports additional configuration options that have to be loaded through *lua-config-file*.

7.1 Managing DNSSEC Trust Anchors in the Lua Configuration

The DNSSEC Trust Anchors and Negative Trust Anchors must be stored in the Lua Configuration file. See the *DNSSEC in the PowerDNS Recursor* for all information about DNSSEC in the PowerDNS Recursor. This page only documents the Lua functions for DNSSEC configuration

addTA(*name*, *dscontent*)

New in version 4.2.0.

New in version 5.1.0: Alternative equivalent YAML setting: *dnssec.trustanchors*.

Adds Trust Anchor to the list of DNSSEC anchors.

Parameters

- **name** (*str*) – The name in the DNS tree from where this Trust Anchor should be used
- **dsrecord** (*str*) – The DS Record content associated with `name`

clearTA([*name*])

New in version 4.2.0.

Remove Trust Anchors for a name from the list of configured trust anchors. When `name` is not given, remove *all* trust anchors instead.

Parameters **name** (*str*) – The name in the DNS tree for which the Trust Anchors should be removed.

addDS(*name*, *dscontent*)

Deprecated since version 4.2.0: Please use `addTA()` instead

Adds a DS record (Trust Anchor) to the configuration

Parameters

- **name** (*str*) – The name in the DNS tree from where this Trust Anchor should be used
- **dsrecord** (*str*) – The DS Record content associated with `name`

clearDS([*name*])

Deprecated since version 4.2.0: Please use `clearTA()` instead

Remove Trust Anchors for a name from the list of configured trust anchors. When `name` is not given, remove *all* trust anchors instead.

Parameters **name** (*str*) – The name in the DNS tree for which the Trust Anchors should be removed.

addNTA(*name*[, *reason*])

New in version 5.1.0: Alternative equivalent YAML setting: *dnssec.negative_trustanchors*.

Adds a Negative Trust Anchor for `name` to the configuration. Please read *Negative Trust Anchors* for operational information on NTAs.

Parameters

- **name** (*str*) – The name in the DNS tree from where this NTA should be used
- **reason** (*str*) – An optional comment to add to this NTA

clearNTA([*name*])

Remove Negative Trust Anchor for `name` from the list of configured trust anchors. When `name` is not given, remove *all* negative trust anchors instead.

Parameters **name** (*str*) – The name in the DNS tree from where this NTA should be removed

readTrustAnchorsFromFile(*fname*[, *interval*])

New in version 4.2.0.

New in version 5.1.0: Alternative equivalent YAML setting: *dnssec.trustanchorfile* and *dnssec.trustanchorfile_interval*.

Reads all DS and DNSKEY records from `fname` (a BIND zone file) and adds these to the Trust Anchors. This function can be used to read distribution provided trust anchors, as for instance `/usr/share/dns/root.key` from Debian's `dns-root-data` package.

Parameters

- **fname** (*str*) – Path to a zone file with Trust Anchors
- **interval** (*int*) – Re-read this file every `interval` hours. By default this is set to 24. Set to 0 to disable automatic re-reads.

7.2 Logging DNS messages with Protocol Buffers

The PowerDNS Recursor has the ability to emit a stream of protocol buffers messages over TCP, containing information about queries, answers and policy decisions.

Messages contain the IP address of the client initiating the query, the one on which the message was received, whether it was received over UDP or TCP, a timestamp and the qname, qtype and qclass of the question. In addition, messages related to responses contain the name, type, class and rdata of A, AAAA and CNAME records present in the response, as well as the response code.

Finally, if a RPZ or custom Lua policy has been applied, response messages also contain the applied policy name and some tags. This is particularly useful to detect and act on infected hosts.

7.2.1 Configuring Protocol Buffer logs

Protobuf export to a server is enabled using the `protobufServer()` directive:

protobufServer(*servers [, options]*))

New in version 4.2.0.

New in version 5.1.0: Alternative equivalent YAML setting: *logging.protobuf_servers*.

Send protocol buffer messages to one or more servers for incoming queries and/or outgoing responses. The client address may be masked using `setProtobufMasks()`, for anonymization purposes.

Parameters

- **servers** (*string or list of strings*) – The IP and port to connect to, or a list of those. If more than one server is configured, all messages are sent to every server.
- **options** (*table*) – A table with `key=value` pairs with options.

Options:

- `timeout=2`: int - Time in seconds to wait when sending a message
- `maxQueuedEntries=100`: int - How many entries will be kept in memory if the server becomes unreachable
- `reconnectWaitTime=1`: int - How long to wait, in seconds, between two reconnection attempts
- `taggedOnly=false`: bool - Only entries with a policy or a policy tag set will be sent
- `asyncConnect`: bool - When set to false (default) the first connection to the server during startup will block up to `timeout` seconds, otherwise the connection is done in a separate thread, after the first message has been queued
- `logQueries=true`: bool - Whether to export queries
- `logResponses=true`: bool - Whether to export responses
- `exportTypes={'A', 'AAAA', 'CNAME'}`: list of strings - The list of record types found in the answer section to export. Record types A, AAAA, CNAME, MX, NS, PTR, SPF, SRV and TXT are supported.

Changed in version 4.7.0.

The values in `exportTypes` can be numeric as well as strings. Symbolic names from `pdns` can be used, e.g. `exportTypes = { pdns.A, pdns.AAAA, pdns.CNAME }`

New in version 4.7.0.

- `logMappedFrom=false`: bool - whether to log the remote address before substitution by *Table Based Proxy Mapping* (the default) or after

Changed in version 5.1.0: Added support for the HTTPS, SVCB and NAPTR record types.

protobufServer(*server*[[[[[[[, *timeout=2*], *maxQueuedEntries=100*], *reconnectWaitTime=1*], *maskV4=32*], *maskV6=128*], *asyncConnect=false*], *taggedOnly=false*])

Deprecated since version 4.2.0.

Parameters

- **server** (*string*) – The IP and port to connect to
- **timeout** (*int*) – Time in seconds to wait when sending a message
- **maxQueuedEntries** (*int*) – How many entries will be kept in memory if the server becomes unreachable
- **reconnectWaitTime** (*int*) – How long to wait, in seconds, between two reconnection attempts
- **maskV4** (*int*) – network mask to apply to the client IPv4 addresses, for anonymization purposes. The default of 32 means no anonymization.
- **maskV6** (*int*) – Same as maskV4, but for IPv6. Defaults to 128.
- **taggedOnly** (*bool*) – Only entries with a policy or a policy tag set will be sent.
- **asyncConnect** (*bool*) – When set to false (default) the first connection to the server during startup will block up to `timeout` seconds, otherwise the connection is done in a separate thread, after the first message has been queued..

setProtobufMasks(*maskv4*, *maskV6*)

New in version 4.2.0.

New in version 5.1.0: Alternative equivalent YAML setting: *logging.protobuf_mask_v4* and *logging.protobuf_mask_v6*.

Parameters

- **maskV4** (*int*) – network mask to apply to the client IPv4 addresses, for anonymization purposes. The default of 32 means no anonymization.

- **maskV6** (*int*) – Same as maskV4, but for IPv6. Defaults to 128.

7.2.2 Logging outgoing queries and responses

While `protobufServer()` only exports the queries sent to the recursor from clients, with the corresponding responses, `outgoingProtobufServer()` can be used to export outgoing queries sent by the recursor to authoritative servers, along with the corresponding responses.

outgoingProtobufServer(*servers*[, *options*])

New in version 4.2.0.

New in version 5.1.0: Alternative equivalent YAML setting: *logging.outgoing_protobuf_servers*.

Send protocol buffer messages to one or more servers for outgoing queries and/or incoming responses.

Parameters

- **servers** (*string or list of strings*) – The IP and port to connect to, or a list of those. If more than one server is configured, all messages are sent to every server.
- **options** (*table*) – A table with `key=value` pairs with options.

Options:

- `timeout=2`: int - Time in seconds to wait when sending a message
- `maxQueuedEntries=100`: int - How many entries will be kept in memory if the server becomes unreachable
- `reconnectWaitTime=1`: int - How long to wait, in seconds, between two reconnection attempts
- `taggedOnly=false`: bool - Only entries with a policy or a policy tag set will be sent
- `asyncConnect`: bool - When set to false (default) the first connection to the server during startup will block up to `timeout` seconds, otherwise the connection is done in a separate thread, after the first message has been queued
- `logQueries=true`: bool - Whether to export queries
- `logResponses=true`: bool - Whether to export responses
- `exportTypes={'A', 'AAAA', 'CNAME'}`: list of strings or qtypes - The list of record types found in the answer section to export. Record types A, AAAA, CNAME, MX, NS, PTR, SPF, SRV and TXT are supported

Changed in version 4.7.0.

The values in `exportTypes` can be numeric as well as strings. Symbolic names from `pdns` can be used, e.g. `exportTypes = { pdns.A, pdns.AAAA, pdns.CNAME }`

Changed in version 5.1.0: Added support for the HTTPS, SVCB and NAPTR records types.

outgoingProtobufServer(*server*[[[[, *timeout=2*], *maxQueuedEntries=100*], *reconnectWaitTime=1*], *asyncConnect=false*])

Deprecated since version 4.2.0.

Parameters

- **server** (*string*) – The IP and port to connect to
- **timeout** (*int*) – Time in seconds to wait when sending a message
- **maxQueuedEntries** (*int*) – How many entries will be kept in memory if the server becomes unreachable
- **reconnectWaitTime** (*int*) – How long to wait, in seconds, between two reconnection attempts

- **asyncConnect** (*bool*) – When set to false (default) the first connection to the server during startup will block up to `timeout` seconds, otherwise the connection is done in a separate thread, after the first message has been queued..

7.2.3 Protocol Buffers Definition

The protocol buffers message types can be found in the dnsmessage.proto file and is included here:

```
/*
 * This file describes the message format used by the protobuf logging feature in␣
↪PowerDNS and dnsdist.
 *
 * MIT License
 *
 * Copyright (c) 2016-now PowerDNS.COM B.V. and its contributors.
 *
 * Permission is hereby granted, free of charge, to any person obtaining a copy
 * of this software and associated documentation files (the "Software"), to deal
 * in the Software without restriction, including without limitation the rights
 * to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
 * copies of the Software, and to permit persons to whom the Software is
 * furnished to do so, subject to the following conditions:
 *
 * The above copyright notice and this permission notice shall be included in all
 * copies or substantial portions of the Software.
 *
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
 * IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
 * FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
 * AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
 * LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
 * OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
 * SOFTWARE.
 */
syntax = "proto2";

message PBDNSMessage {
  enum Type {
    DNSQueryType = 1;                           // Query received by the service
    DNSResponseType = 2;                        // Response returned by the service
    DNSOutgoingQueryType = 3;                   // Query sent out by the service␣
↪to a remote server
    DNSIncomingResponseType = 4;                // Response returned by the remote␣
↪server
  }
  enum SocketFamily {
    INET = 1;                                   // IPv4 (RFC 791)
    INET6 = 2;                                  // IPv6 (RFC 2460)
  }
  enum SocketProtocol {
    UDP = 1;                                    // User Datagram Protocol (RFC 768)
    TCP = 2;                                    // Transmission Control Protocol␣
↪(RFC 793)
    DOT = 3;                                    // DNS over TLS (RFC 7858)
    DOH = 4;                                    // DNS over HTTPS (RFC 8484)
    DNSCryptUDP = 5;                            // DNSCrypt over UDP (https://
↪dnscrypt.info/protocol)
    DNSCryptTCP = 6;                            // DNSCrypt over TCP (https://
↪dnscrypt.info/protocol)
    DOQ = 7;                                    // DNS over QUIC (RFC 9250)
  }
```

(continues on next page)

(continued from previous page)

```
  enum HTTPVersion {
    HTTP1 = 1;                                  // HTTP/1.1
    HTTP2 = 2;                                  // HTTP/2
    HTTP3 = 3;                                  // HTTP/3
  }
  enum PolicyType {
    UNKNOWN = 1;                                // No RPZ policy applied, or␣
↪unknown type
    QNAME = 2;                                  // Policy matched on the QName
    CLIENTIP = 3;                               // Policy matched on the client IP
    RESPONSEIP = 4;                             // Policy matched on one of the␣
↪IPs contained in the answer
    NSDNAME = 5;                                // Policy matched on the name of␣
↪one nameserver involved
    NSIP = 6;                                   // Policy matched on the IP of one␣
↪nameserver involved
  }
  enum PolicyKind {
    NoAction = 1;                               // No action taken
    Drop = 2;                                   // https://tools.ietf.org/html/
↪draft-vixie-dns-rpz-04 3.4
    NXDOMAIN = 3;                               // https://tools.ietf.org/html/
↪draft-vixie-dns-rpz-04 3.1
    NODATA = 4;                                 // https://tools.ietf.org/html/
↪draft-vixie-dns-rpz-04 3.2
    Truncate= 5;                                // https://tools.ietf.org/html/
↪draft-vixie-dns-rpz-04 3.5
    Custom = 6;                                 // https://tools.ietf.org/html/
↪draft-vixie-dns-rpz-04 3.6
  }
  enum VState {
    Indeterminate = 1;
    Insecure = 2;
    Secure = 3;
    BogusNoValidDNSKEY = 4;
    BogusInvalidDenial = 5;
    BogusUnableToGetDSs = 6;
    BogusUnableToGetDNSKEYs = 7;
    BogusSelfSignedDS = 8;
    BogusNoRRSIG = 9;
    BogusNoValidRRSIG = 10;
    BogusMissingNegativeIndication = 11;
    BogusSignatureNotYetValid = 12;
    BogusSignatureExpired = 13;
    BogusUnsupportedDNSKEYAlgo = 14;
    BogusUnsupportedDSDigestType = 15;
    BogusNoZoneKeyBitSet = 16;
    BogusRevokedDNSKEY = 17;
    BogusInvalidDNSKEYProtocol = 18;
  }
  required Type type = 1;                       // Type of event
  optional bytes messageId = 2;                 // UUID, shared by the query and␣
↪the response
  optional bytes serverIdentity = 3;            // ID of the server emitting the␣
↪protobuf message
  optional SocketFamily socketFamily = 4;
  optional SocketProtocol socketProtocol = 5;
  optional bytes from = 6;                      // DNS requestor (client) as 4␣
↪(IPv4) or 16 (IPv6) raw bytes in network byte order
  optional bytes to = 7;                        // DNS responder (server) as 4␣
↪(IPv4) or 16 (IPv6) raw bytes in network byte order
```

(continues on next page)

(continued from previous page)

```
  optional uint64 inBytes = 8;                  // Size of the query or response␣
↪on the wire
  optional uint32 timeSec = 9;                  // Time of message reception␣
↪(seconds since epoch)
  optional uint32 timeUsec = 10;                // Time of message reception␣
↪(additional micro-seconds)
  optional uint32 id = 11;                      // ID of the query/response as␣
↪found in the DNS header

  message DNSQuestion {
    optional string qName = 1;                  // Fully qualified DNS name (with␣
↪trailing dot)
    optional uint32 qType = 2;                  // https://www.iana.org/
↪assignments/dns-parameters/dns-parameters.xhtml#dns-parameters-4
    optional uint32 qClass = 3;                 // Typically 1 (IN), see https://
↪www.iana.org/assignments/dns-parameters/dns-parameters.xhtml#dns-parameters-2
  }
  optional DNSQuestion question = 12;           // DNS query received from client

  message DNSResponse {
    // See exportTypes in https://docs.powerdns.com/recursor/lua-config/protobuf.
↪html#protobufServer
    // for the list of supported resource record types.
    message DNSRR {
      optional string name = 1;                 // Fully qualified DNS name (with␣
↪trailing dot)
      optional uint32 type = 2;                 // https://www.iana.org/
↪assignments/dns-parameters/dns-parameters.xhtml#dns-parameters-4
      optional uint32 class = 3;                // Typically 1 (IN), see https://
↪www.iana.org/assignments/dns-parameters/dns-parameters.xhtml#dns-parameters-2
      optional uint32 ttl = 4;                  // TTL in seconds
      optional bytes rdata = 5;                 // raw address bytes in network␣
↪byte order for A & AAAA; text representation for others, with fully qualified␣
↪(trailing dot) domain names
      optional bool  udr = 6;                   // True if this is the first time␣
↪this RR has been seen for this question
    }
    optional uint32 rcode = 1;                  // DNS Response code, or 65536 for␣
↪a network error including a timeout
    repeated DNSRR rrs = 2;                     // DNS resource records in response
    optional string appliedPolicy = 3;          // Filtering policy (RPZ or Lua)␣
↪applied
    repeated string tags = 4;                   // Additional tags applied
    optional uint32 queryTimeSec = 5;           // Time of the corresponding query␣
↪reception (seconds since epoch)
    optional uint32 queryTimeUsec = 6;          // Time of the corresponding query␣
↪reception (additional micro-seconds)
    optional PolicyType appliedPolicyType = 7;  // Type of the filtering policy␣
↪(RPZ or Lua) applied
    optional string appliedPolicyTrigger = 8;   // The RPZ trigger
    optional string appliedPolicyHit = 9;       // The value (qname or IP) that␣
↪caused the hit
    optional PolicyKind appliedPolicyKind = 10; // The Kind (RPZ action) applied␣
↪by the hit
    optional VState validationState = 11;       // The DNSSEC Validation State
  }

  optional DNSResponse response = 13;
  optional bytes originalRequestorSubnet = 14;  // EDNS Client Subnet value (4 or␣
↪16 raw bytes in network byte order)
  optional string requestorId = 15;             // Username of the requestor
```

(continues on next page)

(continued from previous page)

```
  optional bytes initialRequestId = 16;         // UUID of the incoming query that␣
↪initiated this outgoing query or incoming response
  optional bytes deviceId = 17;                 // Device ID of the requestor␣
↪(could be mac address IP address or e.g. IMEI, format implementation dependent)
  optional bool  newlyObservedDomain = 18;      // True if the domain has not been␣
↪seen before
  optional string deviceName = 19;              // Device name of the requestor
  optional uint32 fromPort = 20;                // Source port of the DNS query␣
↪(client)
  optional uint32 toPort = 21;                  // Destination port of the DNS␣
↪query (server)

  message MetaValue {
    repeated string stringVal = 1;
    repeated int64 intVal = 2;
  }

  message Meta {
    required string key = 1;                    // MUST be unique, so if you have␣
↪multiple values they must be aggregated into on Meta
    required MetaValue value = 2;
  }
  repeated Meta meta = 22;                      // Arbitrary meta-data - to be␣
↪used in future rather than adding new fields all the time

  // The well known EventTrace event numbers
  enum EventType {
                                                // Range 0..99: Generic events
    CustomEvent = 0;                            // A custom event
    ReqRecv = 1;                                // A request was received
    PCacheCheck = 2;                            // A packet cache check was␣
↪initiated or completed; value: bool cacheHit
    AnswerSent = 3;                             // An answer was sent to the client

                                                // Range 100: Recursor events
    SyncRes = 100;                              // Recursor Syncres main function␣
↪has started or completed; value: int rcode
    LuaGetTag = 101;                            // Events below mark start or end␣
↪of Lua hook calls; value: return value of hook
    LuaGetTagFFI = 102;
    LuaIPFilter = 103;
    LuaPreRPZ = 104;
    LuaPreResolve = 105;
    LuaPreOutQuery = 106;
    LuaPostResolve = 107;
    LuaNoData = 108;
    LuaNXDomain = 109;
    LuaPostResolveFFI = 110;
  }

  message Event {
    required int64 ts = 1;                      // Timestamp in ns relative to␣
↪time of creation of event trace data structure
    required EventType event = 2;               // Type of event
    required bool start = 3;                    // true for "start" events, false␣
↪for "completed" events
    optional bool boolVal = 4;                  // Below are optional values␣
↪associated with events
    optional int64 intVal = 5;
    optional string stringVal = 6;
    optional bytes bytesVal = 7;
```

(continues on next page)

(continued from previous page)

```
    optional string custom = 8;                 // The name of the event for␣
↪custom events
  }
  repeated Event trace = 23;

  optional HTTPVersion httpVersion = 24;        // HTTP version used for DNS over␣
↪HTTP
  optional uint64 workerId = 25;                // Thread id
  optional bool packetCacheHit = 26;            // Was it a packet cache hit?
  optional uint32 outgoingQueries = 27;         // Number of outgoing queries used␣
↪to answer the query
}

message PBDNSMessageList {
  repeated PBDNSMessage msg = 1;
}
```

7.2.4 Logging in `dnstap` format using framestreams

Define the following function to enable logging of outgoing queries and/or responses in `dnstap` format. The recursor must have been built with configure `--enable-dnstap` to make this feature available.

`dnstapFrameStreamServer` (*servers*[, *options*])

New in version 4.3.0.

New in version 5.1.0: Alternative equivalent YAML setting: *logging.dnstap_framestream_servers*.

Send dnstap formatted message to one or more framestream servers for outgoing queries and/or incoming responses.

Parameters

- **`servers`** (*`string or list of strings`*) – Either a pathname of a unix domain socket starting with a slash or the IP:port to connect to, or a list of those. If more than one server is configured, all messages are sent to every server.
- **`options`** (*`table`*) – A table with `key=value` pairs with options.

Options:

- `logQueries=true`: bool - log outgoing queries
- `logResponses=true`: bool - log incoming responses

The following options apply to the settings of the framestream library. Refer to the documentation of that library for the default values, exact description and allowable values for these options. For all these options, absence or a zero value has the effect of using the library-provided default value.

- `bufferHint=0`: unsigned
- `flushTimeout=0`: unsigned
- `inputQueueSize=0`: unsigned
- `outputQueueSize=0`: unsigned
- `queueNotifyThreshold=0`: unsigned
- `reopenInterval=0`: unsigned

`dnstapNODFrameStreamServer` (*servers*[, *options*])

New in version 4.8.0.

New in version 5.1.0: Alternative equivalent YAML setting: *logging.dnstap_nod_framestream_servers*.

Send dnstap formatted message for *Newly Observed Domain Tracking* and *Unique Domain Response*. `Message.type` will be set to `CLIENT_QUERY` for NOD and `RESOLVER_RESPONSE` for UDR. The concerned domain name will be attached in the `Message.query_zone` field. UDR notifications will get the reply attached to the `response_message` field.

Parameters

- **servers** (*string or list of strings*) – Either a pathname of a unix domain socket starting with a slash or the IP:port to connect to, or a list of those. If more than one server is configured, all messages are sent to every server.
- **options** (*table*) – A table with `key=value` pairs with options.

Options:

- `logNODs=true`: bool - log NODs
- `logUDRs=false`: bool - log UDRs

The following options apply to the settings of the framestream library. Refer to the documentation of that library for the default values, exact description and allowable values for these options. For all these options, absence or a zero value has the effect of using the library-provided default value.

- `bufferHint=0`: unsigned
- `flushTimeout=0`: unsigned
- `inputQueueSize=0`: unsigned
- `outputQueueSize=0`: unsigned
- `queueNotifyThreshold=0`: unsigned
- `reopenInterval=0`: unsigned

7.3 Response Policy Zones (RPZ)

Response Policy Zone is an open standard developed by Paul Vixie (ISC and Farsight) and Vernon Schryver (Rhyolite), to modify DNS responses based on a policy loaded via a zonefile.

Frequently, Response Policy Zones get to be very large and change quickly, so it is customary to update them over IXFR. It allows the use of third-party feeds, and near real-time policy updates.

7.4 Evaluation order

If multiple RPZs are loaded, they get consulted in the order they were defined in. It is however possible from Lua to make queries skip specific Response Policy Zones.

The evaluation order of RPZ policies is not always straightforward. Before 4.4.0, the recursor first checked whether the source address of the client matched a "Client IP Address" filter in any RPZ zones, then if the qname matched a "QNAME" trigger. It would then start the regular resolution process and check whether any "NSDNAME" or "NSIP" rule was triggered, then after the resolution process was done check whether any of the final records matched a "Response IP Address" rule. It would stop as soon as a match was found and apply the requested decision immediately, unless the decision was a "passthru". In that last case it would resume the normal processing but would only evaluate the rules coming from a policy with a higher order than the one that matched.

Since 4.4.0 the behaviour is a bit different, to better follow the RPZ specifications. The source address of the client is still checked first. Then the normal resolution process starts and the initial qname as well as any CNAME part of the chain starting from the qname is checked against "QNAME" rules. "NSDNAME" and "NSIP" rules are still checked during the remaining part of the process, and "Response IP Address" rules are applied to the final records in the end. This matches the precedence rules from the RPZ specifications that specify that "A policy rule match which occurs at an earlier stage of resolution is preferred to a policy rule match which occurs at a later stage".

For performance and privacy reasons, the order of evaluation does not strictly follow the one mandated by the RPZ specifications. In particular matching on the client IP and qname is done first before any processing, NS IP and NS DNAME matching is done when a nameserver is about to be sent a query, and matching on response records is done then a stage of resolution is done. The RPZ specifications mention that a match on the response record from a higher order RPZ should take precedence on a qname match from a lower one. Doing so would require delaying evaluation of RPZ policies until the whole resolution process has been completed, which would mean that queries might have been sent to a malicious nameserver already, in addition to performance issues.

Note that "RPZ rules do not apply to synthetic data generated by using RPZ rules. For example, if RPZ supplies a CNAME pointing to a walled garden, RPZ policies will not be used while following that CNAME. If RPZ supplies local data giving a particular A record, RPZ policies will not apply to that response IP address", as stated in section 6.1 of the RPZ specifications.

7.5 Configuring RPZ

An RPZ can be loaded from file or transferred from a primary. To load from file, use for example:

```
rpzFile("dblfilename")
```

To transfer from a primary and start IXFR to get updates, use for example:

```
rpzPrimary("192.0.2.4", "policy.rpz")
```

In this example, 'policy.rpz' denotes the name of the zone to query for.

Note: In versions before 4.5.0, `rpzPrimary` is called `rpzMaster`. For backwards compatibility, version 4.5.0 does support `rpzMaster` as a synonym for `rpzPrimary`.

The action to be taken on a match is defined by the zone itself, but in some cases it might be interesting to be able to override it, and always apply the same action regardless of the one specified in the RPZ zone. To load from file and override the default action with a custom CNAME to badserver.example.com., use for example:

```
rpzFile("dblfilename", {defpol=Policy.Custom, defcontent="badserver.example.com"})
```

To instead drop all queries matching a rule, while transferred from a primary.

```
rpzPrimary("192.0.2.4", "policy.rpz", {defpol=Policy.Drop})
```

Note that since 4.2.0, it is possible for the override policy specified via 'defpol' to no longer be applied to local data entries present in the zone by setting the 'defpolOverrideLocalData' parameter to false.

As of version 4.2.0, the first parameter of *`rpzPrimary()`* can be a list of addresses for failover:

```
rpzPrimary({"192.0.2.4","192.0.2.5:5301"}, "policy.rpz", {defpol=Policy.Drop})
```

In the example above, two addresses are specified and will be tried one after another until a response is obtained. The first address uses the default port (53) while the second one uses port 5301. (If no optional port is set, the default port 53 is used)

7.5.1 Extended Errors

DNS messages can include extended error codes and text in the EDNS part of a reply. If set, the Recursor will add the extended error code and text if resolving a name leads to an RPZ hit. This information is then sent to the client, which can inspect the extended information for diagnosis and other purposes. As an example consider

```
rpzPrimary("192.0.2.4","policy.rpz", {extendedErrorCode = 15, extendedErrorExtra =
↪"Blocked by policy"})
```

Resolving a name blocked by this policy will produce `dig` output containing the following line:

; EDE: 15 (Blocked): 42 6c 6f 63 6b 65 64 20 62 79 20 70 6f 6c 69 63 79 ("Blocked by policy")

Check **RFC 8914** for other `extendedErrorCodes`.

7.6 RPZ Configuration Functions

rpzFile (*filename*, *settings*)

New in version 5.1.0: Alternative equivalent YAML setting: *recursor.rpzs*.

Load an RPZ from disk. If multiple files are to be loaded, the zones can be distinguished by setting a `policyName`, see below.

Parameters

- **filename** (*str*) – The filename to load
- **settings** (*{}*) – A table to settings, see below

rpzPrimary (*address*, *name*, *settings*)

Changed in version 4.2.0.

The first parameter can be a list of addresses.

Changed in version 4.5.0.

This function has been renamed from `rpzMaster`.

New in version 5.1.0: Alternative equivalent YAML setting: *recursor.rpzs*.

Load an RPZ from AXFR and keep retrieving with IXFR.

Parameters

- **address** (*str*) – The IP address to transfer the RPZ from. Also accepts a list of addresses since 4.2.0 in which case they will be tried one after another in the submitted order until a response is obtained.
- **name** (*str*) – The name of this RPZ
- **settings** (*{}*) – A table to settings, see below

7.7 RPZ settings

These options can be set in the `settings` of both *rpzPrimary()* and *rpzFile()*.

7.7.1 defcontent

CNAME field to return in case of defpol=Policy.Custom

7.7.2 defpol

Default policy: *Policy.Custom*, *Policy.Drop*, *Policy.NXDOMAIN*, *Policy.NODATA*, *Policy.Truncate*, *Policy.NoAction*.

7.7.3 defpolOverrideLocalData

New in version 4.2.0: Before 4.2.0 local data entries are always overridden by the default policy.

Whether local data entries should be overridden by the default policy. Default is true.

7.7.4 defttl

the TTL of the CNAME field to be synthesized for the default policy. The default is to use the zone's TTL,

7.7.5 extendedErrorCode

New in version 4.5.0.

An extended error code (**RFC 8914**) to set on RPZ hits. See *extended-resolution-errors*.

7.7.6 extendedErrorExtra

New in version 4.5.0.

An extended error extra text (**RFC 8914**) to set on RPZ hits. See *extended-resolution-errors*.

7.7.7 includeSOA

New in version 4.9.0.

Include the RPZ's SOA record to the reply's additional section if modified by a policy hit. Defaults to `false`.

7.7.8 ignoreDuplicates

New in version 5.0.0.

When loading an RPZ, ignore duplicate entries, keeping only the first one present in the zone. Defaults to `false`, duplicate entries will cause failure to load the zone.

7.7.9 maxTTL

The maximum TTL value of the synthesized records, overriding a higher value from `defttl` or the zone. Default is unlimited.

7.7.10 policyName

The name logged as `appliedPolicy` in *protobuf* messages when this policy is applied. Defaults to `rpzFile` for RPZs loaded by `rpzFile()` or the name of the zone for RPZs loaded by `rpzPrimary()`.

7.7.11 tags

New in version 4.4.0.

List of tags as string, that will be added to the policy tags exported over protobuf when a policy of this zone matches.

7.7.12 overridesGettag

New in version 4.4.0.

gettag_ffi can set an answer to a query. By default an RPZ hit overrides this answer, unless this option is set to *false*. The default is *true*.

7.7.13 zoneSizeHint

An indication of the number of expected entries in the zone, speeding up the loading of huge zones by reserving space in advance.

7.8 Extra settings for rpzPrimary

In addition to the settings above the settings for `rpzPrimary()` may contain:

7.8.1 tsigname

The name of the TSIG key to authenticate to the server. When this is set, *tsigalgo* and *tsigsecret* must also be set.

7.8.2 tsigalgo

The name of the TSIG algorithm (like 'hmac-md5') used

7.8.3 tsigsecret

Base64 encoded TSIG secret

7.8.4 refresh

An integer describing the interval between checks for updates. By default, the RPZ zone's default is used. If allowed by *allow-notify-for* and *allow-notify-from*, a `notify` for an RPZ zone will initiate a freshness check.

7.8.5 maxReceivedMBytes

The maximum size in megabytes of an AXFR/IXFR update, to prevent resource exhaustion. The default value of 0 means no restriction.

7.8.6 localAddress

The source IP address to use when transferring the RPZ. When unset, *query-local-address* is used.

7.8.7 axfrTimeout

New in version 4.1.2: Before 4.1.2, the timeout was fixed on 10 seconds.

Changed in version 4.5.12: The same timeout applies to followup IXFR transactions.

Changed in version 4.6.5: The same timeout applies to followup IXFR transactions.

Changed in version 4.7.4: The same timeout applies to followup IXFR transactions.

The timeout in seconds of the total initial AXFR transaction. 20 by default.

7.8.8 dumpFile

New in version 4.2.0.

A path to a file where the recursor will dump the latest version of the RPZ zone after each successful update. This can be used to keep track of changes in the RPZ zone, or to speed up the initial loading of the zone via the *seedFile* parameter. The format of the generated zone file is the same than the one used with `rpzFile()`, and can also be generated via:

> rec_control dump-rpz *zone-name output-file*

7.8.9 seedFile

New in version 4.2.0.

A path to a file containing an existing dump of the RPZ zone. The recursor will try to load the zone from this file on startup, then immediately do an IXFR to retrieve any updates. If the file does not exist or is not valid, the normal process of doing a full AXFR will be used instead. This option allows a faster startup by loading an existing zone from a file instead of retrieving it from the network, then retrieving only the needed updates via IXFR. The format of the zone file is the same than the one used with `rpzFile()`, and can for example be generated via:

> rec_control dump-rpz *zone-name output-file*

It is also possible to use the *dumpFile* parameter in order to dump the latest version of the RPZ zone after each update.

7.9 Policy Actions

If no settings are included, the RPZ is taken literally with no overrides applied. Several Policy Actions exist

7.9.1 Policy.Custom

Will return a NoError, CNAME answer with the value specified with `defcontent`, when looking up the result of this CNAME, RPZ is not taken into account.

7.9.2 Policy.Drop

Will simply cause the query to be dropped.

7.9.3 Policy.NoAction

Will continue normal processing of the query.

7.9.4 Policy.NODATA

Will return a NoError response with no value in the answer section.

7.9.5 Policy.NXDOMAIN

Will return a response with a NXDomain rcode.

7.9.6 Policy.Truncate

will return a NoError, no answer, truncated response over UDP. Normal processing will continue over TCP

7.10 Using Sortlist

Sortlist is a complicated feature which allows for the ordering of A and AAAA records in answers to be modified, optionally dependently on who is asking. Since clients frequently connect to the 'first' IP address they see, this can effectively allow you to make sure that user from, say 10.0.0.0/8 also preferably connect to servers in 10.0.0.0/8.

The syntax consists of a netmask for which this ordering instruction applies, followed by a set of netmask (groups) which describe the desired ordering. So an ordering instruction of "1.0.0.0/8", "2.0.0.0/8" will put anything within 1/8 first, and anything in 2/8 second. Other IP addresses would follow behind the addresses sorted earlier.

If netmasks are grouped, this means these get equal ordering.

7.10.1 `addSortList`

New in version 5.1.0: Alternative equivalent YAML setting: *recursor.sortlists*.

`addSortList()` is used in the *lua-config-file* and is intended to exactly mirror the semantics of the BIND sortlist option, but the syntax is slightly different.

As an example, the following BIND sortlist:

```
{ 17.50.0.0/16; {17.238.240.0/24; 17.138.149.200;
{17.218.242.254; 17.218.252.254;}; 17.38.42.80;
17.208.240.100; }; };
```

Gets transformed into:

```
addSortList("17.50.0.0/16", {"17.238.240.0/24", "17.138.149.200",
{"17.218.242.254", "17.218.252.254"}, "17.38.42.80",
"17.208.240.100" })
```

In other words: each IP address is put within quotes, and are separated by commas instead of semicolons. For the rest everything is identical.

7.11 Zone to Cache

Zone to Cache is a function to load a zone into the Recursor cache periodically, or every time the Lua configuration is loaded, at startup and whenever `rec_control reload-lua-config` is issued. This allows the Recursor to have an always hot cache for these zones. The zone content to cache can be retrieved via zone transfer (AXFR format) or read from a zone file retrieved via http, https or a local file.

7.11.1 Example

To load the root zone from Internic into the recursor once at startup and when the Lua config is reloaded:

```
zoneToCache(".", "url", "https://www.internic.net/domain/root.zone", {␣
↪refreshPeriod = 0 })
```

7.11.2 DNSSEC and ZONEMD validation

Starting with version 4.7.0, the Recursor will do validation of the zone retrieved. Validation consists of two parts: `DNSSEC` and `ZONEMD`. `ZONEMD` is described in **RFC 8976**.

For the `DNSSEC` part, if the global *dnssec* setting is not `off` or `process-no-validate` and the *DS* record from the parent zone or trust anchor indicates the zone is `DNSSEC` signed, the recursor will validate the `DNSKEY` records of the zone. If a `ZONEMD` record is present, it will also validate the `ZONEMD` record. If no `ZONEMD` is present, the `NSEC` or `NSEC3` denial of the `ZONEMD` record will be validated. Note that this is not a full validation of the signatures of all records. The signatures of the remaining records will be verified on-demand once the records are inserted into the cache by the Zone to Cache function.

For the `ZONEMD` part, if the zone has a `ZONEMD` record with a matching serial number, supported digest algorithm and supported scheme, the digest of the zone will be verified.

For both parts failure of validation will prevent the downloaded zone contents from being inserted into the cache. Absence of `DNSSEC` records is not considered a failure if the parent zone or negative trust anchor indicate the zone is `Insecure`. Absence of `ZONEMD` records is not considered a failure unless `DNSSEC` indicates `ZONEMD` records should be present. This behaviour can be tuned with the `zoneToCache` specific *zonemd* and *dnssec* settings described below.

7.11.3 Configuration

zoneToCache (*zone*, *method*, *source*[, *settings*])

New in version 4.6.0.

New in version 5.1.0: Alternative equivalent YAML setting: *recordcache.zonetocaches*.

Load a zone and put it into the Recursor cache periodically.

Parameters

- **zone** (*str*) – The name of the zone to load
- **method** (*str*) – One of `"axfr"`, `"url"` or `"file"`
- **source** (*str*) – A string representing an IP address (when using the `axfr` method), URL (when using the `url` method) or path name (when using the `file` method)
- **settings** (*table*) – A table of settings, see below

7.11.4 Zone to Cache settings

These options can be set in the `settings` of *`zoneToCache()`*.

timeout

The maximum time (in seconds) a retrieval using the `axfr` or `url` method may take. Default is 20 seconds.

tsigname

The name of the TSIG key to authenticate to the server and validate the zone content with when using the `axfr` method. When this is set, *tsigalgo* and *tsigsecret* must also be set.

tsigalgo

The name of the TSIG algorithm (like 'hmac-md5') used.

tsigsecret

Base64 encoded TSIG secret.

refreshPeriod

An integer describing the interval (in seconds) to wait between retrievals. A value of zero means the retrieval is done once at startup and on Lua configuration reload. By default, the refresh value is 86400 (24 hours).

retryOnErrorPeriod

An integer describing the interval (in seconds) to wait before retrying a failed transfer. By default 60 is used.

maxReceivedMBytes

The maximum size in megabytes of an update via the `axfr` or `url` methods, to prevent resource exhaustion. The default value of 0 means no restriction.

localAddress

The source IP address to use when transferring using the `axfr` or `url` methods. For the `axfr` method *query-local-address* is used by default. The default used for `url` method is system dependent.

zonemd

New in version 4.7.0.

A string, possible values: `ignore`: ignore ZONEMD records, `validate`: validate `ZONEMD` if present, `require`: require valid `ZONEMD` record to be present. Default `validate`.

dnssec

New in version 4.7.0.

A string, possible values: `ignore`: do not do `DNSSEC` validation, `validate`: validate `DNSSEC` records as described above but accept an `Insecure` (unsigned) zone, `require`: require `DNSSEC` validation, as described above. Default `validate`.

7.12 Adding Additional Records to Results

Starting with version 4.7.0, the PowerDNS Recursor has the ability to add additional records to query results.

This allows clients to learn useful information without having to do an extra query. Examples of useful information are the related `A` and `AAAA` records to a query for an `MX` record:

```
;; ANSWER SECTION:
example.net.              86362   IN      MX      20 mx2.example.net.
example.net.              86362   IN      MX      10 mx1.example.net.

;; ADDITIONAL SECTION:
mx1.example.net.          86368   IN      A       192.168.1.2
mx2.example.net.          86400   IN      A       192.168.1.3
mx1.example.net.          86372   IN      AAAA    2001:db8::1
mx2.example.net.          86374   IN      AAAA    2001:db8::2
```

The default is that the Recursor never adds additional records to an answer it sends to the client. The default behavior can be changed by using the `addAllowedAdditionalQType()` function in the *lua-config-file*. For each query type allowing additional record processing the Recursor has code to determine the target name to add. The target qtypes to add are configurable as is the mode, specifying how to retrieve the records to add.

An example of a configuration:

```
addAllowedAdditionalQType(pdns.MX, {pdns.A, pdns.AAAA})
addAllowedAdditionalQType(pdns.NAPTR, {pdns.A, pdns.AAAA, pdns.SRV}, {mode=pdns.
↪AdditionalMode.ResolveImmediately})
```

The first line specifies that additional records should be added to the results of `MX` queries using the default mode. The qtype of the records to be added are `A` and `AAAA`. The default mode is `pdns.AdditionalMode.CacheOnlyRequireAuth`; this mode will only look in the record cache.

The second line specifies that three record types should be added to `NAPTR` answers. If needed, the Recursor will do an active resolve to retrieve these records.

Note that with record types such as `NAPTR` which can return records such as `SRV`, which may themselves return additional `A` or `AAAA` records, the above example would not be sufficient to return those additional `A` and/or `AAAA` records. In such a case, you would need to add an additional line to tell the recursor to fetch the additional records for the `SRV` qtype as well. An example configuration for this case is shown below:

```
addAllowedAdditionalQType(pdns.NAPTR, {pdns.A, pdns.AAAA, pdns.SRV}, {mode=pdns.
↪AdditionalMode.ResolveImmediately})
addAllowedAdditionalQType(pdns.SRV, {pdns.A, pdns.AAAA}, {mode=pdns.AdditionalMode.
↪ResolveImmediately})
```

The modes available are:

`pdns.AdditionalMode.Ignore` Do not do any additional processing for this qtype. This is equivalent to not calling `addAllowedAdditionalQType()` for the qtype.

`pdns.AdditionalMode.CacheOnly` Look in the record cache for available records. Allow non-authoritative (learned from additional sections received from authoritative servers) records to be added.

`pdns.AdditionalMode.CacheOnlyRequireAuth` Look in the record cache for available records. Only authoritative records will be added. These are records received from the nameservers for the specific domain.

`pdns.AdditionalMode.ResolveImmediately` Add records from the record cache (including DNSSEC records if relevant). If no record is found in the record cache, actively try to resolve the target name/qtype. This will delay the answer to the client.

`pdns.AdditionalMode.ResolveDeferred` Add records from the record cache (including DNSSEC records if relevant). If no record is found in the record cache and the negative cache also has no entry, schedule a task to resolve the target name/qtype. The next time the query is processed, the cache might hold the relevant information. If a task is pushed, the answer that triggered it will be marked as variable and consequently not stored into the packet cache.

If an additional record is not available at that time the query is stored into the packet cache the answer packet stored in the packet cache will not contain the additional record. Clients repeating the same question will get an answer from the packet cache if the question is still in the packet cache. These answers do not have the additional record, even if the record cache has learned it in the meantime . Clients will only see the additional record once the packet cache entry expires and the record cache is consulted again. The `pdns.AdditionalMode.ResolveImmediately` mode will not have this issue, at the cost of delaying the first query to resolve the additional records needed. The `pdns.AdditionalMode.ResolveDeferred` mode will only store answers in the packet cache if it determines that no deferred tasks are needed, i.e. either a positive or negative answer for potential additional records is available. If the additional records for an answer have low TTLs compared to the records in the answer section, tasks will be pushed often. Until all tasks for the answer have completed the packet cache will not contain the answer, making the packet cache less effective for this specific answer.

7.12.1 Configuring additional record processing

The following function is available to configure additional record processing. Reloading the Lua configuration will replace the current configuration with the new one. Calling `addAllowedAdditionalQType()` multiple times with a specific qtype will replace previous calls with the same qtype.

addAllowedAdditionalQType(*qtype, targets [, options]*)

New in version 4.7.0.

New in version 5.1.0: Alternative equivalent YAML setting: *recursor.allowed_additional_qtypes*.

Allow additional processing for `qtype`.

Parameters

- **qtype** (*int*) – the qtype number to enable additional record processing for. Supported are: `pdns.MX`, `pdns.SRV`, `pdns.SVCB`, `pdns.HTTPS` and `pdns.NAPTR`.
- **targets** (*list of qtype numbers*) – the target qtypes to look for when adding the additionals. For example `{pdns.A, pdns.AAAA}`.
- **options** (*table*) – a table of options. Currently the only option is `mode` having an integer value. For the available modes, see above. If no mode is specified, the default `pdns.AdditionalMode.CacheOnlyRequireAuth` mode is used.

7.13 Table Based Proxy Mapping

Starting with version 4.7.0, the PowerDNS Recursor has the ability to map source IP addresses to alternative addresses, which is for example useful when some clients reach the recursor via a reverse-proxy. The mapped address is used internally for ACL and similar checks. If the *proxy-protocol-from* is also used, the substitution is done on the source address specified in the proxy protocol header.

Depending on context, the incoming address can be

The interface address `I` the source network interface address of the client

The source address `S` the source address as specified in the Proxy protocol

The mapped address `M` the source address mapped by Table Based Proxy Mapping

`S equals I` if no Proxy Protocol is used.

`M equals S` if no Table Based Proxy Mapping is used.

`I` determines if the Proxy Protocol is used (*proxy-protocol-from*).

`S` is passed to Lua functions and RPZ processing

`M` is used for incoming ACL checking (*allow-from*) and to determine the ECS processing (*ecs-add-for*).

An example use:

```
addProxyMapping("127.0.0.0/24", "203.0.113.1")
domains = { "example.com", "example.net" }
addProxyMapping("10.0.0.0/8", "203.0.113.2", domains)
```

The following function is available to configure table based proxy mapping. Reloading the Lua configuration will replace the current configuration with the new one. If the subnets specified in multiple `addProxyMapping()` calls overlap, the most specific one is used. By default, the address *before* mapping `S` is used for internal logging and `Protobuf` messages. See `protobufServer()` on how to tune the source address logged in `Protobuf` messages.

addProxyMapping(*subnet, ip*[, *domains*])

New in version 4.7.0.

New in version 5.1.0: Alternative equivalent YAML setting: *incoming.proxymappings*.

Specify a table based mapping for a subnet.

Parameters

- **subnet** (*string*) – a subnet to match
- **ip** (*string*) – the IP address or IPaddress port combination to match the subnet to.
- **domains** (*array*) – An array of strings used to fill a *DNS Suffix Match Group*.

If the optional `domains` argument is given to this function, only queries for names matching the *DNS Suffix Match Group* will use the value `M` to determine the outgoing ECS; other queries will use the value `S`. The ACL check will be done against the mapped address `M` for all queries, independent of the name queried. If the `domains` argument is absent, no extra condition (apart from matching the subnet) applies to determine the outgoing ECS value.

In addition, *`pdnslog()`* together with `pdns.loglevels` is also supported in the Lua configuration file.

Note: Starting with version 5.1.0, the settings originally specified in a Lua config file can also be put in YAML form. The conversion printed by `rec_control show-yaml` will print these settings if a Lua config file is specified in the config file being converted. You have to choose however: either set Lua settings the old way in the Lua config file, or convert all to YAML. If you are using YAML settings of items originally specified in the Lua config file, do not set *recursor.lua_config_file* anymore. The **Recursor** will check that you do not mix both configuration methods.

CHAPTER

EIGHT

SCRIPTING POWERDNS RECURSOR

In the PowerDNS Recursor, it is possible to modify resolving behaviour using simple scripts written in the Lua programming language.

Lua scripts can be used for load balancing, legal reasons, commercial purposes, to quickly block dangerous domains or override problematic responses.

Because Lua is extremely fast and lightweight, it easily supports hundreds of thousands of queries per second. The Lua language is explained very well in the excellent book Programming in Lua. If you already have programming experience, Learn Lua in 15 Minutes is a great primer.

For extra performance, a Just In Time compiled version of Lua called LuaJIT is supported.

Note: PowerDNS Recursor is capable of handling many queries simultaneously using cooperative user space multi-threading. Blocking functions called from Lua are not cooperative and will monopolize a worker thread while blocked. Avoid blocking calls.

8.1 Configuring Lua scripts

In order to load scripts, the PowerDNS Recursor must have Lua support built in. The packages distributed from the PowerDNS website have this language enabled, other distributions may differ. By default, the Recursor's configure script will attempt to detect if Lua is available.

note: Only one script can be loaded at the same time. If you load a different script, the current one will be replaced (safely)!

If Lua support is available, a script can be configured either via the configuration file, or at runtime via the `rec_control` tool. Scripts can be reloaded or unloaded at runtime with no interruption in operations. If a new script contains syntax errors, the old script remains in force.

On the command line, or in the configuration file, the setting *lua-dns-script* can be used to supply a full path to the Lua script.

At runtime, `rec_control reload-lua-script` can be used to either reload the script from its current location, or, when passed a new filename, load one from a new location. A failure to parse the new script will leave the old script in working order.

Note: It is also possible to precompile scripts using `luac`, and have PowerDNS load the result. This means that switching scripts is faster, and also that you'll be informed about syntax errors at compile time.

Finally, `rec_control unload-lua-script` can be used to remove the currently installed script, and revert to unmodified behaviour.

8.2 The DNSQuestion (dq) object

Apart from the `ipfilter()`-function, all functions work on a `dq` (DNSQuestion) object. This object contains details about the current state of the question. This state can be modified from the various hooks.

The DNSQuestion object contains at least the following fields:

class DNSQuestion

An object that contains everything about the current query. This object has the following attributes:

addPaddingToResponse

New in version 4.5.0.

Whether the response will get EDNS Padding. See *edns-padding-from* and *edns-padding-mode*.

extendedErrorCode

New in version 4.5.0.

The extended error code, if any. See *extended-resolution-errors*.

extendedErrorExtra

New in version 4.5.0.

The extended error extra text, as a string, if any. See *extended-resolution-errors*.

qname

`DNSName` of the name this query is for.

qtype

Type this query is for as an integer, can be compared against `pdns.A`, `pdns.AAAA`.

rcode

current DNS Result Code, which can be overridden, including to several magical values. Before 4.4.0, the rcode can be set to `pdns.DROP` to drop the query, for later versions refer to *Callback Semantics*. Other statuses are normal DNS return codes, like `pdns.NOERROR`, `pdns.NXDOMAIN` etc.

isTcp

Whether the query was received over TCP.

remoteaddr

`ComboAddress` of the requestor. If the proxy protocol is used, this will contain the source address from the proxy protocol header.

localaddr

`ComboAddress` where this query was received on. If the proxy protocol is used, this will contain the destination address from the proxy protocol header.

interface_remoteaddr

Source `ComboAddress` of the packet received by the recursor. If the proxy protocol is not used, the value will match `remoteaddr`.

interface_localaddr

Destination `ComboAddress` of the packet received by the recursor. If the proxy protocol is not used, the value will match `localaddr`.

variable

Boolean which, if set, indicates the recursor should not packet cache this answer. Honored even when returning false from a hook! Important when providing answers that vary over time or based on sender details.

followupFunction

String that signals the nameserver to take one an additional action:

- followCNAMERecords: When adding a CNAME to the answer, this tells the recursor to follow that CNAME. See *CNAME Chain Resolution*
- getFakeAAAARecords: Get a fake AAAA record, see *DNS64*

- getFakePTRRecords: Get a fake PTR record, see *DNS64*
- udpQueryResponse: Do a UDP query and call a handler, see *UDP Query Response*

followupName
see *DNS64*

followupPrefix
see *DNS64*

appliedPolicy
The decision that was made by the policy engine, see *Modifying Policy Decisions*.

appliedPolicy.**policyName**
A string with the name of the policy. Set by *policyName* in the `rpzFile()` and `rpzPrimary()` configuration items. It is advised to overwrite this when modifying the `DNSQuestion.appliedPolicy.policyKind`

appliedPolicy.**policyType**
The type of match for the policy.
- `pdns.policytypes.None` the empty policy type
- `pdns.policytypes.QName` a match on qname
- `pdns.policytypes.ClientIP` a match on client IP
- `pdns.policytypes.ResponseIP` a match on response IP
- `pdns.policytypes.NSDName` a match on the name of a nameserver
- `pdns.policytypes.NSIP` a match on the IP of a nameserver

appliedPolicy.**policyCustom**
The CNAME content for the `pdns.policyactions.Custom` response, a string

appliedPolicy.**policyKind**
The kind of policy response, there are several policy kinds:
- `pdns.policykinds.Custom` will return a NoError, CNAME answer with the value specified in `DNSQuestion.appliedPolicy.policyCustom`
- `pdns.policykinds.Drop` will simply cause the query to be dropped
- `pdns.policykinds.NoAction` will continue normal processing of the query
- `pdns.policykinds.NODATA` will return a NoError response with no value in the answer section
- `pdns.policykinds.NXDOMAIN` will return a response with a NXDomain rcode
- `pdns.policykinds.Truncate` will return a NoError, no answer, truncated response over UDP. Normal processing will continue over TCP

appliedPolicy.**policyTTL**
The TTL in seconds for the `pdns.policyactions.Custom` response

appliedPolicy.**policyTrigger**
The trigger (left-hand) part of the RPZ rule that was matched

appliedPolicy.**policyHit**
The value that was matched. This is a string representing a name or an address.

wantsRPZ
A boolean that indicates the use of the Policy Engine. Can be set to `false` in `prerpz` to disable RPZ for this query.

data
A Lua object reference that is persistent throughout the lifetime of the `DNSQuestion` object for a single query. It can be used to store custom data. Most scripts initialise this to an empty table early on so they can store multiple items.

requestorId
New in version 4.1.0.

A string that will be used to set the `requestorId` field in *protobuf* messages.

deviceId
New in version 4.1.0.

A string that will be used to set the `deviceId` field in *protobuf* messages.

deviceName
New in version 4.3.0.

A string that will be used to set the `deviceName` field in *protobuf* messages.

udpAnswer
Answer to the *udpQuery* when using the `udpQueryResponse` *followupFunction*. Only filled when the call-back function is invoked.

udpQueryDest
Destination IP address to send the UDP packet to when using the `udpQueryResponse` *followupFunction*

udpQuery
The content of the UDP payload when using the `udpQueryResponse` *followupFunction*

udpCallback
The name of the callback function that is called when using the `udpQueryResponse` *followupFunction* when an answer is received.

validationState
New in version 4.1.0.

The result of the DNSSEC validation, accessible from the `postresolve`, `nxdomain` and `nodata` hooks. Possible states are `pdns.validationstates.Indeterminate`, `pdns.validationstates.Bogus`, `pdns.validationstates.Insecure` and `pdns.validationstates.Secure`. The result will always be `pdns.validationstates.Indeterminate` if validation is disabled or was not requested.

detailedValidationState
New in version 4.4.2.

The result of the DNSSEC validation, accessible from the `postresolve`, `nxdomain` and `nodata` hooks. By contrast with *validationState*, there are several Bogus states to be able to better understand the reason for a DNSSEC validation failure.

Possible states are:

- `pdns.validationstates.Indeterminate`
- `pdns.validationstates.BogusNoValidDNSKEY`
- `pdns.validationstates.BogusInvalidDenial`
- `pdns.validationstates.BogusUnableToGetDSs`
- `pdns.validationstates.BogusUnableToGetDNSKEYs`
- `pdns.validationstates.BogusSelfSignedDS`
- `pdns.validationstates.BogusNoRRSIG`
- `pdns.validationstates.BogusNoValidRRSIG`
- `pdns.validationstates.BogusMissingNegativeIndication`
- `pdns.validationstates.BogusSignatureNotYetValid`
- `pdns.validationstates.BogusSignatureExpired`
- `pdns.validationstates.BogusUnsupportedDNSKEYAlgo`
- `pdns.validationstates.BogusUnsupportedDSDigestType`
- `pdns.validationstates.BogusNoZoneKeyBitSet`
- `pdns.validationstates.BogusRevokedDNSKEY`

- `pdns.validationstates.BogusInvalidDNSKEYProtocol`
- `pdns.validationstates.Insecure`
- `pdns.validationstates.Secure`

The result will always be `pdns.validationstates.Indeterminate` is validation is disabled or was not requested. There is a convenience function named `isValidationStateBogus` that accepts such a state and return a boolean indicating whether this state is a Bogus one.

logResponse

New in version 4.2.0.

Whether the response to this query will be exported to a remote protobuf logger, if one has been configured.

tag

The packetcache tag set via *gettag()* or *gettag_ffi()*. Default tag is zero. Internally to the recursor, the tag is interpreted as an unsigned 32-bit integer.

queryTime

New in version 4.8.0.

The time the query was received

queryTime.**tv_sec**

The number of seconds since the Unix epoch.

queryTime.**tv_usec**

The number of microseconds, to be added to the number of seconds in *DNSQuestion.queryTime.tv_sec* to get a high accuracy timestamp.

It also supports the following methods:

:addAnswer(*type*, *content*[, *ttl*, *name*])

Add an answer to the record of `type` with `content`.

Parameters

- **type** (*int*) – The type of record to add, can be `pdns.AAAA` etc.
- **content** (*str*) – The content of the record, will be parsed into wireformat based on the `type`
- **ttl** (*int*) – The TTL in seconds for this record, defaults to 3600
- **name** (DNSName) – The name of this record, defaults to *DNSQuestion.qname*

:addRecord(*type*, *content*, *place*[, *ttl*, *name*])

Add a record of `type` with `content` in section `place`.

Parameters

- **type** (*int*) – The type of record to add, can be `pdns.AAAA` etc.
- **content** (*str*) – The content of the record, will be parsed into wireformat based on the `type`
- **place** (*int*) – The section to place the record, see *DNSRecord.place*
- **ttl** (*int*) – The TTL in seconds for this record, defaults to 3600
- **name** (DNSName) – The name of this record, defaults to *DNSQuestion.qname*

:addPolicyTag(*tag*)

Add policyTag `tag` to the list of policyTags.

Parameters **tag** (*str*) – The tag to add

:getPolicyTags() → {str}

Get the current policy tags as a table of strings.

:setPolicyTags(*tags*)
Set the policy tags to `tags`, overwriting any existing policy tags.

Parameters **tags** (*{str}*) – The policy tags

:discardPolicy(*policyname*)
Skip the filtering policy (for example RPZ) named `policyname` for this query. This is mostly useful in the `prerpz` hook.

Parameters **policyname** (*str*) – The name of the policy to ignore.

:getDH() → DNSHeader
Returns the *DNSHeader* of the query or nil.

:getProxyProtocolValues() → {ProxyProtocolValue}
New in version 4.4.0: Get the Proxy Protocol Type-Length Values if any, as a table of *ProxyProtocolValue* objects.

:getRecords() → {DNSRecord}
Get a table of DNS Records in this DNS Question (or answer by now).

:setRecords(*records*)
After your edits, update the answers of this question

Parameters **records** (*{DNSRecord}*) – The records to put in the packet

:getEDNSFlag(*name*) → bool
Returns true if the EDNS flag with `name` is set in the query.

Parameters **name** (*string*) – Name of the flag.

:getEDNSFlags() → {str}
Returns a list of strings with all the EDNS flag mnemonics in the query.

:getEDNSOption(*num*) → str
Get the EDNS Option with number `num` as a bytestring.

:getEDNSOptions() → {str: str}
Get a map of all EDNS Options

:getEDNSSubnet() → Netmask
Returns the *Netmask* specified in the EDNSSubnet option, or empty if there was none.

8.3 DNSHeader Object

The DNS header as returned by *DNSQuestion:getDH()* represents a header of a DNS message.

class DNSHeader

represents a header of a DNS message.

:getRD() → bool
The value of the Recursion Desired bit.

:getAA() → bool
The value of the Authoritative Answer bit.

:getAD() → bool
The value of the Authenticated Data bit.

:getCD() → bool
The value of the Checking Disabled bit.

:getTC() → bool
The value of the Truncation bit.

:getRCODE() → int
The Response Code of the query

:getOPCODE() → int
The Operation Code of the query

:getID() → int
The ID of the query

8.4 DNSRecord Object

See *DNSRecord*.

8.5 The EDNSOptionView Class

class EDNSOptionView
An object that represents the values of a single EDNS option

:count()
.. versionadded:: 4.2.0
The number of values for this EDNS option.

:getValues()
.. versionadded:: 4.2.0
Return a table of NULL-safe strings values for this EDNS option.

size
The size in bytes of the first value of this EDNS option.

:getContent()
Returns a NULL-safe string object of the first value of this EDNS option.

8.6 The ProxyProtocolValue Class

class ProxyProtocolValue
New in version 4.4.0.

An object that represents the value of a Proxy Protocol Type-Length Value

:getContent() → str
Returns a NULL-safe string object.

:getType() → int
Returns the type of this value.

8.7 DNS names and comparing them

The PowerDNS Recursor uses a native format for the names it handles. This native format is exposed to Lua as well.

8.7.1 The DNSName object

The PowerDNS Recursor's Lua engine has the notion of a `DNSName`, an object that represents a name in the DNS. It is returned by several functions and has several functions to programmatically interact with it. `DNSNames` can

be compared against each other using the `:equal` function or the == operator. As names in the DNS are case-insensitive, `www.powerdns.com` is equal to `Www.PowerDNS.cOM`.

Creating a *DNSName* is done with *newDN()*. The PowerDNS Recursor will complain loudly if the name is invalid (e.g. too long, dot in the wrong place).

A small example of the functionality of a *DNSName* is shown below:

```
myname = newDN("www.example.com")
print(myname:countLabels()) -- prints "3"
print(myname:wirelength()) -- prints "17"
name2 = newDN(myname)
name2:chopoff() -- returns true, as 'www' was stripped
print(name2:countLabels()) -- prints "2"
if myname:isPartOf(name2) then -- prints "it is"
  print('it is')
end
```

Functions and methods of a `DNSName`

newDN(*name*) → DNSName

Returns the *DNSName* object of `name`.

Parameters **name** (*string*) – The name to create a DNSName for

class DNSName

A `DNSName` object represents a name in the DNS. It is returned by several functions and has several functions to programmatically interact with it.

:canonCompare(*name*) → bool

Performs a comparison of DNS names in canonical order. Returns true if the DNSName comes before `name`. See https://tools.ietf.org/html/rfc4034#section-6

Parameters **name** (`DNSName`) – The name to compare to

:makeRelative(*name*) → DNSName

Returns a new DNSName that is relative to `name`

```
name = newDN("bb.a.example.com.")
parent = newDN("example.com.")
rel = name:makeRelative(parent) -- contains DNSName("bb.a.")
```

Parameters **name** (`DNSName`) – The name to compare to

:isPartOf(*name*) → bool

Returns true if the DNSName is part of the DNS tree of `name`.

Parameters **name** (`DNSName`) – The name to check against

:toString() → string

Returns a human-readable form of the DNSName

:toStringNoDot() → string

Returns a human-readable form of the DNSName without the trailing dot

:chopOff() → bool

Removes the left-most label and returns `true`. `false` is returned if no label was removed

:countLabels() → int

Returns the number of DNSLabels in the name

:wireLength() → int

Returns the length in bytes of the DNSName as it would be on the wire.

DNSName::getRawLabels() -> [string]
Returns a table that contains the raw labels of the DNSName

DNSName::countLabels() -> int
Returns the number of labels of the DNSName

DNSName::equal(name) -> bool
Perform a comparison of the DNSName to the given `name`. You can also compare directly two DNSName objects using the == operator

Parameters **name** (*string*) – The name to compare to

8.7.2 DNS Suffix Match Group

The *newDS()* function creates a `DNS Suffix Match Group` that allows fast checking if a *DNSName* is part of a group. This could e.g. be used to answer questions for known malware domains. To check e.g. the `dq.qname` against a list:

```
m = newDS()
m:add({'example.com', 'example.net'})
m:check(dq.qname) -- Would be true is dq.qname is a name in example.com or example.
↪net
```

newDS() → DNSSuffixMatchGroup
Creates a new `DNS Suffix Match Group`.

class DNSSuffixMatchGroup
This class represents a group of DNS names that can be used to quickly compare a single *DNSName* against.

:add(*domain*)
:add(*dnsname*)
:add(*domains*)
Add one or more domains to the `DNS Suffix Match Group`.

Parameters

- **domain** (*str*) – A domain name to add
- **dnsname** (DNSName) – A dnsname to add
- **domains** (*{str}*) – A list of domain names to add

:check(*dnsname*) → bool
Check `dnsname` against the `DNS Suffix Match Group`. Returns `true` if it is matched, `false` otherwise.

Parameters **dnsname** (DNSName) – The dnsname to check

:toString() → str
Returns a string of the set of suffixes matched by the `DNS Suffix Match Group`.

8.8 DNS Record

DNS record objects are returned by *DNSQuestion:getRecords()* and accepted by *DNSQuestion:addAnswer()*, *DNSQuestion:addRecord()* and *DNSQuestion:setRecords()*.

class DNSRecord
Represents a single DNS record. It has these attributes:

name
The name of the record. A *DNSName*.

place
The place where the record is located,

- 0 for the question section
- 1 for the answer section
- 2 for the authority section
- 3 for the additional section

ttl
The TTL of the record

type
The type of the record (as an integer). Can for example be compared to `pdns.A`.

And the following methods:

:changeContent (*newcontent*)
Replace the record content with `newcontent`. The type and class cannot be changed.

Parameters **newcontent** (*str*) – The replacing content

:getCA() → ComboAddress
If the record type is A or AAAA, a *ComboAddress* representing the content is returned, nil otherwise.

:getContent() → str
Return a string representation of the record content.

8.9 The ComboAddress class

IP addresses are moved around in a native format, called ComboAddress within PowerDNS. ComboAddresses can be IPv4 or IPv6, and unless you want to know, you don't need to.

Make a *ComboAddress* with:

```
newCA("::1")
```

A *ComboAddress* can be compared against a NetmaskGroup with the *NetMaskGroup:match()* function.

To compare the address (so not the port) of two ComboAddresses, use `:equal`:

```
a = newCA("[::1]:56")
b = newCA("[::1]:53")
a == b      -- false, port mismatch
a:equal(b)  -- true
```

To convert an address to human-friendly representation, use *:toString* or *:toStringWithPort*. To get only the port number, use *:getPort()*.

newCA (*address*) → ComboAddress
Creates a *ComboAddress*.

Parameters **address** (*string*) – The address to convert

class ComboAddress
An object representing an IP address and port tuple.

:getPort() → int
The portnumber.

:getRaw() → str
A bytestring representing the address.

:isIPv4() → bool
True if the address is an IPv4 address.

:isIPv6() → bool
True if the address is an IPv6 address.

:isMappedIPv4() → bool
True if the address is an IPv4 address mapped into an IPv6 one.

:mapToIPv4() → ComboAddress
If the address is an IPv4 mapped into an IPv6 one, return the corresponding IPv4 `ComboAddress`.

:toString() → str
Returns the IP address without the port number as a string.

:toStringWithPort() → str
Returns the IP address with the port number as a string.

:truncate(*bits*)
Truncate to the supplied number of bits

Parameters **bits** (*int*) – The number of bits to truncate to

8.10 Netmasks and NetMaskGroups

There are two classes in the PowerDNS Recursor that can be used to match IP addresses against.

8.10.1 Netmask class

The `Netmask` class represents an IP netmask.

```
mask = newNetmask("192.0.2.1/24")
mask:isIPv4() -- true
mask:match("192.0.2.8") -- true
```

newNetmask(*mask*) → Netmask
Creates a new `Netmask`.

Parameters **mask** (*str*) – The mask to convert.

class Netmask
Represents a netmask.

:empty() → bool
True if the netmask doesn't contain a valid address.

:getBits() → int
The number of bits in the address.

:getNetwork() → ComboAddress
Returns a `ComboAddress` representing the network (no mask applied).

:getMaskedNetwork() → ComboAddress
Returns a `ComboAddress` representing the network (truncating according to the mask).

:isIpv4() → bool
Deprecated since version 4.3.0: Use `isIPv4()`.

True if the netmask is an IPv4 netmask.

:isIPv4() → bool
New in version 4.3.0.

True if the netmask is an IPv4 netmask.

:isIpv6() → bool
Deprecated since version 4.3.0: Use `isIPv6()`.

True if the netmask is an IPv6 netmask.

:isIPv6() → bool
New in version 4.3.0.

True if the netmask is an IPv6 netmask.

:match(*address*) → bool
True if the address passed in address matches

Parameters **address** (*str*) – IP Address to match against.

:toString() → str
Returns a human-friendly representation.

8.10.2 NetMaskGroup class

NetMaskGroups are more powerful than plain Netmasks. They can be matched against netmasks objects:

```
nmg = newNMG()
nmg:addMask("127.0.0.0/8")
nmg:addMasks({"213.244.168.0/24", "130.161.0.0/16"})
nmg:addMasks(dofile("bad-ips.lua")) -- a lua script file that contains: return {
↪"ip1","ip2"..}

if nmg:match(dq.remoteaddr) then
  print("Intercepting query from ", dq.remoteaddr)
end
```

Prefixing a mask with `!` excludes that mask from matching.

newNMG([*masks*]) → NetMaskGroup
Changed in version 4.6.0: Added the optional `masks` parameter.

Returns a new *`NetMaskGroup`*. If no masks are passed, the object is empty.

Parameters **masks** (*{str}*) – The masks to add.

class NetMaskGroup
IP addresses are passed to Lua in native format.

:addMask(*mask*)
Adds `mask` to the NetMaskGroup.

Parameters **mask** (*str*) – The mask to add.

:addMasks(*masks*)
Adds `masks` to the NetMaskGroup.

Parameters **mask** (*{str}*) – The masks to add.

:match(*address*) → bool
Returns true if `address` matches any of the masks in the group.

Parameters **address** (`ComboAddress`) – The IP address to match the netmasks against.

8.11 Policy Events

Since 4.4.0, the Lua hook *`policyEventFilter()`* is called along with a *`PolicyEvent`* object whenever a filtering policy matches.

8.11.1 PolicyEvent class

class PolicyEvent

Represents an event related to a filtering policy.

:addPolicyTag (*tag*)

Add policyTag `tag` to the list of policyTags.

Parameters **tag** (*str*) – The tag to add

:getPolicyTags () → {str}

Get the current policy tags as a table of strings.

:setPolicyTags (*tags*)

Set the policy tags to `tags`, overwriting any existing policy tags.

Parameters **tags** (*{str}*) – The policy tags

:discardPolicy (*policyname*)

Skip the filtering policy (for example RPZ) named `policyname` for this query.

Parameters **policyname** (*str*) – The name of the policy to ignore.

appliedPolicy

The decision that was made by the policy engine, see *Modifying Policy Decisions* and *DNSQuestion.appliedPolicy* for the attributes of *PolicyEvent.appliedPolicy*.

qname

DNSName of the name the query is for.

qtype

Type the query is for as an integer, can be compared against `pdns.A`, `pdns.AAAA`.

isTcp

Whether the query was received over TCP.

remote

ComboAddress of the requestor.

8.12 Lua Scripting and Statistics

The Lua engine can generate and retrieve metrics.

8.12.1 Generating Metrics

Custom metrics can be added which will be shown in the output of 'rec_control get-all' and sent to the metrics server over the Carbon protocol. They will also appear in the JSON HTTP API.

Create a custom metric with:

```
myMetric=getMetric("myspecialmetric")
```

getMetric (*name*[, *prometheusName*]) → Metric

Returns the *Metric* object with the name `name`, creating the metric if it does not exist.

Parameters **name** (*str*) – The metric to retrieve

New in version 4.5.0.

Parameters **prometheusName** (*string*) – The optional Prometheus specific name.

class Metric

Represents a custom metric

Metric::inc()
Increase metric by 1

Metric::incBy(amount)
Increase metric by amount

Parameters **amount** (*int*) –

Metric::set(to)
Set metric to value `to`

Parameters **to** (*int*) –

Metric::get() -> int
Get value of metric

Metrics are shared across all of PowerDNS and are fully atomic and high performance. A *Metric* object is effectively a pointer to an atomic value.

Note that metrics live in the same namespace as 'system' metrics. So if you generate one that overlaps with a PowerDNS stock metric, you will get double output and weird results.

8.12.2 Looking at Statistics

New in version 4.1.0.

Statistics can be retrieved from Lua using the *getStat()* call.

getStat (*name*) → int
Returns the value of a statistic.

Parameters **name** (*string*) – The name of the statistic.

For example, to retrieve the number of cache misses:

```
cacheMisses = getStat("cache-misses")
```

Please be aware that retrieving statistics is a relatively costly operation, and as such should for example not be done for every query.

8.13 Logging from the Lua scripts

To log messages with the main PowerDNS Recursor process, use *pdnslog()*. *pdnslog()* can also write out to a syslog loglevel if specified. Use `pdnslog(message, pdns.loglevels.LEVEL)` with the correct pdns.loglevels entry. Entries are listed in the following table:

pdnslog (*message*)
pdnslog (*message*, *level*)
Log `message` at the Info level if ``level` is not set.

Parameters

- **msg** (*str*) – The message to log
- **level** (*int*) – The log level to log at, see below.

- All - `pdns.loglevels.All`
- Alert - `pdns.loglevels.Alert`
- Critical - `pdns.loglevels.Critical`
- Error - `pdns.loglevels.Error`
- Warning - `pdns.loglevels.Warning`

- Notice - `pdns.loglevels.Notice`
- Info - `pdns.loglevels.Info`
- Debug - `pdns.loglevels.Debug`
- None - `pdns.loglevels.None`

8.14 Intercepting queries with Lua

To get a quick start, we have supplied a sample script that showcases all functionality described below.

Queries can be intercepted in many places:

- before any packet parsing begins (*`ipfilter()`*)
- before the packet cache has been looked up (*`gettag()`* and its FFI counterpart, *`gettag_ffi()`*)
- before any filtering policy have been applied (*`prerpz()`*)
- before the resolving logic starts to work (*`preresolve()`*)
- after the resolving process failed to find a correct answer for a domain (*`nodata()`*, *`nxdomain()`*)
- after the whole process is done and an answer is ready for the client (*`postresolve()`* and its FFI counterpart, *`postresolve_ffi()`*).
- before an outgoing query is made to an authoritative server (*`preoutquery()`*)
- after a filtering policy hit has occurred (*`policyEventFilter()`*)

8.14.1 Writing Lua PowerDNS Recursor scripts

Addresses and DNS Names are not passed as strings but as native objects. This allows for easy checking against Netmasks and domain sets. It also means that to print such names, the `:toString` method must be used (or even `:toStringWithPort` for addresses).

Once a script is loaded, PowerDNS looks for the interception functions in the loaded script. All of these functions are optional.

If `ipfilter` returns `true`, the query is dropped. If `preresolve` returns `true`, it will indicate it handled a query, and the recursor will send the result as constructed in the function to the client. If it returns `false`, the Recursor will continue processing. For the other functions, the return value will indicate that an alteration to the result has been made. In that case the potentially changed rcode, records and policy will be processed and DNSSEC validation will be automatically disabled since the content might not be genuine anymore. At specific points the Recursor will check if policy handling should take place. These points are immediately after `preresolve`, after resolving and after `nxdomain`, `nodata` and `postresolve`.

8.14.2 Interception Functions

`ipfilter`(*remoteip*, *localip*, *dh*) → bool

This hook gets queried immediately after consulting the packet cache, but before parsing the DNS packet. If this hook returns something else than `false`, the packet is dropped. However, because this check is after the packet cache, the IP address might still receive answers that require no packet parsing.

With this hook, undesired traffic can be dropped rapidly before using precious CPU cycles for parsing. As an example, to filter all queries coming from 1.2.3.0/24, or with the AD bit set:

```
badips = newNMG()
badips:addMask("1.2.3.0/24")

function ipfilter(rem, loc, dh)
```

(continues on next page)

(continued from previous page)

```
    return badips:match(rem) or dh:getAD()
end
```

This hook does not get the full *DNSQuestion* object, since filling out the fields would require packet parsing, which is what we are trying to prevent with this function.

Parameters

- **remoteip** (ComboAddress) – The IP(v6) address of the requestor
- **localip** (ComboAddress) – The address on which the query arrived.
- **dh** (DNSHeader) – The DNS Header of the query.

gettag(*remote*, *ednssubnet*, *localip*, *qname*, *qtype*, *ednsoptions*, *tcp*, *proxyprotocolvalues*) → multiple values
gettag(*remote*, *ednssubnet*, *localip*, *qname*, *qtype*, *ednsoptions*, *tcp*) → int
gettag(*remote*, *ednssubnet*, *localip*, *qname*, *qtype*, *ednsoptions*) → int

Changed in version 4.1.0: The `tcp` parameter was added.

Changed in version 4.4.0: The `proxyprotocolvalues` parameter was added.

The *gettag()* function is invoked when **Recursor** attempts to discover in which packetcache an answer is available.

This function must return an unsigned 32-bit integer, which is the tag number of the packetcache. The tag is used to partition the packet cache. The default tag (when *gettag()* is not defined) is zero. If *gettag()* throws an exception, the zero tag is used.

In addition to the tag, this function can return a table of policy tags and a few more values to be passed to the resolving process. The resulting tag number can be accessed via *dq.tag* in the *preresolve()* hook, and the policy tags via *dq:getPolicyTags()* in every hook.

New in version 4.1.0: It can also return a table whose keys and values are strings to fill the *DNSQuestion.data* table, as well as a `requestorId` value to fill the *DNSQuestion.requestorId* field and a `deviceId` value to fill the *DNSQuestion.deviceId* field.

New in version 4.3.0: Along the `deviceId` value that can be returned, it was added a `deviceName` field to fill the *DNSQuestion.deviceName* field.

New in version 4.4.0: A `routingTag` can be returned, which is used as an extra name to identify records in the record cache. If a routing tag is set and a record would be stored with an ENDS subnetmask in the record cache, it will be stored with the tag instead. New request using the same tag will be served by the record in the records cache, avoiding querying authoritative servers.

The tagged packetcache can e.g. be used to answer queries from cache that have e.g. been filtered for certain IPs (this logic should be implemented in *gettag()*). This ensure that queries are answered quickly compared to setting *dq.variable* to true. In the latter case, repeated queries will not be found in the packetcache and pass through the entire resolving process, and all relevant Lua hooks wil be called.

Parameters

- **remote** (ComboAddress) – The sender's IP address
- **ednssubnet** (Netmask) – The EDNS Client subnet that was extracted from the packet
- **localip** (ComboAddress) – The IP address the query was received on
- **qname** (DNSName) – The domain name the query is for
- **qtype** (*int*) – The query type of the query
- **ednsoptions** – A table whose keys are EDNS option codes and values are *EDNSOptionView* objects. This table is empty unless the *gettag-needs-edns-options* option is set.

- **tcp** (*bool*) – Added in 4.1.0, a boolean indicating whether the query was received over UDP (false) or TCP (true).
- **proxyprotocolvalues** – Added in 4.4.0, a table of `ProxyProtocolValue` objects representing the Type-Length Values received via the Proxy Protocol, if any.

Returns `tag [, policyTags [, data [, reqId [, deviceId [, deviceName [, routingTag ]]]]]]`

gettag_ffi (*param*) → optional Lua object

New in version 4.1.2.

Changed in version 4.3.0: The ability to craft answers was added.

This function is the FFI counterpart of the `gettag()` function, and offers the same functionality. It accepts a single parameter which can be accessed and modified using *FFI accessors*.

Like the non-FFI version, it has the ability to set a tag for the packetcache, policy tags, a routing tag, the `DNSQuestion.requestorId` and `DNSQuestion.deviceId` values and to fill the `DNSQuestion.data` table. It also offers ways to mark the answer as variable so it's not inserted into the packetcache, to set a cap on the TTL of the returned records, and to generate a response by adding records and setting the RCode. It can also instruct the recursor to do a proper resolution in order to follow any *CNAME* records added in this step.

If this function does not set the tag or an exception is thrown, the zero tag is assumed.

prerpz (*dq*) → bool

This hook is called before any filtering policy have been applied, making it possible to completely disable filtering by setting `dq.wantsRPZ` to false. Using the `dq:discardPolicy()` function, it is also possible to selectively disable one or more filtering policy, for example RPZ zones, based on the content of the `dq` object. Currently, the return value of this function is ignored.

As an example, to disable the "malware" policy for example.com queries:

```
function prerpz(dq)
  -- disable the RPZ policy named 'malware' for example.com
  if dq.qname:equal('example.com') then
    dq:discardPolicy('malware')
  end
  return false
end
```

Parameters **dq** (`DNSQuestion`) – The DNS question to handle

preresolve (*dq*) → bool

This function is called before any DNS resolution is attempted, and if this function indicates it, it can supply a direct answer to the DNS query, overriding the internet. This is useful to combat botnets, or to disable domains unacceptable to an organization for whatever reason.

Parameters **dq** (`DNSQuestion`) – The DNS question to handle

postresolve (*dq*) → bool

is called right before returning a response to a client (and, unless `dq.variable` is set, to the packet cache too). It allows inspection and modification of almost any detail in the return packet.

Parameters **dq** (`DNSQuestion`) – The DNS question to handle

postresolve_ffi (*handle*) → bool

New in version 4.7.0.

This is the FFI counterpart of `postresolve()`. It accepts a single parameter which can be passed to the functions listed in *Lua FFI API*. The accessor functions retrieve and modify various aspects of the answer returned to the client.

nxdomain (*dq*) → bool

is called after the DNS resolution process has run its course, but ended in an 'NXDOMAIN' situation,

indicating that the domain does not exist. Works entirely like `postresolve()`, but saves a trip through Lua for answers which are not NXDOMAIN.

Parameters **dq** (`DNSQuestion`) – The DNS question to handle

nodata(*dq*) → bool
is just like `nxdomain()`, except it gets called when a domain exists, but the requested type does not. This is where one would implement *DNS64*.

Parameters **dq** (`DNSQuestion`) – The DNS question to handle

preoutquery(*dq*) → bool
This hook is not called in response to a client packet, but fires when the Recursor wants to talk to an authoritative server.

When this hook sets the special result code `-3`, the whole DNS client query causing this outgoing query gets a `ServFail`.

However, this function can also return records like `preresolve()`.

Parameters **dq** (`DNSQuestion`) – The DNS question to handle.

In the case of `preoutquery()`, only a few attributes if the `dq` object are filled in:

- `dq.remoteaddr` containing the target nameserver address
- `dq.localaddr`
- `dq.qname`
- `dq.qtype`
- `dq.isTcp`

Do not rely on other attributes having a value and do not call any method of the `dq` object apart from the record set manipulation methods.

policyEventFilter(*event*) → bool
New in version 4.4.0.

This hook is called when a filtering policy has been hit, before the decision has been applied, making it possible to change a policy decision by altering its content or to skip it entirely. Using the `event:discardPolicy()` function, it is also possible to selectively disable one or more filtering policy, for example RPZ zones. The return value indicates whether the policy hit should be completely ignored (true) or applied (false), possibly after editing the action to take in that latter case (see *Modifying Policy Decisions* below). when true is returned, the resolution process will resume as if the policy hit never took place.

Parameters **event** (`PolicyEvent`) – The event to handle

As an example, to ignore the result of a policy hit for the example.com domain:

```
function policyEventFilter(event)
  if event.qname:equal("example.com") then
    -- ignore that policy hit
    return true
  end
  return false
end
```

To alter the decision of the policy hit instead:

```
function policyEventFilter(event)
  if event.qname:equal("example.com") then
    -- replace the decision with a custom CNAME
    event.appliedPolicy.policyKind = pdns.policykinds.Custom
    event.appliedPolicy.policyCustom = "example.net"
    -- returning false so that the hit is not ignored
```

(continues on next page)

(continued from previous page)

```
    return false
  end
  return false
end
```

Callback Semantics

The functions which modify or influence the query flow should all return `true` when they have performed an action which alters the rcode, result or applied policy. When a function returns `false`, the nameserver will process the query normally until a new function is called.

`ipfilter()` and *`preresolve()`* callbacks must return `true` if they have taken over the query and wish that the nameserver should not proceed with processing.

If a function has taken over a request, it can set an rcode (usually 0), and specify a table with records to be put in the answer section of a packet. An interesting rcode is *NXDOMAIN* (3, or `pdns.NXDOMAIN`), which specifies the non-existence of a domain. Instead of setting an rcode and records, it can also set fields in the applied policy to influence further processing.

The *`ipfilter()`* and *`preoutquery()`* hooks are different, in that *`ipfilter()`* can only return a true of false value, and that *`preoutquery()`* can also set rcode -3 to signify that the whole query should be terminated.

The *`policyEventFilter()`* has a different meaning as well, where returning true means that the policy hit should be ignored and normal processing should be resumed.

A minimal sample script:

```
function nxdomain(dq)
    print("Intercepting NXDOMAIN for: ",dq.qname:toString())
    if dq.qtype == pdns.A
    then
        dq.rcode=0 -- make it a normal answer
        dq:addAnswer(pdns.A, "192.168.1.1")
        return true
    end
    return false
end
```

Warning: Please do NOT use the above sample script in production! Responsible NXDomain redirection requires more attention to detail.

Useful `rcodes` include 0 or `pdns.NOERROR` for no error and `pdns.NXDOMAIN` for `NXDOMAIN`. Before 4.4.0, `pdns.DROP` can also be used to drop the question without any further processing. Such a drop is accounted in the `policy-drops` metric.

Starting with recursor 4.4.0, the method to drop a request is to set the `dq.appliedPolicy.policyKind` to the value `pdns.policykinds.Drop`.

```
function nxdomain(dq)
    print("Intercepting and dropping NXDOMAIN for: ",dq.qname:toString())
    if dq.qtype == pdns.A
    then
        dq.appliedPolicy.policyKind = pdns.policykinds.Drop
    end
    return false
end
```

Note: to drop a query set `policyKind` and return `false`, to indicate the Recursor should process the `Drop` action.

8.14.3 DNS64

The `getFakeAAAARecords` and `getFakePTRRecords` followupFunctions can be used to implement DNS64. See *DNS64 support* for more information.

To get fake AAAA records for DNS64 usage, set dq.followupFunction to `getFakeAAAARecords`, dq.followupPrefix to e.g. "64:ff9b::" and dq.followupName to the name you want to synthesize an IPv6 address for.

For fake reverse (PTR) records, set dq.followupFunction to `getFakePTRRecords` and set dq.followupName to the name to look up and dq.followupPrefix to the same prefix as used with `getFakeAAAARecords`.

8.14.4 Follow up actions

When modifying queries, it might be needed that the Recursor does some extra work after the function returns. The *`dq.followupFunction`* can be set in this case.

CNAME chain resolution

It may be useful to return a CNAME record for Lua, and then have the PowerDNS Recursor continue resolving that CNAME. This can be achieved by setting dq.followupFunction to `followCNAMERecords` and dq.followupDomain to "www.powerdns.com". PowerDNS will do the rest.

UDP Query Response

The `udpQueryResponse` *`dq.followupFunction`* allows you to query a simple key-value store over UDP asynchronously.

Several dq variables can be set:

- *`dq.udpQueryDest`*: destination IP address to send the UDP packet to
- *`dq.udpQuery`*: The content of the UDP payload
- *`dq.udpCallback`*: The name of the callback function that is called when an answer is received

The callback function must accept the `dq` object and can find the response to the UDP query in *`dq.udpAnswer`*.

In this callback function, *`dq.followupFunction`* can be set again to any of the available functions for further processing.

This example script queries a simple key/value store over UDP to decide on whether or not to filter a query:

```
--[[
This implements a two-step domain filtering solution where the status of an IP␣
↪address
and a domain name need to be looked up.
To do so, we use the udpQuestionResponse answers which generically allows us to do␣
↪asynchronous
lookups via UDP.
Such lookups can be slow, but they won't block PowerDNS while we wait for them.

To benefit from this hook,
..

To test, use the 'kvresp' example program provided.
--]]

function preresolve (dq)
    pdnslog("preresolve handler called for: "..dq.remoteaddr:toString()..", local:
↪"..dq.localaddr:toString()..", "..dq.qname:toString()..", "..dq.qtype)
```

(continues on next page)

(continued from previous page)

```
    dq.followupFunction="udpQueryResponse"
    dq.udpCallback="gotdomaindetails"
    dq.udpQueryDest=newCA("127.0.0.1:5555")
    dq.udpQuery = "DOMAIN "..dq.qname:toString()
    return true;
end

function gotdomaindetails(dq)
    pdnslog("gotdomaindetails called, got: "..dq.udpAnswer)

    if(dq.udpAnswer == "0")
    then
        pdnslog("This domain needs no filtering, not looking up this domain")
        dq.followupFunction=""
        return false
    end
    pdnslog("Domain might need filtering for some users")
    dq.variable = true -- disable packet cache

    local data={}
    data["domaindetails"]= dq.udpAnswer
    dq.data=data
    dq.udpQuery="IP "..dq.remoteaddr:toString()
    dq.udpCallback="gotipdetails"
    pdnslog("returning true in gotipdetails")
    return true
end

function gotipdetails(dq)
    dq.followupFunction=""
    pdnslog("So status of IP is "..dq.udpAnswer.." and status of domain is "..dq.
↪data.domaindetails)

    if(dq.data.domaindetails=="1" and dq.udpAnswer=="1")
    then
        pdnslog("IP wants filtering and domain is of the filtered kind")
        dq:addAnswer(pdns.CNAME, "blocked.powerdns.com")
        return true
    else
        pdnslog("Returning false (normal resolution should proceed, for this user)
↪")
        return false
    end
end
```

8.14.5 Example Script

```
pdnslog("pdns-recursor Lua script starting!", pdns.loglevels.Warning)

blockset = newDS()
blockset:add{"powerdns.org", "xxx"}

dropset = newDS()
dropset:add("123.cn")

malwareset = newDS()
malwareset:add("nl")

magic2 = newDN("www.magic2.com")
```

(continues on next page)

(continued from previous page)

```
magicMetric = getMetric("magic")

badips = newNMG()
badips:addMask("127.1.0.0/16")

-- this check is applied before any packet parsing is done
function ipfilter(rem, loc, dh)
  pdnslog("ipfilter called, rem: "..rem:toStringWithPort().." loc: "..
↪loc:toStringWithPort().." match:"..tostring(badips:match(rem)))
  pdnslog("id: "..dh:getID().." aa: "..tostring(dh:getAA()).." ad: "..
↪tostring(dh:getAD()).." arcount: "..dh:getARCOUNT())
  pdnslog("ports: "..rem:getPort().." "..loc:getPort())
  return badips:match(rem)
end

-- shows the various ways of blocking, dropping, changing questions
-- return false to say you did not take over the question, but we'll still listen␣
↪to 'variable'
-- to selectively disable the cache
function preresolve(dq)
  pdnslog("Got question for "..dq.qname:toString().." from "..dq.
↪remoteaddr:toString().." to "..dq.localaddr:toString())

  local ednssubnet = dq:getEDNSSubnet()
  if ednssubnet then
    pdnslog("Packet EDNS subnet source: "..ednssubnet:toString()..", "..
↪ednssubnet:getNetwork():toString())
  end

  local a = dq:getEDNSOption(3)
  if a then
    pdnslog("There is an EDNS option 3 present: "..a)
  end

  loc = newCA("127.0.0.1")
  if dq.remoteaddr:equal(loc) then
    pdnslog("Query from loopback")
  end

  -- note that the comparisons below are CaSe InSensiTivE and you don't have to␣
↪worry about trailing dots
  if dq.qname:equal("magic.com") then
    magicMetric:inc()
    pdnslog("Magic!")
  else
    pdnslog("not magic..")
  end

  if dq.qname == magic2 then
    pdnslog("Faster magic") -- compares against existing DNSName
  end

  if blockset:check(dq.qname) then
    dq.variable = true      -- disable packet cache in any case
    if dq.qtype == pdns.A then
      dq:addAnswer(pdns.A, "1.2.3.4")
      dq:addAnswer(pdns.TXT, "\"Hello!\"", 3601) -- ttl
      return true
    end
  end
```

(continues on next page)

(continued from previous page)

```
  if dropset:check(dq.qname) then
   pdnslog("dopping query")
   dq.appliedPolicy.policyKind = pdns.policykinds.Drop
   return false -- recursor still needs to handle the policy
  end

  if malwareset:check(dq.qname) then
    dq:addAnswer(pdns.CNAME, "blog.powerdns.com.")
    dq.rcode = 0
    dq.followupFunction = "followCNAMERecords"    -- this makes PowerDNS lookup␣
↪your CNAME
    return true
  end

  return false
end

-- this implements DNS64

function nodata(dq)
  if dq.qtype == pdns.AAAA then
    dq.followupFunction = "getFakeAAAARecords"
    dq.followupName = dq.qname
    dq.followupPrefix="fe80::"
    return true
  end

  if dq.qtype == pdns.PTR then
    dq.followupFunction = "getFakePTRRecords"
    dq.followupName = dq.qname
    dq.followupPrefix = "fe80::"
    return true
  end
  return false
end

-- postresolve runs after the packet has been answered, and can be used to change␣
↪things
-- or still drop
function postresolve(dq)
  pdnslog("postresolve called for "..dq.qname:toString())
  local records = dq:getRecords()
  for k,v in pairs(records) do
    pdnslog(k.." "..v.name:toString().." "..v:getContent())
    if v.type == pdns.A and v:getContent() == "185.31.17.73" then
      pdnslog("Changing content!")
      v:changeContent("130.161.252.29")
      v.ttl = 1
    end
  end
  dq:setRecords(records)
  return true
end

nxdomainsuffix = newDN("com")

function nxdomain(dq)
  pdnslog("nxdomain called for: "..dq.qname:toString())
  if dq.qname:isPartOf(nxdomainsuffix) then
    dq.rcode = 0 -- make it a normal answer
```

(continues on next page)

(continued from previous page)

```
    dq:addAnswer(pdns.CNAME, "ourhelpfulservice.com")
    dq:addAnswer(pdns.A, "1.2.3.4", 60, "ourhelpfulservice.com")
    return true
  end
  return false
end
```

Dropping all traffic from botnet-infected users

Frequently, DoS attacks are performed where specific IP addresses are attacked, often by queries coming in from open resolvers. These queries then lead to a lot of queries to 'authoritative servers' which actually often aren't nameservers at all, but just targets of attack.

This specific script is, as of January 2015, useful to prevent traffic to ezdns.it related traffic from creating CPU load. This script requires PowerDNS Recursor 4.x or later.

```
lethalgroup=newNMG()
lethalgroup:addMask("192.121.121.0/24") -- touch these nameservers and original␣
↪query gets dropped

function preoutquery(dq)
    print("pdns wants to ask "..dq.remoteaddr:toString().." about "..dq.
↪qname:toString().." "..dq.qtype.." on behalf of requestor "..dq.
↪localaddr:toString())
    if(lethalgroup:match(dq.remoteaddr))
    then
        print("We matched the group "..lethalgroup:tostring()."! killing query␣
↪dead from requestor "..dq.localaddr:toString())
        dq.rcode = -3 -- "kill"
        return true
    end
    return false
end
```

8.14.6 Modifying Policy Decisions

The PowerDNS Recursor has a *policy engine based on Response Policy Zones (RPZ)*. Starting with version 4.0.1 of the recursor, it is possible to alter this decision inside the Lua hooks.

If the decision is modified in a Lua hook, `false` should be returned, as the query is not actually handled by Lua so the decision is picked up by the Recursor.

Before 4.4.0, the result of the policy decision is checked after `preresolve()` and `postresolve()`. Beginning with version 4.4.0, the policy decision is checked after `preresolve()` and any `policyEventFilter()` call instead.

For example, if a decision is set to `pdns.policykinds.NODATA` by the policy engine and is unchanged in `preresolve()`, the query is replied to with a NODATA response immediately after `preresolve()`.

Example script

```
-- This script demonstrates modifying policies for versions before 4.4.0.
-- Starting with 4.4.0, it is preferred to use a policyEventFilter.
-- Dont ever block my own domain and IPs
myDomain = newDN("example.com")

myNetblock = newNMG()
```

(continues on next page)

(continued from previous page)

```
myNetblock:addMasks({"192.0.2.0/24"})

function preresolve(dq)
  if dq.qname:isPartOf(myDomain) and dq.appliedPolicy.policyKind ~= pdns.
↪policykinds.NoAction then
    pdnslog("Not blocking our own domain!")
    dq.appliedPolicy.policyKind = pdns.policykinds.NoAction
  end
  return false
end

function postresolve(dq)
  if dq.appliedPolicy.policyKind ~= pdns.policykinds.NoAction then
    local records = dq:getRecords()
    for k,v in pairs(records) do
      if v.type == pdns.A then
        local blockedIP = newCA(v:getContent())
        if myNetblock:match(blockedIP) then
          pdnslog("Not blocking our IP space")
          dq.appliedPolicy.policyKind = pdns.policykinds.NoAction
        end
      end
    end
  end
  return false
end
```

8.14.7 SNMP Traps

PowerDNS Recursor, when compiled with SNMP support, has the ability to act as a SNMP agent to provide SNMP statistics and to be able to send traps from Lua.

For example, to send a custom SNMP trap containing the qname from the `preresolve` hook:

```
function preresolve(dq)
  sendCustomSNMPTrap('Trap from preresolve, qname is '..dq.qname:toString())
  return false
end
```

8.14.8 Maintenance callback

Starting with version 4.2.0 of the recursor, it is possible to define a *maintenance()* callback function that will be called periodically. This function expects no argument and doesn't return any value.

```
function maintenance()
    -- This would be called every second
    -- Perform here your maintenance
end
```

The interval can be configured through the *lua-maintenance-interval* setting.

8.15 Lua FFI API

PowerDNS Recursor provides a set of functions available through the LUA FFI library that allow you to interact with handle passed to `gettag_ffi()` and `postresolve_ffi()`.

8.15.1 Functions for `gettag_ffi()`

pdns_ffi_param_get_qname (*pdns_ffi_param_t* ref*) → const char*
Get the query's qualified name

pdns_ffi_param_get_qtype (*const pdns_ffi_param_t* ref*) → uint16_t
Get the query's type

pdns_ffi_param_get_remote (*pdns_ffi_param_t* ref*) → const char*
Get the sender's IP address

pdns_ffi_param_get_remote_port (*const pdns_ffi_param_t* ref*) → uint16_t
Get the sender's port

pdns_ffi_param_get_local (*pdns_ffi_param_t* ref*) → const char*
Get the local IP address the query was received on

pdns_ffi_param_get_local_port (*const pdns_ffi_param_t* ref*) → uint16_t
Get the local port the query was received on

pdns_ffi_param_get_edns_cs (*pdns_ffi_param_t* ref*) → const char*
Get query's EDNS client subnet

pdns_ffi_param_get_edns_cs_source_mask (*const pdns_ffi_param_t* ref*) → uint8_t
Get query's EDNS client subnet mask

pdns_ffi_param_get_edns_options (*pdns_ffi_param_t* ref, const pdns_ednsoption_t** out*) → size_t
Get query's EDNS options. Returns the length of the resulting *out* array

pdns_ffi_param_get_edns_options_by_code (*pdns_ffi_param_t* ref, uint16_t optionCode, const pdns_ednsoption_t** out*) → size_t
Get query's EDNS option for a given code. Returns the length of the resulting *out* array

pdns_ffi_param_get_proxy_protocol_values (*pdns_ffi_param_t* ref, const pdns_proxyprotocol_value_t** out*) → size_t
Get query's proxy protocol values. Returns the length of the resulting *out* array

pdns_ffi_param_get_edns_cs_raw (*pdns_ffi_param_t* ref, const void** net, size_t* netSize*) → void
Fill out *net* with query's EDNS client subnet

pdns_ffi_param_get_remote_raw (*pdns_ffi_param_t* ref, const void** addr, size_t* addrSize*) → void
Fill out *addr* with sender's IP address

pdns_ffi_param_get_qname_raw (*pdns_ffi_param_t* ref, const char** qname, size_t* qnameSize*) → void
Fill out *qname* with query's qualified name

pdns_ffi_param_get_local_raw (*pdns_ffi_param_t* ref, const void** addr, size_t* addrSize*) → void
Fill out *addr* with local IP address the query was received on

pdns_ffi_param_set_tag (*pdns_ffi_param_t* ref, unsigned int tag*) → void
Tag the query with the given number

pdns_ffi_param_add_policytag (*pdns_ffi_param_t* ref, const char* name*) → void
Add the given tag to the query

pdns_ffi_param_set_requestorid (*pdns_ffi_param_t* ref, const char* name*) → void
Set query's requestor ID

pdns_ffi_param_set_devicename (*pdns_ffi_param_t* ref, const char* name*) → void
Set query's device name

pdns_ffi_param_set_deviceid (*pdns_ffi_param_t* ref, size_t len, const void* name*) → void
Set query's device ID

pdns_ffi_param_set_routingtag (*pdns_ffi_param_t* ref*, *const char* name*) → void
Set routing tag which is used as an extra name to identify records in the record cache, see `gettag()`

pdns_ffi_param_set_variable (*pdns_ffi_param_t* ref*, *bool variable*) → void
Mark as variable and ensure it's not inserted into the packetcache

pdns_ffi_param_set_ttl_cap (*pdns_ffi_param_t* ref*, *uint32_t ttl*) → void
Cap the max TTL of the returned records

pdns_ffi_param_set_log_query (*pdns_ffi_param_t* ref*, *bool logQuery*) → void
Turn on/off query logging

pdns_ffi_param_set_log_response (*pdns_ffi_param_t* ref*, *bool logResponse*) → void
Turn on/off response logging

pdns_ffi_param_set_rcode (*pdns_ffi_param_t* ref*, *int rcode*) → void
Set response RCode

pdns_ffi_param_set_follow_cname_records (*pdns_ffi_param_t* ref*, *bool follow*) → void
Instruct the recursor to do a proper resolution in order to follow any CNAME records added

pdns_ffi_param_set_extended_error_code (*pdns_ffi_param_t* ref*, *uint16_t code*) → void
Set extended DNS error info code

pdns_ffi_param_set_extended_error_extra (*pdns_ffi_param_t* ref*, *size_t len*, *const char* extra*) → void
Set extended DNS error extra text

pdns_ffi_param_set_padding_disabled (*pdns_ffi_param_t* ref*, *bool disabled*) → void
Disable padding

pdns_ffi_param_add_record (*pdns_ffi_param_t* ref*, *const char* name*, *uint16_t type*, *uint32_t ttl*, *const char* content*, *size_t contentSize*, *pdns_record_place_t place*) → bool
Adds a record. Returns true if it was correctly added, false otherwise

pdns_ffi_param_add_meta_single_string_kv (*pdns_ffi_param_t* ref*, *const char* key*, *const char* val*) → void
New in version 4.6.0.

This function allows you to add an arbitrary string value for a given key in the `meta` field of the produced *protobuf* log message

pdns_ffi_param_add_meta_single_int64_kv (*pdns_ffi_param_t *ref*, *const char* key*, *int64_t val*) → void
New in version 4.6.0.

This function allows you to add an arbitrary int value for a given key in the `meta` field of the produced *protobuf* log message

8.15.2 Functions for `postresolve_ffi()`

New in version 4.7.0.

All functions below were added in version 4.7.0.

pdns_postresolve_ffi_handle_get_qname (*pdns_postresolve_ffi_handle_t* ref*) → const char*
Get the name queried as a string.

pdns_postresolve_ffi_handle_get_qname_raw (*pdns_postresolve_ffi_handle_t* ref*, *const char** qname*, *size_t* qnameSize*) → void
Get the name queried (and its size) in DNS wire format.

pdns_postresolve_ffi_handle_get_qtype (*const pdns_postresolve_ffi_handle_t* ref*) → uint16
Get the qtype of the query.

pdns_postresolve_ffi_handle_get_rcode(*const pdns_postresolve_ffi_handle_t* ref) → uint16

Get the rcode returned by the resolving process.

pdns_postresolve_ffi_handle_set_rcode(*const pdns_postresolve_ffi_handle_t* ref, *uint16_t rcode*) → void

Set the rcode to be returned.

pdns_postresolve_ffi_handle_get_appliedpolicy_kind(*const pdns_postresolve_ffi_handle_t* ref) → pdns_policy_kind_t

Get the applied policy.

pdns_postresolve_ffi_handle_set_appliedpolicy_kind(*pdns_postresolve_ffi_handle_t* ref, *pdns_policy_kind_t kind*) → void

Set the applied policy.

pdns_postresolve_ffi_handle_get_record(*pdns_postresolve_ffi_handle_t* ref, *unsigned int i*, *pdns_ffi_record_t* record*, *bool raw*) → bool

Get a record indexed by i. Returns false if no record is available at index i.

pdns_postresolve_ffi_handle_set_record(*pdns_postresolve_ffi_handle_t* ref, *unsigned int i*, *const char* content*, *size_t contentLen*, *bool raw*) → bool

Set the record at index i.

pdns_postresolve_ffi_handle_clear_records(*pdns_postresolve_ffi_handle_t* ref) → void

Clear all records.

pdns_postresolve_ffi_handle_add_record(*pdns_postresolve_ffi_handle_t* ref, *const char* name*, *uint16_t type*, *uint32_t ttl*, *const char* content*, *size_t contentLen*, *pdns_record_place_t place*, *bool raw*) → bool

Add a record to the existing records.

pdns_postresolve_ffi_handle_get_authip(*pdns_postresolve_ffi_handle_t* ref) → const char*

Get a string representation of the IP address of the authoritative server that answered the query. The string might by empty if the address is not available.

pdns_postresolve_ffi_handle_get_authip_raw(*pdns_postresolve_ffi_handle_t* ref, *const void** addr*, *size_t* addrSize*) → void

Get the raw IP address (in network byte order) and size of the raw IP address of the authoritative server that answered the query. The string might be empty if the address is not available.

8.16 Other functions

These are some functions that don't really have a place in one of the other categories.

getregisteredname(*name*) → str

Returns the shortest domain name based on Mozilla's Public Suffix List. In general it will tell you the 'registered domain' for a given name.

For example `getregisteredname('www.powerdns.com')` returns "powerdns.com"

Parameters **name** (*str*) – The name to check for.

getRecursorThreadId() → int

returns an unsigned integer identifying the thread handling the current request.

pdnsrandom([*upper_bound*])

Get a random number.

Parameters **upper_bound** (*int*) – The upper bound. You will get a random number below this upper bound.

8.17 Checking available features

New in version 4.3.0.

To check if a Lua feature is available, consult the global `pdns_features` table. This table contains string keys with values of type boolean, string or number. If a key is absent the value will evaluate to `nil`, indicating the feature is not available.

Currently, the following keys are defined:

```
pdns_feature["PR8001_devicename"] = true
```

CHAPTER

NINE

DNS64 SUPPORT

DNS64, described in **RFC 6147** is a technology to allow IPv6-only clients to receive special IPv6 addresses that are proxied to IPv4 addresses. This proxy service is then called NAT64.

As an example, let's say an IPv6 only client would want to connect to `www.example.com`, it would request the AAAA records for that name. However, if `example.com` does not actually have an IPv6 address, what we do is 'fake up' an IPv6 address. We do this by retrieving the A records for `www.example.com`, and translating them to AAAA records. Elsewhere, a NAT64 device listens on these IPv6 addresses, and extracts the IPv4 address from each packet, and proxies it on.

As of 4.4.0, an efficient implementation is built the recursor and can be enabled via the using the *dns64-prefix setting*.

9.1 Native DNS64 support

Native DNS64 processing will happen after calling a `nodata` or `nxdomain` Lua hook (if defined), but before calling a `postresolve` or `postresolve_ffi` Lua hook (if defined).

To consider native DNS64 processing the following conditions must be met:

- The *dns64-prefix* is defined.
- A `nodata` or `nxdomain` Lua hook did not return `true`.
- The original query type was `AAAA`.
- The result code of the `AAAA` query was not `NXDomain`.
- No relevant answer was received: the result code was `NoError` with no relevant answer records, or an error unequal to `NXDomain` occurred.
- If DNSSEC processing is requested the validation result was not `Bogus`.

Before version 4.8.0, only `NoError` results were considers candidates for DNS64 processing.

9.2 Scripted DNS64 Support

On earlier versions or for maximum flexibility, DNS64 support is included in the *Scripting PowerDNS Recursor*. This allows for example to hand out custom IPv6 gateway ranges depending on the location of the requestor, enabling the use of NAT64 services close to the user.

Apart from faking AAAA records, it is also possible to also generate the associated PTR records. This makes sure that reverse lookup of DNS64-generated IPv6 addresses generate the right name. The procedure is similar, a request for an IPv6 PTR is converted into one for the corresponding IPv4 address.

To setup DNS64, with both forward and reverse records, create the following Lua script and save it to a file called `dns64.lua`

```
-- this small script implements dns64 without any specials or customization
prefix = "fe80::21b:77ff:0:0"

function nodata ( dq )
  if dq.qtype ~= pdns.AAAA then
    return false
  end  --  only AAAA records

  -- don't fake AAAA records if DNSSEC validation failed
  if dq.validationState == pdns.validationstates.Bogus then
     return false
  end

  dq.followupFunction = "getFakeAAAARecords"
  dq.followupPrefix = prefix
  dq.followupName = dq.qname
  return true
end

-- the ip6.arpa address is the reverse of the prefix address above
function preresolve ( dq )
  if dq.qtype == pdns.PTR and dq.qname:isPartOf(newDN("f.f.7.7.b.1.2.0.0.0.0.0.0.0.0.
↪0.0.0.0.0.0.0.8.e.f.ip6.arpa.")) then
    dq.followupFunction = "getFakePTRRecords"
    dq.followupPrefix = prefix
    dq.followupName = dq.qname
    return true
  end
  return false
end
```

Where fe80::21b:77ff:0:0 is your "Pref64" translation prefix and the "ip6.arpa" string is the reversed form of this Pref64 address. Now ensure your script gets loaded by specifying it with *lua-dns-script=dns64.lua*.

On our wiki, a user has kindly supplied an example script with support for multiple prefixes.

To enhance DNS64, see the *Scripting PowerDNS Recursor* documentation.

CHAPTER

TEN

METRICS AND STATISTICS

The PowerDNS Recursor collects many statistics about itself.

10.1 Regular Statistics Log

Every half hour or so (configurable with *statistics-interval*, the recursor outputs a line with statistics. To force the output of statistics, send the process a SIGUSR1. A line of statistics looks like this:

```
stats: 346362 questions, 7388 cache entries, 1773 negative entries, 18% cache hits
stats: cache contended/acquired 1583/56041728 = 0.00282468%
stats: throttle map: 3, ns speeds: 1487, failed ns: 15, ednsmap: 1363
stats: outpacket/query ratio 54%, 0% throttled, 0 no-delegation drops
stats: 217 outgoing tcp connections, 0 queries running, 9155 outgoing timeouts
stats: 4536 packet cache entries, 82% packet cache hits
stats: thread 0 has been distributed 175728 queries
stats: thread 1 has been distributed 169484 queries
stats: 1 qps (average over 1800 seconds)
```

This means that in total 346362 queries were received and there are 7388 different name/type combinations in the record cache, each entry may have multiple records attached to it.

There are 1773 items in the negative cache, items of which it is known that don't exist and won't do so for the near future. 18% of incoming questions not handled by the packets cache could be answered without any additional queries going out to the net. The record cache was consulted or modified 56041728 times, and 1583 of those accesses caused lock contention.

Next a line with the sizes of maps that can be consulted by **`rec_control`** is printed.

The outpacket/query ratio means that on average, 0.54 packets were needed to answer a question. This ratio can be greater than 100% since additional queries could be needed to actually recurse the DNS and figure out the addresses of nameservers.

0% of queries were not performed because identical queries had gone out previously and failed, saving load on servers worldwide. 217 outgoing tcp connections were done, there were 0 queries running at the moment and 9155 queries to authoritative servers saw timeouts.

The packets cache had 4536 entries and 82% of queries were served from it. The workload of the worker queries was 175728 and 169484 respectively. Finally, measured in the last half hour, an average of 1 qps was performed.

10.2 Multi-threading and metrics

Some metrics are collected in thread-local variables, and an aggregate values is computed to report. Other statistics are recorded in global memory and each thread updates the one instance, taking proper precautions to make sure consistency is maintained. The only exception are the *cpu-msec-thread-N* metrics, which report per-thread data.

10.3 Sending metrics to Graphite/Metronome over Carbon

For carbon/graphite/metronome, we use the following namespace. Everything starts with 'pdns.', which is then followed by the local hostname. Thirdly, we add 'recursor' to signify the daemon generating the metrics. This is then rounded off with the actual name of the metric. As an example: 'pdns.ns1.recursor.questions'.

Care has been taken to make the sending of statistics as unobtrusive as possible, the daemons will not be hindered by an unreachable carbon server, timeouts or connection refused situations.

To benefit from our carbon/graphite support, either install Graphite, or use our own lightweight statistics daemon, Metronome, currently available on GitHub.

To enable sending metrics, set *carbon-server*, possibly *carbon-interval* and possibly *carbon-ourname* in the configuration.

> **Warning:** If your hostname includes dots, they will be replaced by underscores so as not to confuse the namespace.
>
> If you include dots in *carbon-ourname*, they will **not** be replaced by underscores. As PowerDNS assumes you know what you are doing if you override your hostname.

10.4 Getting Metrics from the Recursor

Should Carbon not be the preferred way of receiving metrics, several other techniques can be employed to retrieve them.

10.4.1 Using the Webserver

The *API* exposes a statistics endpoint at

`GET /api/v1/servers/:server_id/statistics`

This endpoint exports all statistics in a single JSON document.

10.4.2 Using `rec_control`

Metrics can also be gathered on the system itself by invoking *rec_control*:

```
rec_control get-all
```

Single statistics can also be retrieved with the `get` command, e.g.:

```
rec_control get all-outqueries
```

External programs can use this technique to scrape metrics, though it is preferred to use a Prometheus export.

10.4.3 Using Prometheus export

The internal web server exposes Prometheus formatted metrics at

`GET /metrics`

The Prometheus name are the names listed in *metricnames*, prefixed with `pdns_recursor_` and with hyphens substituted by underscores. For example:

```
# HELP pdns_recursor_all_outqueries Number of outgoing UDP queries since starting
# TYPE pdns_recursor_all_outqueries counter
pdns_recursor_all_outqueries 7
```

10.5 Sending metrics over SNMP

The recursor can export statistics over SNMP and send traps from *Lua*, provided support is compiled into the Recursor and *snmp-agent* set.

For the details of all values that can be retrieved using SNMP, see the SNMP MIB.

10.6 Gathered Information

These statistics are gathered.

It should be noted that answers0-1 + answers1-10 + answers10-100 + answers100-1000 + answers-slow + packetcache-hits + over-capacity-drops + policy-drops = questions.

Also note that unauthorized-tcp and unauthorized-udp packets do not end up in the 'questions' count.

10.6.1 almost-expired-pushed

New in version 4.6.

number of almost-expired tasks pushed

10.6.2 almost-expired-run

New in version 4.6.

number of almost-expired tasks run

10.6.3 almost-expired-exceptions

New in version 4.6.

number of almost-expired tasks that caused an exception

10.6.4 aggressive-nsec-cache-entries

New in version 4.5.

number of entries in the aggressive NSEC cache

10.6.5 aggressive-nsec-cache-nsec-hits

New in version 4.5.

number of negative answers generated from NSEC entries by the aggressive NSEC cache

10.6.6 aggressive-nsec-cache-nsec3-wc-hits

New in version 4.5.

number of answers synthesized from NSEC entries and wildcards by the NSEC aggressive cache

10.6.7 aggressive-nsec-cache-nsec3-wc-hits

New in version 4.5.

number of answers synthesized from NSEC entries and wildcards by the NSEC3 aggressive cache

10.6.8 all-outqueries

counts the number of outgoing queries since starting, this includes UDP, TCP, DoT queries both over IPv4 and IPv6

10.6.9 answers-slow

counts the number of queries answered after 1 second

10.6.10 answers0-1

counts the number of queries answered within 1 millisecond

10.6.11 answers1-10

counts the number of queries answered within 10 milliseconds

10.6.12 answers10-100

counts the number of queries answered within 100 milliseconds

10.6.13 answers100-1000

counts the number of queries answered within 1 second

10.6.14 auth4-answers-slow

counts the number of queries answered by auth4s after 1 second (4.0)

10.6.15 auth4-answers0-1

counts the number of queries answered by auth4s within 1 millisecond (4.0)

10.6.16 auth4-answers1-10

counts the number of queries answered by auth4s within 10 milliseconds (4.0)

10.6.17 auth4-answers10-100

counts the number of queries answered by auth4s within 100 milliseconds (4.0)

10.6.18 auth4-answers100-1000

counts the number of queries answered by auth4s within 1 second (4.0)

10.6.19 auth6-answers-slow

counts the number of queries answered by auth6s after 1 second (4.0)

10.6.20 auth6-answers0-1

counts the number of queries answered by auth6s within 1 millisecond (4.0)

10.6.21 auth6-answers1-10

counts the number of queries answered by auth6s within 10 milliseconds (4.0)

10.6.22 auth6-answers10-100

counts the number of queries answered by auth6s within 100 milliseconds (4.0)

10.6.23 auth6-answers100-1000

counts the number of queries answered by auth6s within 1 second (4.0)

10.6.24 auth-xxx-answers

where `xxx` is an rcode name (`noerror`, `formerr`, `servfail`, `nxdomain`, `notimp`, `refused`, `yxdomain`, `yxrrset`, `nxrrset`, `notauth`, `rcode10`, `rcode11`, `rcode2`, `rcode13`, `rcode14`, `rcode15`). Counts the rcodes returned by authoritative servers. The corresponding Prometheus metrics consist of multiple entries of the form `pdns_recursor_auth_rcode_answers{rcode="xxx"}`.

10.6.25 auth-zone-queries

counts the number of queries to locally hosted authoritative zones (*auth-zones*) since starting

10.6.26 cache-bytes

size of the cache in bytes (disabled by default, see *stats-rec-control-disabled-list*) This metric is a rough estimate and takes a long time to compute, and is therefore not enabled in default outputs.

10.6.27 cache-entries

shows the number of entries in the cache

10.6.28 cache-hits

counts the number of cache hits since starting, this does **not** include hits that got answered from the packet-cache

10.6.29 cache-misses

counts the number of cache misses since starting

10.6.30 case-mismatches

counts the number of mismatches in character case since starting

10.6.31 chain-resends

number of queries chained to existing outstanding query

10.6.32 client-parse-errors

counts number of client packets that could not be parsed

10.6.33 concurrent-queries

shows the number of MThreads currently running

10.6.34 cpu-msec-thread-n

shows the number of milliseconds spent in thread n. Available since 4.1.12.

10.6.35 cpu-iowait

New in version 4.4.

Time spent waiting for I/O to complete by the whole system, in units of USER_HZ.

10.6.36 cpu-steal

New in version 4.4.

Stolen time, which is the time spent by the whole system in other operating systems when running in a virtualized environment, in units of USER_HZ.

10.6.37 cumul-authanswers-x

New in version 4.6.

Cumulative counts of answer times of authoritative servers in buckets less than x microseconds. (disabled by default, see *stats-rec-control-disabled-list*) These metrics are useful for Prometheus and not listed in other outputs by default.

10.6.38 cumul-clientanswers-x

New in version 4.6.

Cumulative counts of our answer times to clients in buckets less or equal than x microseconds. These metrics include packet cache hits. These metrics are useful for Prometheus and not listed in other outputs by default.

10.6.39 dns64-prefix-answers

New in version 4.6.

number of `AAAA` and `PTR` answers generated by *dns64-prefix* matching.

10.6.40 dnssec-authentic-data-queries

New in version 4.2.

number of queries received with the AD bit set

10.6.41 dnssec-check-disabled-queries

New in version 4.2.

number of queries received with the CD bit set

10.6.42 dnssec-queries

number of queries received with the DO bit set

10.6.43 dnssec-result-bogus

number of responses sent, packet-cache hits excluded, that were in the DNSSEC Bogus state. Since 4.4.2 detailed counters are available, see below. Since 4.5.0, if *x-dnssec-names* is set, a separate set of `x-dnssec-result-...` metrics become available, counting the DNSSEC validation results for names suffix-matching a name in `x-dnssec-names`.

10.6.44 dnssec-result-bogus-no-valid-dnskey

New in version 4.4.2.

number of responses sent, packet-cache hits excluded, that were in the Bogus state because a valid DNSKEY could not be found.

10.6.45 dnssec-result-bogus-invalid-denial

New in version 4.4.2.

number of responses sent, packet-cache hits excluded, that were in the Bogus state because a valid denial of existence proof could not be found.

10.6.46 dnssec-result-bogus-unable-to-get-dss

New in version 4.4.2.

number of responses sent, packet-cache hits excluded, that were in the Bogus state because a valid DS could not be retrieved.

10.6.47 dnssec-result-bogus-unable-to-get-dnskeys

New in version 4.4.2.

number of responses sent, packet-cache hits excluded, that were in the Bogus state because a valid DNSKEY could not be retrieved.

10.6.48 dnssec-result-bogus-self-signed-ds

New in version 4.4.2.

number of responses sent, packet-cache hits excluded, that were in the Bogus state because a DS record was signed by itself.

10.6.49 dnssec-result-bogus-no-rrsig

New in version 4.4.2.

number of responses sent, packet-cache hits excluded, that were in the Bogus state because required RRSIG records were not present in an answer.

10.6.50 dnssec-result-bogus-no-valid-rrsig

New in version 4.4.2.

number of responses sent, packet-cache hits excluded, that were in the Bogus state because only invalid RRSIG records were present in an answer.

10.6.51 dnssec-result-bogus-missing-negative-indication

New in version 4.4.2.

number of responses sent, packet-cache hits excluded, that were in the Bogus state because a NODATA or NXDOMAIN answer lacked the required SOA and/or NSEC(3) records.

10.6.52 dnssec-result-bogus-signature-no-yet-valid

New in version 4.4.2.

number of responses sent, packet-cache hits excluded, that were in the Bogus state because the signature inception time in the RRSIG was not yet valid.

10.6.53 dnssec-result-bogus-signature-expired

New in version 4.4.2.

number of responses sent, packet-cache hits excluded, that were in the Bogus state because the signature expired time in the RRSIG was in the past.

10.6.54 dnssec-result-bogus-unsupported-dnskey-algo

New in version 4.4.2.

number of responses sent, packet-cache hits excluded, that were in the Bogus state because a DNSKEY RRset contained only unsupported DNSSEC algorithms.

10.6.55 dnssec-result-bogus-unsupported-ds-digest-type

New in version 4.4.2.

number of responses sent, packet-cache hits excluded, that were in the Bogus state because a DS RRset contained only unsupported digest types.

10.6.56 dnssec-result-bogus-no-zone-key-bit-set

New in version 4.4.2.

number of responses sent, packet-cache hits excluded, that were in the Bogus state because no DNSKEY with the Zone Key bit set was found.

10.6.57 dnssec-result-bogus-revoked-dnskey

New in version 4.4.2.

number of responses sent, packet-cache hits excluded, that were in the Bogus state because all DNSKEYs were revoked.

10.6.58 dnssec-result-bogus-invalid-dnskey-protocol

New in version 4.4.2.

number of responses sent, packet-cache hits excluded, that were in the Bogus state because all DNSKEYs had invalid protocols.

10.6.59 dnssec-result-indeterminate

number of DNSSEC validations that had the Indeterminate state

10.6.60 dnssec-result-insecure

number of responses sent, packet-cache hits excluded, that were in the Insecure state

10.6.61 dnssec-result-nta

number of responses sent, packet-cache hits excluded, that were in the NTA (negative trust anchor) state

10.6.62 dnssec-result-secure

number of responses sent, packet-cache hits excluded, that were in the Secure state

10.6.63 dnssec-validations

number of responses sent, packet-cache hits excluded, for which a DNSSEC validation was requested by either the client or the configuration

10.6.64 dont-outqueries

number of outgoing queries dropped because of *dont-query* setting (since 3.3)

10.6.65 dot-outqueries

counts the number of outgoing DoT queries since starting, both using IPv4 and IPv6

10.6.66 qname-min-fallback-success

New in version 4.3.0.

number of successful queries due to fallback mechanism within *qname-minimization* setting.

10.6.67 ecs-queries

number of outgoing queries adorned with an EDNS Client Subnet option (since 4.1)

10.6.68 ecs-responses

number of responses received from authoritative servers with an EDNS Client Subnet option we used (since 4.1)

10.6.69 ecs-v4-response-bits-*

New in version 4.2.0.

number of responses received from authoritative servers with an IPv4 EDNS Client Subnet option we used, of this subnet size (1 to 32). (disabled by default, see *stats-rec-control-disabled-list*)

10.6.70 ecs-v6-response-bits-*

New in version 4.2.0.

number of responses received from authoritative servers with an IPv6 EDNS Client Subnet option we used, of this subnet size (1 to 128). (disabled by default, see *stats-rec-control-disabled-list*)

10.6.71 edns-ping-matches

number of servers that sent a valid EDNS PING response

10.6.72 edns-ping-mismatches

number of servers that sent an invalid EDNS PING response

10.6.73 failed-host-entries

number of addresses in the failed NS cache.

10.6.74 fd-usage

Number of currently used file descriptors. Currently, this metric is available on Linux and OpenBSD only.

10.6.75 ignored-packets

counts the number of non-query packets received on server sockets that should only get query packets

10.6.76 ipv6-outqueries

number of outgoing queries over IPv6 using UDP, since version 5.0.0 also including TCP and DoT

10.6.77 ipv6-questions

counts all client initiated queries using IPv6

10.6.78 maintenance-usec

time spent doing internal maintenance, including Lua maintenance

10.6.79 maintenance-calls

number of times internal maintenance has been called, including Lua maintenance

10.6.80 malloc-bytes

returns the number of bytes allocated by the process (broken, always returns 0)

10.6.81 max-cache-entries

currently configured maximum number of cache entries

10.6.82 max-chain-length

maximum chain length

10.6.83 max-chain-weight

maximum chain weight. The weight of a chain of outgoing queries is the product of the number of chained queries by the size of the response received from the external authoritative server.

10.6.84 max-packetcache-entries

currently configured maximum number of packet cache entries

10.6.85 max-mthread-stack

maximum amount of thread stack ever used

10.6.86 negcache-entries

shows the number of entries in the negative answer cache

10.6.87 no-packet-error

number of erroneous received packets

10.6.88 nod-events

New in version 4.9.0.

Count of NOD events

10.6.89 udr-events

New in version 4.9.0.

Count of UDR events

10.6.90 nod-lookups-dropped-oversize

Number of NOD lookups dropped because they would exceed the maximum name length

10.6.91 noedns-outqueries

number of queries sent out without EDNS

10.6.92 noerror-answers

counts the number of times it answered NOERROR since starting

10.6.93 non-resolving-nameserver-entries

number of entries in the non-resolving NS name cache

10.6.94 noping-outqueries

number of queries sent out without ENDS PING

10.6.95 nsset-invalidations

number of times an nsset was dropped because it no longer worked

10.6.96 nsspeeds-entries

shows the number of entries in the NS speeds map

10.6.97 nxdomain-answers

counts the number of times it answered NXDOMAIN since starting

10.6.98 outgoing-timeouts

counts the number of timeouts on outgoing UDP queries since starting

10.6.99 outgoing4-timeouts

counts the number of timeouts on outgoing UDP IPv4 queries since starting (since 4.0)

10.6.100 outgoing6-timeouts

counts the number of timeouts on outgoing UDP IPv6 queries since starting (since 4.0)

10.6.101 over-capacity-drops

questions dropped because over maximum concurrent query limit (since 3.2)

10.6.102 packetcache-bytes

size of the packet cache in bytes (since 3.3.1) (disabled by default, see *stats-rec-control-disabled-list*) This metric is a rough estimate and takes a long time to compute, and is therefore not enabled in default outputs.

10.6.103 packetcache-entries

size of packet cache (since 3.2)

10.6.104 packetcache-hits

packet cache hits (since 3.2)

10.6.105 packetcache-misses

packet cache misses (since 3.2)

10.6.106 policy-drops

packets dropped because of (Lua) policy decision

10.6.107 policy-hits

Number of policy decisions based on Lua (`type = "filter"`), or RPZ (`type = "rpz"`). RPZ hits include the *policyName*. These metrics are useful for Prometheus and not listed in other outputs by default.

10.6.108 policy-result-noaction

packets that were not acted upon by the RPZ/filter engine

10.6.109 policy-result-drop

packets that were dropped by the RPZ/filter engine

10.6.110 policy-result-nxdomain

packets that were replied to with NXDOMAIN by the RPZ/filter engine

10.6.111 policy-result-nodata

packets that were replied to with no data by the RPZ/filter engine

10.6.112 policy-result-truncate

packets that were forced to TCP by the RPZ/filter engine

10.6.113 policy-result-custom

packets that were sent a custom answer by the RPZ/filter engine

10.6.114 proxy-protocol-invalid

New in version 4.4.

Invalid proxy-protocol headers received.

10.6.115 qa-latency

shows the current latency average, in microseconds, exponentially weighted over past 'latency-statistic-size' packets

10.6.116 query-pipe-full-drops

New in version 4.2.

questions dropped because the query distribution pipe was full

10.6.117 questions

counts all end-user initiated queries with the RD bit set

10.6.118 rebalanced-queries

New in version 4.1.12.

number of queries balanced to a different worker thread because the first selected one was above the target load configured with 'distribution-load-factor'

10.6.119 record-cache-acquired

New in version 4.4.0.

number of record cache lock acquisitions

10.6.120 record-cache-contended

New in version 4.4.0.

number of contended record cache lock acquisitions

10.6.121 resource-limits

Counts the number of queries that could not be performed because of resource limits. This counter is increased when Recursor encounters a network issue that does not seem to be caused by the remote end. For example when it runs out of file descriptors (monitor *fd-usage*) or when there is no route to a given IP address.

10.6.122 security-status

security status based on *Security Polling*

10.6.123 server-parse-errors

counts number of server replied packets that could not be parsed

10.6.124 servfail-answers

counts the number of times it answered SERVFAIL since starting

10.6.125 spoof-prevents

number of times PowerDNS considered itself spoofed, and dropped the data

10.6.126 sys-msec

number of CPU milliseconds spent in 'system' mode

10.6.127 taskqueue-pushed

New in version 4.5.0.

number of tasks pushed to the taskqueue

10.6.128 taskqueue-expired

New in version 4.5.0.

number of tasks expired before they could be run

10.6.129 taskqueue-size

New in version 4.5.0.

number of tasks currently in the taskqueues

10.6.130 tcp-client-overflow

number of times an IP address was denied TCP access because it already had too many connections

10.6.131 tcp-clients

counts the number of currently active TCP/IP clients

10.6.132 tcp-outqueries

counts the number of outgoing TCP queries since starting, both using IPv4 and IPV6

10.6.133 tcp-questions

counts all incoming TCP queries (since starting)

10.6.134 throttle-entries

shows the number of entries in the throttle map

10.6.135 throttled-out

counts the number of throttled outgoing UDP queries since starting

10.6.136 throttled-outqueries

idem to throttled-out

10.6.137 too-old-drops

questions dropped that were too old

10.6.138 truncated-drops

New in version 4.2.

questions dropped because they were larger than 512 bytes

10.6.139 empty-queries

New in version 4.2.

questions dropped because they had a QD count of 0

10.6.140 unauthorized-tcp

number of TCP questions denied because of allow-from restrictions

10.6.141 unauthorized-udp

number of UDP questions denied because of allow-from restrictions

10.6.142 source-disallowed-notify

number of NOTIFY operations denied because of allow-notify-from restrictions

10.6.143 zone-disallowed-notify

number of NOTIFY operations denied because of allow-notify-for restrictions

10.6.144 unexpected-packets

number of answers from remote servers that were unexpected (might point to spoofing)

10.6.145 unreachables

number of times nameservers were unreachable since starting

10.6.146 uptime

number of seconds process has been running (since 3.1.5)

10.6.147 user-msec

number of CPU milliseconds spent in 'user' mode

10.6.148 variable-responses

New in version 4.2.

Responses that were marked as 'variable'. This could be because of EDNS Client Subnet or Lua rules that indicate this variable status (dependent on time or who is asking, for example).

10.6.149 x-our-latency

New in version 4.1: Not yet proven to be reliable

PowerDNS measures per query how much time has been spent waiting on authoritative servers. In addition, the Recursor measures the total amount of time needed to answer a question. The difference between these two durations is a measure of how much time was spent within PowerDNS. This metric is the average of that difference, in microseconds.

10.6.150 x-ourtime0-1

New in version 4.1: Not yet proven to be reliable

Counts responses where between 0 and 1 milliseconds was spent within the Recursor. See *x-our-latency* for further details.

10.6.151 x-ourtime1-2

New in version 4.1: Not yet proven to be reliable

Counts responses where between 1 and 2 milliseconds was spent within the Recursor. See *x-our-latency* for further details.

10.6.152 x-ourtime2-4

New in version 4.1: Not yet proven to be reliable

Counts responses where between 2 and 4 milliseconds was spent within the Recursor. Since 4.1. See *x-our-latency* for further details.

10.6.153 x-ourtime4-8

New in version 4.1: Not yet proven to be reliable

Counts responses where between 4 and 8 milliseconds was spent within the Recursor. See *x-our-latency* for further details.

10.6.154 x-ourtime8-16

New in version 4.1: Not yet proven to be reliable

Counts responses where between 8 and 16 milliseconds was spent within the Recursor. See *x-our-latency* for further details.

10.6.155 x-ourtime16-32

New in version 4.1: Not yet proven to be reliable

Counts responses where between 16 and 32 milliseconds was spent within the Recursor. See *x-our-latency* for further details.

10.6.156 x-ourtime-slow

New in version 4.1: Not yet proven to be reliable

Counts responses where more than 32 milliseconds was spent within the Recursor. See *x-our-latency* for further details.

10.6.157 x-dnssec-result-...

New in version 4.5.0.

See *dnssec-result-bogus*.

CHAPTER

ELEVEN

PERFORMANCE GUIDE

To get the best out of the PowerDNS recursor, which is important if you are doing thousands of queries per second, please consider the following.

A busy server may need hundreds of file descriptors on startup, and deals with spikes better if it has that many available later on. Linux by default restricts processes to 1024 file descriptors, which should suffice most of the time, but Solaris has a default limit of 256. This can be raised using the `ulimit` command or via the `LimitNOFILE` unit directive when `systemd` is used. FreeBSD has a default limit that is high enough for even very heavy duty use.

Limit the size of the caches to a sensible value. Cache hit rate does not improve meaningfully beyond a few million *max-cache-entries*, reducing the memory footprint reduces CPU cache misses. See below for more information about the various caches.

When deploying (large scale) IPv6, please be aware some Linux distributions leave IPv6 routing cache tables at very small default values. Please check and if necessary raise `sysctl net.ipv6.route.max_size`.

Set *threads* to your number of CPU cores minus the number of distributor threads.

11.1 Threading and distribution of queries

When running with several threads, you can either ask PowerDNS to start one or more special threads to dispatch the incoming queries to the workers by setting *pdns-distributes-queries* to `yes`, or let the worker threads handle the incoming queries themselves. The latter is the default since version 4.9.0.

The dispatch thread enabled by *pdns-distributes-queries* tries to send the same queries to the same thread to maximize the cache-hit ratio. If the incoming query rate is so high that the dispatch thread becomes a bottleneck, you can increase *distributor-threads* to use more than one.

If *pdns-distributes-queries* is set to `no` and either `SO_REUSEPORT` support is not available or the *reuseport* directive is set to `no`, all worker threads share the same listening sockets.

This prevents a single thread from having to handle every incoming queries, but can lead to thundering herd issues where all threads are awoken at once when a query arrives.

If `SO_REUSEPORT` support is available and *reuseport* is set to `yes`, which is the default since version 4.9.0, separate listening sockets are opened for each worker thread and the query distributions is handled by the kernel, avoiding any thundering herd issue as well as preventing the distributor thread from becoming the bottleneck. The next section discusses how to determine if the mechanism is working properly.

11.1.1 Imbalance

Due to the nature of the distribution method used by the kernel imbalance with the new default settings of *reuseport* and *pdns-distributes-queries* may occur if you have very few clients. Imbalance can be observed by reading the periodic statistics reported by **`Recursor`**:

```
Jun 26 11:06:41 pepper pdns-recursor[10502]: msg="Queries handled by thread"␣
↪subsystem="stats" level="0" prio="Info" tid="0" ts="1687770401.359" count="7"␣
↪thread="0"
Jun 26 11:06:41 pepper pdns-recursor[10502]: msg="Queries handled by thread"␣
↪subsystem=" stats" level="0" prio="Info" tid="0" ts="1687770401.359" count=
↪"535167" thread="1"
Jun 26 11:06:41 pepper pdns-recursor[10502]: msg="Queries handled by thread"␣
↪subsystem=" stats" level="0" prio="Info" tid="0" ts="1687770401.359" count="5"␣
↪thread="2"
```

In the above log lines we see that almost all queries are processed by thread 1. This can typically be observed when using `dnsdist` in front of **`Recursor`**.

When using `dnsdist` with a single `newServer` to a recursor instance in its configuration, the kernel will regard `dnsdist` as a single client unless you use the `sockets` parameter to `newServer` to increase the number of source ports used by `dnsdist`. The following guideline applies for the `dnsdist` case:

- Be generous with the `sockets` setting of `newServer`. A starting points is to configure twice as many sockets as **`Recursor`** threads.
- As long as the threads of the **`Recursor`** as not overloaded, some imbalance will not impact performance significantly.
- If you want to reduce imbalance, increase the value of `sockets` even more.

11.1.2 Non-Linux systems

On some systems setting *reuseport* to `yes` does not have the desired effect at all. If your systems shows great imbalance in the number of queries processed per thread (as reported by the periodic statistics report), try switching *reuseport* to `no` and/or setting *pdns-distributes-queries* to `yes`.

New in version 4.1.0: The *cpu-map* parameter can be used to pin worker threads to specific CPUs, in order to keep caches as warm as possible and optimize memory access on NUMA systems.

New in version 4.2.0: The *distributor-threads* parameter can be used to run more than one distributor thread.

Changed in version 4.9.0: The *reuseport* parameter now defaults to `yes`.

Changed in version 4.9.0: The *pdns-distributes-queries* parameter now defaults to `no`.

11.2 MTasker and MThreads

PowerDNS **`Recursor`** uses a cooperative multitasking in userspace called `MTasker`, based either on `boost::context` if available, or on `System V ucontexts` otherwise. For maximum performance, please make sure that your system supports `boost::context`, as the alternative has been known to be quite slower.

The maximum number of simultaneous MTasker threads, called `MThreads`, can be tuned via *max-mthreads*, as the default value of 2048 might not be enough for large-scale installations. This setting limits the number of mthreads *per physical (Posix) thread*. The threads that create mthreads are the distributor and worker threads.

When a `MThread` is started, a new stack is dynamically allocated for it. The size of that stack can be configured via the *stack-size* parameter, whose default value is 200 kB which should be enough in most cases.

To reduce the cost of allocating a new stack for every query, the recursor can cache a small amount of stacks to make sure that the allocation stays cheap. This can be configured via the *stack-cache-size* setting. This limit is per physical (Posix) thread. The only trade-off of enabling this cache is a slightly increased memory consumption, at worst equals to the number of stacks specified by *stack-cache-size* multiplied by the size of one stack, itself specified via *stack-size*.

Linux limits the number of memory mappings a process can allocate by the `vm.max_map_count` kernel parameter. A single `MThread` stack can take up to 3 memory mappings. Starting with version 4.9, it is advised to check

and if needed update the value of `sysctl vm.max_map_count` to make sure that the **Recursor** can allocate enough stacks under load; suggested value is at least `4 * (threads + 2) * max-mthreads`. Some Linux distributions use a default value of about one million, which should be enough for most configurations. Other distributions default to 64k, which can be too low for large setups.

11.3 Performance tips

For best PowerDNS Recursor performance, use a recent version of your operating system, since this generally offers the best event multiplexer implementation available (`kqueue`, `epoll`, `ports` or `/dev/poll`).

On AMD/Intel hardware, wherever possible, run a 64-bit binary. This delivers a nearly twofold performance increase. On UltraSPARC, there is no need to run with 64 bits.

Consider performing a 'profiled build' by building with `gprof` support enabled, running the recursor a bit then feed that info into the next build. This is good for a 20% performance boost in some cases.

When running with >3000 queries per second, and running Linux versions prior to 2.6.17 on some motherboards, your computer may spend an inordinate amount of time working around an ACPI bug for each call to gettimeofday. This is solved by rebooting with `clock=tsc` or upgrading to a 2.6.17 kernel. This is relevant if dmesg shows `Using pmtmr for high-res timesource`.

11.4 Memory usage

Recursor keeps all the data it needs in memory. The default configuration uses a little more than 1GB when the caches are full. Depending on configuration, memory usage can amount to many gigabytes for a large installation.

> **Warning:** Avoid swapping. The memory access patterns of **Recursor** are random. This means that it will cause trashing (the OS spending lots of time pulling in and writing out memory pages) if **Recursor** uses more physical memory than available and performance will be severely impacted.

Below the memory usage observed for a specific test case are described. Please note that depending on OS, version of system libraries, version of the **Recursor**, features used and usage patterns these numbers may vary. Test and observe your system to learn more about the memory requirements specific to your case.

The most important subsystems that use memory are:

- The packet cache. The amount of memory used in a test case was about 500 bytes per entry
- The record cache. The amount of memory used in a test case was about 850 bytes per entry
- Authoritative zones loaded. Memory usage is dependent on the size and number loaded.
- RPZ zones loaded. Memory usage is dependent on the size and number loaded.
- NOD DBs. Memory usage is dependent on specific settings of this subsystem.

An estimate for the memory used by its caches for a **Recursor** having 2 million record cache entries and 1 million packet cache entries is `2e6 * 850 * + 1e6 * 500 = about 2GB`.

11.5 Connection tracking and firewalls

A Recursor under high load puts a severe stress on any stateful (connection tracking) firewall, so much so that the firewall may fail.

Specifically, many Linux distributions run with a connection tracking firewall configured. For high load operation (thousands of queries/second), It is advised to either turn off iptables completely, or use the `NOTRACK` feature to make sure client DNS traffic bypasses the connection tracking.

Sample Linux command lines would be:

```
## IPv4
## NOTRACK rules for 53/udp, keep in mind that you also need your regular rules␣
↪for 53/tcp
iptables -t raw -I OUTPUT -p udp --sport 53 -j CT --notrack
iptables -t raw -I PREROUTING -p udp --dport 53 -j CT --notrack
iptables -I INPUT -p udp --dport 53 -j ACCEPT

## IPv6
## NOTRACK rules for 53/udp, keep in mind that you also need your regular rules␣
↪for 53/tcp
ip6tables -t raw -I OUTPUT -p udp --sport 53 -j CT --notrack
ip6tables -t raw -I PREROUTING -p udp --dport 53 -j CT --notrack
ip6tables -I INPUT -p udp --dport 53 -j ACCEPT
```

When using FirewallD (Centos 7+ / Red Hat 7+ / Fedora 21+), connection tracking can be disabled via direct rules. The settings can be made permanent by using the `--permanent` flag:

```
## IPv4
## NOTRACK rules for 53/udp, keep in mind that you also need your regular rules␣
↪for 53/tcp
firewall-cmd --direct --add-rule ipv4 raw OUTPUT 0 -p udp --sport 53 -j CT --
↪notrack
firewall-cmd --direct --add-rule ipv4 raw PREROUTING 0 -p udp --dport 53 -j CT --
↪notrack
firewall-cmd --direct --add-rule ipv4 filter INPUT 0 -p udp --dport 53 -j ACCEPT

## IPv6
## NOTRACK rules for 53/udp, keep in mind that you also need your regular rules␣
↪for 53/tcp
firewall-cmd --direct --add-rule ipv6 raw OUTPUT 0 -p udp --sport 53 -j CT --
↪notrack
firewall-cmd --direct --add-rule ipv6 raw PREROUTING 0 -p udp --dport 53 -j CT --
↪notrack
firewall-cmd --direct --add-rule ipv6 filter INPUT 0 -p udp --dport 53 -j ACCEPT
```

Following the instructions above, you should be able to attain very high query rates.

11.6 Tuning Incoming TCP and Out-of-Order processing

In general TCP uses more resources than UDP, so beware! It is impossible to give hard numbers for the various parameters as each site is different. Instead we describe the mechanism and relevant metrics so you can study your setup and change the proper settings if needed.

Each incoming TCP connection uses a file descriptor in addition to the file descriptors for other purposes, like contacting authoritative servers. When the recursor starts up, it will check if enough file descriptors are available and complain if not.

When a query is received over a TCP connection, first the packet cache is consulted. If an answer is found it will be returned immediately. If no answer is found, the Recursor will process *max-concurrent-requests-per-tcp-connection* queries per incoming TCP connection concurrently. If more than this number of queries is pending for this TCP connection, the remaining queries will stay in the TCP receive buffer to be processed later. Each of the queries processed will consume an mthread until processing is done. A response to a query is sent immediately when it becomes available; the response can be sent before other responses to queries that were received earlier by the Recursor. This is the Out-of-Order feature which greatly enhances performance, as a single slow query does not prevent other queries to be processed.

Before version 5.0.0, TCP queries are processed by either the distributer thread(s) if *pdns-distributes-queries* is true, or by worker threads if *pdns-distributes-queries* is false. Starting with version 5.0.0, **Recursor** has dedicated thread(s) processing TCP queries.

The maximum number of mthreads consumed by TCP queries is *max-tcp-clients* times *max-concurrent-requests-per-tcp-connection*. Before version 5.0.0, if *pdns-distributes-queries* is false, this number should be (much) lower than *max-mthreads*, to also allow UDP queries to be handled as these also consume mthreads. Note that *max-mthreads* is a per Posix thread setting. This means that the global maximum number of mthreads is (#distributor threads + #worker threads) * max-mthreads.

If you expect few clients, you can increase *max-concurrent-requests-per-tcp-connection*, to allow more concurrency per TCP connection. If you expect many clients and you have increased *max-tcp-clients*, reduce *max-concurrent-requests-per-tcp-connection* number to prevent mthread starvation or increase the maximum number of mthreads.

To increase the maximum number of concurrent queries consider increasing *max-mthreads*, but be aware that each active mthread consumes more than 200k of memory. To see the current number of mthreads in use consult the *concurrent-queries* metric. If a query could not be handled due to mthread shortage, the *over-capacity-drops* metric is increased.

As an example, if you have typically 200 TCP clients, and the default maximum number of mthreads of 2048, a good number of concurrent requests per TCP connection would be 5. Assuming a worst case packet cache hit ratio, if all 200 TCP clients fill their connections with queries, about half (5 * 200) of the mthreads would be used by incoming TCP queries, leaving the other half for incoming UDP queries. Note that starting with version 5.0.0, TCP queries are processed by dedicated TCP thread(s), so the sharing of mthreads between UDP and TCP queries no longer applies.

The total number of incoming TCP connections is limited by *max-tcp-clients*. There is also a per client address limit: *max-tcp-per-client* to limit the impact of a single client. Consult the *tcp-clients* metric for the current number of TCP connections and the *tcp-client-overflow* metric to see if client connection attempts were rejected because there were too many existing connections from a single address.

11.7 TCP Fast Open Support

On Linux systems, the recursor can use TCP Fast Open for passive (incoming, since 4.1) and active (outgoing, since 4.5) TCP connections. TCP Fast Open allows the initial SYN packet to carry data, saving one network round-trip. For details, consult **RFC 7413**.

On Linux systems, to enable TCP Fast Open, it might be needed to change the value of the `net.ipv4.tcp_fastopen` sysctl. Value 0 means Fast Open is disabled, 1 is only use Fast Open for active connections, 2 is only for passive connections and 3 is for both.

The operation of TCP Fast Open can be monitored by looking at these kernel metrics:

```
netstat -s | grep TCPFastOpen
```

Please note that if active (outgoing) TCP Fast Open attempts fail in particular ways, the Linux kernel stops using active TCP Fast Open for a while for all connections, even connection to servers that previously worked. This behaviour can be monitored by watching the `TCPFastOpenBlackHole` kernel metric and influenced by setting the `net.ipv4.tcp_fastopen_blackhole_timeout_sec` sysctl. While developing active TCP Fast Open, it was needed to set `net.ipv4.tcp_fastopen_blackhole_timeout_sec` to zero to circumvent the issue, since it was triggered regularly when connecting to authoritative nameservers that did not respond.

At the moment of writing, some Google operated nameservers (both recursive and authoritative) indicate Fast Open support in the TCP handshake, but do not accept the cookie they sent previously and send a new one for each connection. Google is working to fix this.

If you operate an anycast pool of machines, make them share the TCP Fast Open Key by setting the `net.ipv4.tcp_fastopen_key` sysctl, otherwise you will create a similar issue some Google servers have.

To determine a good value for the *tcp-fast-open* setting, watch the `TCPFastOpenListenOverflow` metric. If this value increases often, the value might be too low for your traffic, but note that increasing it will use kernel resources.

11.8 Running with a local root zone

Running with a local root zone as described in **RFC 8806** can help reduce traffic to the root servers and reduce response times for clients. Since 4.6.0 PowerDNS Recursor supports two ways of doing this.

Running a local Authoritative Server for the root zone

- The first method is to have a local Authoritative Server that has a copy of the root zone and forward queries to it. Setting up an PowerDNS Authoritative Server to serve a copy of the root zone looks like:

  pdnsutil create-secondary-zone . ip1 ip2

  where `ip1` and `ip2` are servers willing to serve an AXFR for the root zone; **RFC 8806** contains a list of candidates in appendix A. The Authoritative Server will periodically make sure its copy of the root zone is up-to-date. The next step is to configure a forward zone to the IP `ip` of the Authoritative Server in the settings file or the Recursor:

  forward-zones=.=ip

  The Recursor will use the Authoritative Server to ask questions about the root zone, but if it learns about delegations still follow those. Multiple Recursors can use this Authoritative Server.

- The second method is to cache the root zone as described in *Zone to Cache*. Here each Recursor will download and fill its cache with the contents of the root zone. Depending on the `timeout` parameter, this will be done once or periodically. Refer to *Zone to Cache* for details.

11.9 Recursor Caches

The PowerDNS Recursor contains a number of caches, or information stores:

11.9.1 Nameserver speeds cache

The "NSSpeeds" cache contains the average latency to all remote authoritative servers.

11.9.2 Negative cache

The "Negcache" contains all domains known not to exist, or record types not to exist for a domain.

11.9.3 Recursor Cache

The Recursor Cache contains all DNS knowledge gathered over time. This is also known as the "record cache".

11.9.4 Packet Cache

The Packet Cache contains previous answers sent to clients. If a question comes in that matches a previous answer, this is sent back directly.

The Packet Cache is consulted first, immediately after receiving a packet. This means that a high hitrate for the Packet Cache automatically lowers the cache hitrate of subsequent caches.

11.10 Measuring performance

The PowerDNS Recursor exposes many *metrics* that can be graphed and monitored.

11.11 Event Tracing

Event tracing is an experimental feature introduced in version 4.6.0 that allows following the internals of processing queries in more detail.

In certain spots in the resolving process event records are created that contain an identification of the event, a timestamp, potentially a value and an indication if this was the start or the end of an event. This is relevant for events that describe stages in the resolving process.

At this point in time event logs of queries can be exported using a protobuf log or they can be written to the log file.

Note that this is an experimental feature that will change in upcoming releases.

Currently, an event protobuf message has the following definition:

```
  enum EventType {
                                                  // Range 0..99: Generic events
    CustomEvent = 0;                              // A custom event
    ReqRecv = 1;                                  // A request was received
    PCacheCheck = 2;                              // A packet cache check was␣
↪initiated or completed; value: bool cacheHit
    AnswerSent = 3;                               // An answer was sent to the client

                                                  // Range 100: Recursor events
    SyncRes = 100;                                // Recursor Syncres main function␣
↪has started or completed; value: int rcode
    LuaGetTag = 101;                              // Events below mark start or end␣
↪of Lua hook calls; value: return value of hook
    LuaGetTagFFI = 102;
    LuaIPFilter = 103;
    LuaPreRPZ = 104;
    LuaPreResolve = 105;
    LuaPreOutQuery = 106;
    LuaPostResolve = 107;
    LuaNoData = 108;
    LuaNXDomain = 109;
}
```

```
message Event {
  required uint64 ts = 1;
  required EventType event = 2;
  required bool start = 3;
  optional bool boolVal = 4;
  optional int64 intVal = 5;
  optional string stringVal = 6;
  optional bytes bytesVal = 7;
  optional string custom = 8;
}
repeated Event trace = 23;
```

Event traces can be enabled by either setting *event-trace-enabled* or by using the *rec_control* subcommand `set-event-trace-enabled`.

An example of a trace (timestamps are relative in nanoseconds) as shown in the logfile:

```
- ReqRecv(70);
- PCacheCheck(411964);
- PCacheCheck(416783,0,done);
- SyncRes(441811);
- SyncRes(337233971,0,done);
 -AnswerSent(337266453)
```

The packet cache check event has two events. The first signals the start of packet cache lookup, and the second the completion of the packet cache lookup with result 0 (not found). The SynRec event also has two entries. The value (0) is the return value of the SyncRes function.

An example of a trace with a packet cache hit):

```
- ReqRecv(60);
- PCacheCheck(22913);
- PCacheCheck(113255,1,done);
- AnswerSent(117493)
```

Here it can be seen that packet cache returns 1 (found).

An example where various Lua related events can be seen:

```
ReqRecv(150);
PCacheCheck(26912);
PCacheCheck(51308,0,done);
LuaIPFilter(56868);
LuaIPFilter(57149,0,done);
LuaPreRPZ(82728);
LuaPreRPZ(82918,0,done);
LuaPreResolve(83479);
LuaPreResolve(210621,0,done);
SyncRes(217424);
LuaPreOutQuery(292868);
LuaPreOutQuery(292938,0,done);
LuaPreOutQuery(24702079);
LuaPreOutQuery(24702349,0,done);
LuaPreOutQuery(43055303);
LuaPreOutQuery(43055634,0,done);
SyncRes(80470320,0,done);
LuaPostResolve(80476592);
LuaPostResolve(80476772,0,done);
AnswerSent(80500247)
```

There is no packet cache hit, so SyncRes is called which does a couple of outgoing queries.

CHAPTER

TWELVE

MANUAL PAGES

12.1 pdns_recursor

12.1.1 Synopsis

pdns_recursor [*OPTION*]...

12.1.2 Description

pdns_recursor is a high performance, simple and secure recursing nameserver. It currently powers hundreds of millions internet connections.

The recursor is configured via a configuration file, but each item in that file can be overridden on the command line.

This manpage lists the core set of features needed to get the PowerDNS Recursor working, for full and up to date details head to https://doc.powerdns.com/.

12.1.3 Examples

To listen on 192.0.2.53 and allow the 192.0.2.0/24 subnet to recurse, and run as in the background, execute:

```
# pdns_recursor --local-address=192.0.2.53 --allow-from=192.0.2.0/24 --daemon
```

To stop the recursor by hand, run:

```
# rec_control quit
```

However, the recommended way of starting and stopping the recursor is to use `systemctl(1)` or the init.d script.

12.1.4 Options

For authoritative listing of options, consult the online documentation at <*https://doc.powerdns.com/*>

--allow-from=<networks> If set, only allow these comma separated *networks*, with network mask to recurse. For example: 192.0.2.0/24,203.0.113.128/25.

--auth-zones=<authzones> Where *authzone* is <zonename>=<filename>. Serve *zonename* from *filename* authoritatively. For example: ds9a.nl=/var/zones/ds9a.nl,powerdns.com=/var/zones/powerdns.com.

--chroot=<directory> chroot the process to *directory*.

--client-tcp-timeout=<num> Timeout in seconds when talking to TCP clients.

--config Show the current configuration. Since 4.8.0 there are three optional values: `--config=default` to show the default configuration. `--config=diff` show modified options in the current configuration. `--config=check` to check the current configuration for errors.

--config-dir=<directory> Location of configuration directory (recursor.conf), the default depends on the SYSCONFDIR option at build-time, which is usually /etc/powerdns. The default can be found with `pdns_recursor --config | grep ' config-dir='`.

--daemon Operate as a daemon.

--entropy-source=<file> Read new entropy from *file*, defaults to /dev/urandom.

--export-etc-hosts If set, this flag will export the hostnames and IP addresses mentioned in /etc/hosts.

--forward-zones=<forwardzones> Where *forwardzone* is <zonename>=<address>. Queries for *zonename* will be forwarded to *address*. *address* should be an IP address, not a hostname (to prevent chicken and egg problems). Example: forward-zones= ds9a.nl=213.244.168.210, powerdns.com=127.0.0.1.

--forward-zones-file=<filename> Similar to *–forward-zones*, but read the options from *filename*. *filename* should contain one zone per line, like: ds9a.nl=213.244.168.210.

--help Show a summary of options.

--hint-file=<filename> Load root hints from this *filename*

--local-address=<address> Listen on *address*, separated by spaces or commas. Addresses specified can include port numbers; any which do not include port numbers will listen on *–local-port*.

--local-port=<port> Listen on *port*.

--log-common-errors If we should log rather common errors.

--max-cache-entries=<num> Maximum number of entries in the main cache.

--max-negative-ttl=<num> maximum number of seconds to keep a negative cached entry in memory.

--max-tcp-clients=<num> Maximum number of simultaneous TCP clients.

--max-tcp-per-client=<num> If set, maximum number of TCP sessions per client (IP address).

--query-local-address=<address[,address...]> Use *address* as Source IP address when sending queries.

--quiet Suppress logging of questions and answers.

--server-id=<text> Return *text* WHen queried for 'id.server' TXT, defaults to hostname.

--serve-rfc1918 On by default, this makes the server authoritatively aware of: 10.in-addr.arpa, 168.192.in-addr.arpa and 16-31.172.in-addr.arpa, which saves load on the AS112 servers. Individual parts of these zones can still be loaded or forwarded.

--setgid=<gid> If set, change group id to *gid* for more security.

--setuid=<uid> If set, change user id to *uid* for more security.

--single-socket If set, only use a single socket for outgoing queries.

--socket-dir=<directory> The controlsocket will live in *directory*.

--spoof-nearmiss-max=<num> If non-zero, assume spoofing after this many near misses.

--trace if we should output heaps of logging.

--version-string=<text> *text* WILL be reported on version.pdns or version.bind queries.

12.1.5 See also

rec_control(1) *systemctl(1)* https://docs.powerdns.com/recursor

12.2 rec_control

12.2.1 Synopsis

rec_control [*OPTION*]... *COMMAND* [*COMMAND-OPTION*]...

12.2.2 Description

rec_control allows the operator to query and control a running instance of the PowerDNS Recursor.

rec_control talks to the recursor via a the 'controlsocket'. Which is usually located in `/var/run` . The *–socket-dir* or the *–config-dir* and *–config-name* switches control to which process **rec_control** connects.

12.2.3 Examples

To see if the Recursor is alive, run:

```
# rec_control ping
```

To stop the recursor by hand, run:

```
# rec_control quit
```

To dump the cache to disk, execute:

```
# rec_control dump-cache /tmp/the-cache
```

Note: Before version 4.5.0, for each command that writes to a file, **pdns_recursor** would open the file to write to. Starting with 4.5.0, the files are opened by the **rec_control** command itself using the credentials and the current working directory of the user running **rec_control**. A single minus - can be used as a filename to write the data to the standard output stream.

12.2.4 Options

--help provide this helpful message.

--config-dir=<path> Directory where the recursor.conf lives.

--config-name=<name> Name of the virtual configuration.

--socket-dir=<path> Where the controlsocket will live, please use **–config-dir** instead.

--socket-pid=<pid> When running in SMP mode, pid of **pdns_recursor** to control.

--timeout=<num> Number of seconds to wait for the remote PowerDNS Recursor to respond.

12.2.5 Commands

add-dont-throttle-names NAME [NAME...] Add names for nameserver domains that may not be throttled.

add-dont-throttle-netmasks NETMASK [NETMASK...] Add netmasks for nameservers that may not be throttled.

add-nta *DOMAIN* [*REASON*] Add a Negative Trust Anchor for *DOMAIN*, suffixed optionally with *REASON*.

add-ta *DOMAIN DSRECORD* Add a Trust Anchor for *DOMAIN* with DS record data *DSRECORD*. This adds the new Trust Anchor to the existing set of Trust Anchors for *DOMAIN*.

current-queries Shows the currently active queries.

clear-dont-throttle-names NAME [NAME...] Remove names that are not allowed to be throttled. If *NAME* is *, remove all

clear-dont-throttle-netmasks NETMASK [NETMASK...] Remove netmasks that are not allowed to be throttled. If *NETMASK* is *, remove all

clear-nta *DOMAIN*... Remove Negative Trust Anchor for one or more *DOMAIN*s. Set domain to * to remove all NTA's.

clear-ta [*DOMAIN*]... Remove Trust Anchor for one or more *DOMAIN*s. Note that removing the root trust anchor is not possible.

dump-cache *FILENAME* Dumps the entire cache to *FILENAME*. This file should not exist already, PowerDNS will refuse to overwrite it. While dumping, the recursor might not answer questions.

Typical PowerDNS Recursors run multiple threads, therefore you'll see duplicate, different entries for the same domains. The negative cache is also dumped to the same file. The per-thread positive and negative cache dumps are separated with an appropriate comment.

dump-dot-probe-map *FILENAME* Dump the contents of the DoT probe map to the *FILENAME* mentioned.

dump-edns *FILENAME* Dumps the EDNS status to the filename mentioned. This file should not exist already, PowerDNS will refuse to overwrite it. While dumping, the recursor will not answer questions.

dump-failedservers *FILENAME* Dump the contents of the failed server map to the *FILENAME* mentioned. This file should not exist already, PowerDNS will refuse to overwrite it otherwise. While dumping, the recursor will not answer questions.

dump-non-resolving *FILENAME* Dump the contents of the map of nameserver names that did not resolve to an address. This file should not exist already, PowerDNS will refuse to overwrite it otherwise. While dumping, the recursor will not answer questions.

dump-nsspeeds *FILENAME* Dumps the nameserver speed statistics to the *FILENAME* mentioned. This file should not exist already, PowerDNS will refuse to overwrite it. While dumping, the recursor will not answer questions. Statistics are kept per thread, and the dumps end up in the same file.

dump-rpz *ZONE NAME FILE NAME* Dumps the content of the RPZ zone named *ZONE NAME* to the *FILENAME* mentioned. This file should not exist already, PowerDNS will refuse to overwrite it otherwise. While dumping, the recursor will not answer questions. For details on how RPZ are named see https://docs.powerdns.com/recursor/lua-config/rpz.html#policyname.

dump-saved-parent-ns-sets *FILE NAME* Dump the entries of the map containing saved parent NS sets that were successfully used in resolving. The total number of entries is also printed in the header. An entry is saved if the recursor sees that the parent set includes names not in the child set. This is an indication of a misconfigured domain.

dump-throttlemap *FILENAME* Dump the contents of the throttle map to the *FILENAME* mentioned. This file should not exist already, PowerDNS will refuse to overwrite it otherwise. While dumping, the recursor will not answer questions.

get *STATISTIC* [*STATISTIC*]... Retrieve a statistic. For items that can be queried, see https://docs.powerdns.com/recursor/metrics.html.

get-all Retrieve all known statistics.

get-dont-throttle-names Get the list of names that are not allowed to be throttled.

get-dont-throttle-netmasks Get the list of netmasks that are not allowed to be throttled.

get-ntas Get a list of the currently configured Negative Trust Anchors.

get-tas Get a list of the currently configured Trust Anchors.

get-parameter *KEY* [*KEY*]... Retrieves the specified configuration parameter(s).

get-proxymapping-stats Get the list of proxy-mapped subnets and associated counters.

get-qtypelist Retrieves QType statistics. Queries from cache aren't being counted yet.

get-remotelogger-stats Retrieves the remote logger statistics, per type and address.

hash-password [*WORK-FACTOR*] Asks for a password then returns the hashed and salted version, to use as a webserver password or API key. This command does not contact the recursor but does the hashing inside rec_control. An optional scrypt work factor can be specified, in power of two. The default is 1024.

help Shows a list of supported commands understood by the running **`pdns_recursor`**

list-dnssec-algos List supported (and potentially disabled) DNSSEC algorithms.

ping Check if server is alive.

quit Request shutdown of the recursor, exiting the process while letting the OS clean up resources.

quit-nicely Request nice shutdown of the recursor. This method allows all threads to finish their current work and releases resources before exiting. This is the preferred method to stop the recursor.

reload-acls Reloads ACLs.

reload-lua-script [*FILENAME*] (Re)loads Lua script *FILENAME*. If *FILENAME* is empty, attempt to reload the currently loaded script. This replaces the script currently loaded.

reload-lua-config [*FILENAME*] (Re)loads Lua configuration *FILENAME*. If *FILENAME* is empty, attempt to reload the currently loaded file. Note that *FILENAME* will be fully executed, any settings changed at runtime that are not modified in this file, will still be active. The effects of reloading do not always take place immediately, as some subsystems reload and replace configuration in an asynchronous way.

reload-zones Reload authoritative and forward zones. Retains current configuration in case of errors.

set-carbon-server *CARBON SERVER* [*CARBON OURNAME*] Set the carbon-server setting to *CARBON SERVER*. If *CARBON OURNAME* is not empty, also set the carbon-ourname setting to *CARBON OURNAME*.

set-dnssec-log-bogus *SETTING* Set dnssec-log-bogus setting to *SETTING*. Set to `on` or `yes` to log DNSSEC validation failures and to `no` or `off` to disable logging these failures.

set-ecs-minimum-ttl *NUM* Set ecs-minimum-ttl-override to *NUM*.

set-max-aggr-nsec-cache-size *NUM* Change the maximum number of entries in the NSEC aggressive cache. If the cache is disabled by setting its size to 0 in the config, the cache size cannot be set by this command. Setting the size to 0 by this command still keeps the cache, but makes it mostly ineffective as it is emptied periodically.

set-max-cache-entries *NUM* Change the maximum number of entries in the DNS cache. If reduced, the cache size will start shrinking to this number as part of the normal cache purging process, which might take a while.

set-max-packetcache-entries *NUM* Change the maximum number of entries in the packet cache. If reduced, the cache size will start shrinking to this number as part of the normal cache purging process, which might take a while.

set-minimum-ttl *NUM* Set minimum-ttl-override to *NUM*.

set-event-trace-enabled *NUM* Set logging of event trace messages, `0` = disabled, `1` = protobuf, `2` = log file, `3` = protobuf and log file.

show-yaml [*FILE*] Show Yaml representation of odl-style config.

top-queries Shows the top-20 queries. Statistics are over the last 'stats-ringbuffer-entries' queries.

top-pub-queries Shows the top-20 queries grouped by public suffix list. Statistics are over the last 'stats-ringbuffer-entries' queries.

top-largeanswer-remotes Shows the top-20 remote hosts causing large answers. Statistics are over the last 'stats-ringbuffer-entries' queries.

top-remotes Shows the top-20 most active remote hosts. Statistics are over the last 'stats-ringbuffer-entries' queries.

top-servfail-queries Shows the top-20 queries causing servfail responses. Statistics are over the last 'stats-ringbuffer-entries' queries.

top-bogus-queries Shows the top-20 queries causing bogus responses. Statistics are over the last 'stats-ringbuffer-entries' queries.

top-pub-servfail-queries Shows the top-20 queries causing servfail responses grouped by public suffix list. Statistics are over the last 'stats-ringbuffer-entries' queries.

top-pub-bogus-queries Shows the top-20 queries causing bogus responses grouped by public suffix list. Statistics are over the last 'stats-ringbuffer-entries' queries.

top-servfail-remotes Shows the top-20 most active remote hosts causing servfail responses. Statistics are over the last 'stats-ringbuffer-entries' queries.

top-bogus-remotes Shows the top-20 most active remote hosts causing bogus responses. Statistics are over the last 'stats-ringbuffer-entries' queries.

top-timeouts Shows the top-20 most active downstream timeout destinations. Statistics are over the last 'stats-ringbuffer-entries' queries.

trace-regex ***REGEX FILE*** Emit resolution trace for matching queries. No arguments disables tracing. Before version 4.9.0, there was no *FILE* argument, traces were always written to the log. Starting with version 4.9.0, trace information is written to the file specified, which may be − for the standard out stream.

Queries matching this regular expression will generate voluminous tracing output. Be aware that matches from the packet cache will still not generate tracing. To unset the regex, pass **trace-regex** without a new regex.

The regular expression is matched against domain queries terminated with a dot. For example the regex `'powerdns.com$'` will not match a query for `'www.powerdns.com'`, since the attempted match will be with `'www.powerdns.com.'`.

In addition, since this is a regular expression, to exclusively match queries for `'www.powerdns.com'`, one should escape the dots: `'^www\.powerdns\.com\.$'`. Note that the single quotes prevent further interpretation of the backslashes by the shell.

Multiple matches can be chained with the `|` operator. For example, to match all queries for Dutch (`.nl`) and German (`.de`) domain names, use: `'\.nl\.$|\.de\.$'`.

unload-lua-script Unloads Lua script if one was loaded.

version Report running version.

wipe-cache ***DOMAIN*** **[*DOMAIN*] [...]** Wipe entries for *DOMAIN* (exact name match) from the cache. This is useful if, for example, an important server has a new IP address, but the TTL has not yet expired. Multiple domain names can be passed. *DOMAIN* can be suffixed with a `$.` to delete the whole tree from the cache. i.e. `powerdns.com$` will remove all cached entries under and including the powerdns.com name.

Note: this command also wipes the negative cache.

Warning: Don't just wipe "www.somedomain.com", its NS records or CNAME target may still be undesired, so wipe "somedomain.com" as well.

wipe-cache-typed ***qtype DOMAIN*** **[*DOMAIN*] [...]** Same as wipe-cache, but only wipe records of type *qtype*.

12.2.6 See also

pdns_recursor(1) https://docs.powerdns.com/recursor

CHAPTER

THIRTEEN

BUILT-IN WEBSERVER AND HTTP API

The PowerDNS Recursor features a built-in built-in webserver that exposes a JSON/REST API. This API allows for controlling several functions and reading statistics.

The following documents contain the information for the PowerDNS API:

13.1 Data format

The API accepts and emits JSON. The `Accept:` header determines the output format. An unknown value or `*/*` will cause a `400 Bad Request`.

All text is UTF-8 and HTTP headers will reflect this.

Data types:

- empty fields: `null` but present
- Regex: implementation defined
- Dates: ISO 8601

13.1.1 General Collections Interface

Collections generally support `GET` and `POST` with these meanings:

GET

Retrieve a list of all entries.

The special `type` and `url` fields are included in the response objects:

- `type`: name of the resource type
- `url`: url to the object

Response format:

```
[
  obj1
  [, further objs]
]
```

Example:

```
[
  {
    "type": "AType",
    "id": "anid",
```

(continues on next page)

(continued from previous page)

```
    "url": "/atype/anid",
    "a_field": "a_value"
  },
  {
    "type": "AType",
    "id": "anotherid",
    "url": "/atype/anotherid",
    "a_field": "another_value"
  }
]
```

POST

Create a new entry. The client has to supply the entry in the request body, in JSON format. `application/x-www-form-urlencoded` data MUST NOT be sent.

Clients SHOULD not send the 'url' field.

Client body:

```
obj1
```

Example:

```
{
  "type": "AType",
  "id": "anewid",
  "a_field": "anew_value"
}
```

13.1.2 REST

- GET: List/Retrieve. Success reply: `200 OK`
- POST: Create. Success reply: `201 Created`, with new object as body.
- PUT: Update. Success reply: `200 OK`, with modified object as body. For some operations, `204 No Content` is returned instead (and the modified object is not given in the body).
- DELETE: Delete. Success reply: `200 OK`, no body.

13.1.3 not-so-REST

For interactions that do not directly map onto CRUD, we use these:

- GET: Query. Success reply: `200 OK`
- PUT: Action/Execute. Success reply: `200 OK`

Action/Execute methods return a JSON body of this format:

```
{
  "message": "result message"
}
```

13.1.4 Authentication

The PowerDNS daemons accept a static API Key, configured with the *api-key* option, which has to be sent in the `X-API-Key` header.

13.1.5 Errors

Response code `4xx` or `5xx`, depending on the situation. Never return `2xx` for an error!

- Invalid JSON body from client: `400 Bad Request`
- JSON body from client not a hash: `400 Bad Request`
- Input validation failed: `422 Unprocessable Entity`

Error responses have a JSON body of this format:

```
{
  "error": "short error message",
  "errors": [
    {  },
  ]
}
```

Where `errors` is optional, and the contents are error-specific.

Common Error Causes

400 Bad Request

1. The client body was not a JSON document, or it could not be parsed, or the root element of the JSON document was not a hash.
2. The client did not send an `Accept:` header, or it was set to `*/*`.
3. For requests that operate on a zone, the `zone_id` URL part was invalid. To get a valid `zone_id`, list the zones with the `/api/v1/servers/:server_id/zones` endpoint.

13.2 Server

`Server`

An object representing a single PowerDNS server. In the built-in API, only one Server exists (called "localhost").

A proxy that allows control of multiple servers MUST NOT return `localhost`, but SHOULD return other servers.

Object Properties

- **`type`** (*`string`*) – Set to "Server"
- **`id`** (*`string`*) – The id of the server, "localhost"
- **`daemon_type`** (*`string`*) – "recursor" for the PowerDNS Recursor and "authoritative" for the Authoritative Server
- **`version`** (*`string`*) – The version of the server software
- **`url`** (*`string`*) – The API endpoint for this server
- **`config_url`** (*`string`*) – The API endpoint for this server's configuration
- **`zones_url`** (*`string`*) – The API endpoint for this server's zones

Example:

```
{
  "type": "Server",
  "id": "localhost",
  "url": "/api/v1/servers/localhost",
  "daemon_type": "recursor",
  "version": "4.1.0",
  "config_url": "/api/v1/servers/localhost/config{/config_setting}",
  "zones_url": "/api/v1/servers/localhost/zones{/zone}",
}
```

Note: the servers collection is read-only, and the only allowed returned server is read-only as well. A control proxy could return modifiable resources.

13.3 Zones

13.3.1 Zone

A Zone object represents a forward or authoritative DNS Zone.

A Resource Record Set (below as "RRset") are all records for a given name and type.

Comments are per-RRset.

Zone

Represents a configured zone in the PowerDNS server.

Object Properties

- **id** (*string*) – Opaque zone id (string), assigned by the server, should not be interpreted by the application. Guaranteed to be safe for embedding in URLs.
- **name** (*string*) – Name of the zone (e.g. "example.com.") MUST have a trailing dot
- **type** (*string*) – Set to "Zone"
- **url** (*string*) – API endpoint for this zone
- **kind** (*string*) – Zone kind, one of "Native", "Forwarded".
- **rrsets** (*[RRSet]*) – RRSets in this zone
- **servers** (*[str]*) – For zones of type "Forwarded", addresses to send the queries to
- **recursion_desired** (*bool*) – For zones of type "Forwarded", Whether or not the RD bit should be set in the query
- **notify_allowed** (*bool*) – For zones of type "Forwarded", Whether or not to permit incoming NOTIFY to wipe cache for the domain

To properly process new zones, the following conditions must be true:

- `forward-zones`, `forward-zones-recurse` and/or `auth-zones` settings must be set (possibly to the empty string) in a configuration file. These settings must not be overridden on the command line. Setting these options on the command line will override what has been set in the dynamically generated configuration files.
- For configuration changes to work *include-dir* and *api-config-dir* should have the same value for old-style settings. When using YAML settings *recursor.include_dir* and *webservice.api_dir* must have a different value.

13.3.2 RRSet

RRSet

This represents a Resource Record set (all record with the same name and type).

Object Properties

- **name** (*string*) – Name for record set (e.g. "www.powerdns.com.")
- **type** (*string*) – Type of this record (e.g. "A", "PTR", "MX")
- **ttl** (*integer*) – DNS TTL of the records, in seconds. MUST NOT be included when `changetype` is set to "DELETE".
- **changetype** (*string*) – MUST be added when updating the RRSet. Must be `REPLACE` or `DELETE`. With `DELETE`, all existing RRs matching `name` and `type` will be deleted, including all comments. With `REPLACE`: when `records` is present, all existing RRs matching `name` and `type` will be deleted, and then new records given in `records` will be created. If no records are left, any existing comments will be deleted as well. When `comments` is present, all existing comments for the RRs matching `name` and `type` will be deleted, and then new comments given in `comments` will be created.
- **records** (*[RREntry]*) – All records in this RRSet. When updating records, this is the list of new records (replacing the old ones). Must be empty when `changetype` is set to `DELETE`. An empty list results in deletion of all records (and comments).
- **comments** (*[Comment]*) – List of *Comment*. Must be empty when `changetype` is set to `DELETE`. An empty list results in deletion of all comments. `modified_at` is optional and defaults to the current server time.

13.3.3 RREntry

RREntry

The RREntry object represents a single record in an *RRSet*.

Object Properties

- **content** (*string*) – The content of this record
- **disabled** (*bool*) – Whether or not this record is disabled
- **set-ptr** (*bool*) – If set to true, the server will find the matching reverse zone and create a PTR there. Existing PTR records are replaced. If no matching reverse *Zone*, an error is thrown. Only valid in client bodies, only valid for A and AAAA types. Not returned by the server. This feature (set-ptr) has been removed in 4.4.0.

13.3.4 Comment

Comment

Object Properties

- **content** (*string*) – The actual comment
- **account** (*string*) – Name of an account that added the comment
- **modified_at** (*integer*) – Timestamp of the last change to the comment

13.4 ConfigSetting

ConfigSetting

Represents a configuration item (as found in *PowerDNS Recursor Settings*)

Object Properties

- **type** (*string*) – set to "ConfigSetting"

- **name** (*string*) – The name of this setting (e.g. 'webserver-port')
- **value** (*string*) – The value of setting `name`

Example:

```
{
  "name": "webserver-port",
  "type": "ConfigSetting",
  "value": "8081"
}
```

13.5 StatisticItem

StatisticItem

Represents a single statistic item (as found in *Gathered Information*)

Object Properties

- **type** (*string*) – set to "StatisticItem"
- **name** (*string*) – The name of this item
- **value** (*string*) – The value of this item

13.6 Webserver

To launch the internal webserver, add a *webserver* to the configuration file. This will instruct PowerDNS to start a webserver on localhost at port 8081, without password protection. By default the webserver listens on localhost, meaning only local users (on the same host) will be able to access the webserver. Since the default ACL before 4.1.0 allows access from everywhere if *webserver-address* is set to a different value, we strongly advise the use of a password protection. The webserver lists a lot of potentially sensitive information about the PowerDNS process, including frequent queries, frequently failing queries, lists of remote hosts sending queries, hosts sending corrupt queries etc. The webserver does not allow remote management. The following webserver related configuration items are available:

- *webserver*: If set to anything but 'no', a webserver is launched.
- *webserver-address*: Address to bind the webserver to. Defaults to 127.0.0.1, which implies that only the local computer is able to connect to the nameserver! To allow remote hosts to connect, change to 0.0.0.0 or the physical IP address of your nameserver.
- *webserver-password*: If set, viewers will have to enter this password in order to gain access to the statistics.
- *webserver-port*: Port to bind the webserver to.
- *webserver-allow-from*: Netmasks that are allowed to connect to the webserver

13.7 Enabling the API

To enable the API, the webserver and the HTTP API need to be enabled. Add these lines to the `recursor.conf`:

```
webserver=yes
webserver-port=8082
api-key=changeme
```

And restart `pdns_recursor`, the following examples should start working:

```
curl -v -H 'X-API-Key: changeme' http://127.0.0.1:8082/api/v1/servers/localhost |␣
↪jq .
curl -v -H 'X-API-Key: changeme' http://127.0.0.1:8082/api/v1/servers/localhost/
↪zones | jq .
```

A few examples for zone manipulation follow, first one is to create a forwarding zone:

```
curl --no-progress-meter -H 'X-API-Key: changeme' -H 'Content-type: application/
↪json' -X POST --data-binary @- http://localhost:8082/api/v1/servers/localhost/
↪zones << EOF | jq
{
  "name": "example.com.",
  "type": "Zone",
  "kind": "Forwarded",
  "servers": ["192.168.178.1", "192.168.178.2:5353"],
  "recursion_desired" : false
}
EOF
```

Example output of the above command:

```
{
  "id": "example.com.",
  "kind": "Forwarded",
  "name": "example.com.",
  "records": [],
  "recursion_desired": false,
  "servers": [
    "192.168.178.1:53",
    "192.168.178.2:5353"
  ],
  "url": "/api/v1/servers/localhost/zones/example.com."
}
```

To delete the forwarding zone added above:

```
curl --no-progress-meter -H 'X-API-Key: changeme' -X DELETE http://localhost:8082/
↪api/v1/servers/localhost/zones/example.com.
```

13.8 URL Endpoints

All API endpoints for the PowerDNS Recursor are documented here:

13.8.1 Prometheus Data Endpoint

New in version 4.3.0.

GET /metrics

Get statistics from Recursor in Prometheus format. Uses *webserver-password* and returned list can be controlled with *stats-api-blacklist*

Example request:

```
curl -i -u=#:webpassword http://127.0.0.1:8081/metrics
```

Example response:

```
HTTP/1.1 200 OK
Connection: close
Content-Length: 19203
Content-Type: text/plain
Server: PowerDNS/0.0.16480.0.g876dd46192

# HELP pdns_recursor_all_outqueries Number of outgoing UDP queries since␣
↪starting
# TYPE pdns_recursor_all_outqueries counter
pdns_recursor_all_outqueries 20
# HELP pdns_recursor_answers_slow Number of queries answered after 1 second
# TYPE pdns_recursor_answers_slow counter
pdns_recursor_answers_slow 0
# HELP pdns_recursor_answers0_1 Number of queries answered within 1␣
↪millisecond
# TYPE pdns_recursor_answers0_1 counter
pdns_recursor_answers0_1 0
# HELP pdns_recursor_answers1_10 Number of queries answered within 10␣
↪milliseconds
# TYPE pdns_recursor_answers1_10 counter

...
```

13.8.2 API root endpoints

GET /api

Version discovery endpoint.

Example response:

```
[
  {
    "url": "/api/v1",
    "version": 1
  }
]
```

GET /api/v1

APIv1 root endpoint. Gives some information about the current API.

Not yet implemented:

- `api_features`
- `servers_modifiable`
- `oauth`

Example response:

```
{
  "server_url": "/api/v1/servers{/server}",
  "api_features": []
}
```

13.8.3 Server endpoint

GET /api/v1/servers

Server collection access.

GET /api/v1/servers/:server_id

Returns a single *Server*

Parameters

- **server_id** – The name of the server.

13.8.4 Configuration endpoint

GET /api/v1/servers/:server_id/config
Returns all *ConfigSetting* for a single server

Parameters

- **server_id** – The name of the server

POST /api/v1/servers/:server_id/config

Note: Not implemented

Creates a new config setting. This is useful for creating configuration for new backends.

Parameters

- **server_id** – The name of the server

GET /api/v1/servers/:server_id/config/:config_setting_name
Retrieve a single setting

Parameters

- **server_id** – The name of the server
- **config_setting_name** – The name of the setting to retrieve

PUT /api/v1/servers/:server_id/config/:config_setting_name
Change a single setting

Note: Only *allow-from* and *allow-notify-from* can be set.

Note: For configuration changes to work *include-dir* and *api-config-dir* should have the same value for old-style settings. When using YAML settings *recursor.include_dir* and *webservice.api_dir* must have a different value.

Parameters

- **server_id** – The name of the server
- **config_setting_name** – The name of the setting to change

Example request

```
PUT /api/v1/servers/localhost/config/allow-from HTTP/1.1
Host: localhost:8082
User-Agent: curl/7.54.1
Accept: application/json
X-Api-Key: secret
Content-Type: application/json
Content-Length: 48

{ "name": "allow-from", "value": ["127.0.0.0/8"] }
```

Example response

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
Connection: close
Content-Length: 48
Content-Security-Policy: default-src 'self'; style-src 'self' 'unsafe-inline'
Content-Type: application/json
Server: PowerDNS/0.0.g00799130f
X-Content-Type-Options: nosniff
X-Frame-Options: deny
X-Permitted-Cross-Domain-Policies: none
X-Xss-Protection: 1; mode=block

{"name": "allow-from", "value": ["127.0.0.0/8"]}
```

13.8.5 Statistics endpoint

GET /api/v1/servers/:server_id/statistics?statistic=:statistic

Query PowerDNS internal statistics. Returns a list of *StatisticItem* elements.

The *metricnames* page describes the names and meanings of these items.

Parameters

- **server_id** – The name of the server

Query Parameters

- **statistic** – If set to the name of a specific statistic, only this value is returned. If no statistic with that name exists, the response has a 422 status and an error message

Example response:

```
[{"name": "all-outqueries", "type": "StatisticItem", "value": "341"}, {"name":
↪"answers-slow", "type": "StatisticItem", "value": "0"}, {"name": "answers0-1
↪", "type": "StatisticItem", "value": "0"}, {"name": "answers1-10", "type":
↪"StatisticItem", "value": "0"}, {"name": "answers10-100", "type":
↪"StatisticItem", "value": "0"}, {"name": "answers100-1000", "type":
↪"StatisticItem", "value": "0"}, {"name": "auth4-answers-slow", "type":
↪"StatisticItem", "value": "200"}, {"name": "auth4-answers0-1", "type":
↪"StatisticItem", "value": "13"}, {"name": "auth4-answers1-10", "type":
↪"StatisticItem", "value": "1"}, {"name": "auth4-answers10-100", "type":
↪"StatisticItem", "value": "68"}, {"name": "auth4-answers100-1000", "type":
↪"StatisticItem", "value": "19"}, {"name": "auth6-answers-slow", "type":
↪"StatisticItem", "value": "0"}, {"name": "auth6-answers0-1", "type":
↪"StatisticItem", "value": "0"}, {"name": "auth6-answers1-10", "type":
↪"StatisticItem", "value": "0"}, {"name": "auth6-answers10-100", "type":
↪"StatisticItem", "value": "0"}, {"name": "auth6-answers100-1000", "type":
↪"StatisticItem", "value": "0"}, {"name": "cache-entries", "type":
↪"StatisticItem", "value": "124"}, {"name": "cache-hits", "type":
↪"StatisticItem", "value": "0"}, {"name": "cache-misses", "type":
↪"StatisticItem", "value": "0"}, {"name": "case-mismatches", "type":
↪"StatisticItem", "value": "0"}, {"name": "chain-resends", "type":
↪"StatisticItem", "value": "0"}, {"name": "client-parse-errors", "type":
↪"StatisticItem", "value": "0"}, {"name": "concurrent-queries", "type":
↪"StatisticItem", "value": "1"}, {"name": "dlg-only-drops", "type":
↪"StatisticItem", "value": "0"}, {"name": "dnssec-queries", "type":
↪"StatisticItem", "value": "0"}, {"name": "dnssec-result-bogus", "type":
↪"StatisticItem", "value": "0"}, {"name": "dnssec-result-indeterminate", "type
↪": "StatisticItem", "value": "0"}, {"name": "dnssec-result-insecure", "type
↪": "StatisticItem", "value": "0"}, {"name": "dnssec-result-nta", "type":
↪"StatisticItem", "value": "0"}, {"name": "dnssec-result-secure", "type":
↪"StatisticItem", "value": "9"}, {"name": "dnssec-validations", "type":
↪"StatisticItem", "value": "9"}, {"name": "dont-outqueries", "type":
↪"StatisticItem", "value": "0"}, {"name": "edns-ping-matches", "type":
↪"StatisticItem", "value": "0"}, {"name": "edns-ping-mismatches", "type":
↪"StatisticItem", "value": "0"}, {"name": "failed-host-entries", "type":
↪"StatisticItem", "value": "0"}, {"name": "fd-usage", "type": "StatisticItem",
↪ "value": "25"}, {"name": "ignored-packets", "type": "StatisticItem", "value
↪": "0"}, {"name": "ipv6-outqueries", "type": "StatisticItem", "value": "0"},
↪{"name": "ipv6-questions", "type": "StatisticItem", "value": "0"}, {"name":
```

(continues on next page)

(continued from previous page)

13.8.6 Zones endpoint

GET /api/v1/servers/:server_id/zones
Get all zones from the server.

Query Parameters

- **server_id** – The name of the server

POST /api/v1/servers/:server_id/zones
Creates a new domain. The client body must contain a *Zone*.

Query Parameters

- **server_id** – The name of the server

GET /api/v1/servers/:server_id/zones/:zone_id
Returns zone information.

Query Parameters

- **server_id** – The name of the server
- **zone_id** – The id number of the *Zone*

DELETE /api/v1/servers/:server_id/zones/:zone_id
Deletes this zone, all attached metadata and rrsets.

Query Parameters

- **server_id** – The name of the server
- **zone_id** – The id number of the *Zone*

13.8.7 Query Tracing endpoint

Note: Not yet implemented

PUT /api/v1/servers/:server_id/trace
Configure query tracing.

Query Parameters

- **server_id** – The name of the server

Client body:

```
{
  "domains": "<regex_string>"
}
```

Set `domains` to `null` to turn off tracing.

GET /api/v1/servers/:server_id/trace
Retrieve query tracing log and current config.

Query Parameters

- **server_id** – The name of the server

Response Body:

```
{
  "domains": "<Regex>",
  "log": [
    "<log_line>"
  ]
}
```

13.8.8 Cache manipulation endpoint

PUT /api/v1/servers/:server_id/cache/flush?domain=:domain
Flush the positive, negative and packet cache for a given domain name.

Query Parameters

- **server_id** – The name of the server
- **domain** – The domainname to flush for

New in version 4.1.3.

Query Parameters

- **subtree** – If set to *true*, also flush the whole subtree (default = *false*)

New in version 4.4.0.

Query Parameters

- **type** – If set the recursor only flushes records of the specified type name.

Example Response:

```
{
  "count": 10,
  "result": "Flushed cache."
}
```

13.8.9 Failure logging endpoint

Note: Not yet implemented

PUT /api/v1/servers/:server_id/failure
Configure query failure logging.

Query Parameters

- **server_id** – The name of the server

Example client body:

```
{
  "top-domains": 100,
  "domains": ".*\\.example\\.com$"
}
```

Property int top-domains Number of top resolved domains that are automatically monitored for failures.

Property string domains A Regex of domains that are additionally monitored for resolve failures.

GET /api/v1/servers/:server_id/failure

> **Note:** Not yet implemented

Retrieve query failure logging and current config.

Example response body:

```
{
  "top-domains": 100,
  "domains": ".*\\.example\\.com$",
  "log": [
    {
      "first_occurred": 1234567890,
      "domain": "www.example.net",
      "qtype": "A",
      "failure": "dnssec-parent-validation-failed",
      "failed_parent": "example.com",
      "details": "foo bar",
      "queried_servers": [
        {
          "name": "ns1.example.net",
          "address": "192.0.2.53"
        }
      ]
    }
  ]
}
```

Property string failed_parent The parent domain, this is generally OPTIONAL.

Property string failure_code Reason of failure.

- `dnssec-validation-failed`: DNSSEC Validation failed for this domain.
- `dnssec-parent-validation-failed`: DNSSEC Validation failed for one of the parent domains. Response MUST contain `failed_parent`.
- `nxdomain`: This domain was not present on the authoritative nameservers.
- `nodata`: ???
- `all-servers-unreachable`: All auth nameservers that have been tried did not respond.
- `parent-unresolvable`: Response MUST contain `failed_parent`.
- `refused`: All auth nameservers that have been tried responded with REFUSED.
- `servfail`: All auth nameservers that have been tried responded with SERVFAIL.

Property string domain The domain queried

13.8.10 RPZ Statistics endpoint

New in version 4.1.2.

GET /api/v1/servers/:server_id/rpzstatistics

Query PowerDNS for *Response Policy Zones* statistics.

Statistics are mapped per configured RPZ zone. The statistics are:

last_update UNIX timestamp when the latest update was received

records Number of records in the RPZ

serial Current SOA serial of the RPZ zone

transfers_failed Number of times a transfer failed

transfers_full Number of times an AXFR succeeded

transfers_success Number of times an AXFR or IXFR succeeded

Example response:

```
{
  "myRPZ": {
    "last_update": 1521798212,
    "records": 1343149,
    "serial": 5489,
    "transfers_failed": 0,
    "transfers_full": 3,
    "transfers_success": 478
  }
}
```

13.8.11 jsonstat endpoint

GET /jsonstat

Get statistics from recursor in JSON format. The `Accept` request header is ignored. This endpoint accepts a `command` and `name` query for different statistics:

- `get-query-ring`: Retrieve statistics from the query subsection. `name` can be `servfail-queries` or `queries`. Supports optional argument `public-filtered` which if set to any value will group queries by the public suffix list.
- `get-remote-ring`: Retrieve statistics from the remotes subsection. `name` can be `remotes`, `servfail-remotes`, `bogus-remotes` (added in 4.2.0), `large-answer-remotes`, or `timeouts` (added in 4.2.0).

Example request:

```
GET /jsonstat?command=get-query-ring&name=servfail-queries HTTP/1.1
Host: example.com
Accept: application/json, text/javascript
X-API-Key: examplekey
```

Example response:

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
Connection: close
Content-Length: 94
Content-Security-Policy: default-src 'self'; style-src 'self' 'unsafe-
↪inline'
Content-Type: application/json
Server: PowerDNS/4.1.11
X-Content-Type-Options: nosniff
X-Frame-Options: deny
X-Permitted-Cross-Domain-Policies: none
X-Xss-Protection: 1; mode=block

{"entries": [[2, "wpad.americas.hpecorp.net", "A"], [1, "wpad.americas.
↪hpecorp.net", "AAAA"]]}
```

Example request:

```
GET /jsonstat?command=get-query-ring&name=queries HTTP/1.1
Host: example.com
Accept: application/json, text/javascript
X-API-Key: examplekey
```

Example response:

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
Connection: close
Content-Length: 69
Content-Security-Policy: default-src 'self'; style-src 'self' 'unsafe-
↪inline'
Content-Type: application/json
Server: PowerDNS/4.1.11
X-Content-Type-Options: nosniff
X-Frame-Options: deny
X-Permitted-Cross-Domain-Policies: none
X-Xss-Protection: 1; mode=block

{"entries": [[1, "a.powerdns.com", "A"], [1, "b.powerdns.com", "A"]]}
```

Example request:

```
GET /jsonstat?command=get-query-ring&name=queries&public-filtered=true␣
↪HTTP/1.1
Host: example.com
Accept: application/json, text/javascript
X-API-Key: examplekey
```

Example response:

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
Connection: close
Content-Length: 39
Content-Security-Policy: default-src 'self'; style-src 'self' 'unsafe-
↪inline'
Content-Type: application/json
Server: PowerDNS/4.1.11
X-Content-Type-Options: nosniff
X-Frame-Options: deny
X-Permitted-Cross-Domain-Policies: none
X-Xss-Protection: 1; mode=block

{"entries": [[2, "powerdns.com", "A"]]}
```

Example request:

```
GET /jsonstat?command=get-remote-ring&name=remotes HTTP/1.1
Host: example.com
Accept: application/json, text/javascript
X-API-Key: examplekey
```

Example response:

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
Connection: close
Content-Length: 62
Content-Security-Policy: default-src 'self'; style-src 'self' 'unsafe-
↪inline'
```

(continues on next page)

(continued from previous page)

```
Content-Type: application/json
Server: PowerDNS/4.1.11
X-Content-Type-Options: nosniff
X-Frame-Options: deny
X-Permitted-Cross-Domain-Policies: none
X-Xss-Protection: 1; mode=block

{"entries": [[11, "10.0.2.15"], [7, "::1"], [4, "127.0.0.1"]]}
```

Example response:

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
Connection: close
Content-Length: 43
Content-Security-Policy: default-src 'self'; style-src 'self' 'unsafe-
↪inline'
Content-Type: application/json
Server: PowerDNS/4.1.11
X-Content-Type-Options: nosniff
X-Frame-Options: deny
X-Permitted-Cross-Domain-Policies: none
X-Xss-Protection: 1; mode=block

{"entries": [[2, "::1"], [1, "127.0.0.1"]]}
```

Example request:

```
GET /jsonstat?command=get-remote-ring&name=bogus-remotes HTTP/1.1
Host: example.com
Accept: application/json, text/javascript
X-API-Key: examplekey
```

Example response:

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
Connection: close
Content-Length: 32
Content-Security-Policy: default-src 'self'; style-src 'self' 'unsafe-
↪inline'
Content-Type: application/json
Server: PowerDNS/4.2.0-alpha1
X-Content-Type-Options: nosniff
X-Frame-Options: deny
X-Permitted-Cross-Domain-Policies: none
X-Xss-Protection: 1; mode=block

{"entries": [[20, "127.0.0.1"]]}
```

Example request:

```
GET /jsonstat?command=get-remote-ring&name=servfail-remotes HTTP/1.1
Host: example.com
Accept: application/json, text/javascript
X-API-Key: examplekey
```

Example response:

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
```

(continues on next page)

(continued from previous page)

```
Connection: close
Content-Length: 31
Content-Security-Policy: default-src 'self'; style-src 'self' 'unsafe-
↪inline'
Content-Type: application/json
Server: PowerDNS/4.1.11
X-Content-Type-Options: nosniff
X-Frame-Options: deny
X-Permitted-Cross-Domain-Policies: none
X-Xss-Protection: 1; mode=block

{"entries": [[4, "127.0.0.1"]]}
```

Example request:

```
GET /jsonstat?command=get-remote-ring&name=large-answer-remotes HTTP/1.
↪1
Host: example.com
Accept: application/json, text/javascript
X-API-Key: examplekey
```

Example response:

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
Connection: close
Content-Length: 43
Content-Security-Policy: default-src 'self'; style-src 'self' 'unsafe-
↪inline'
Content-Type: application/json
Server: PowerDNS/4.1.11
X-Content-Type-Options: nosniff
X-Frame-Options: deny
X-Permitted-Cross-Domain-Policies: none
X-Xss-Protection: 1; mode=block

{"entries": [[2, "127.0.0.1"], [1, "::1"]]}
```

Example request:

```
GET /jsonstat?command=get-remote-ring&name=timeouts HTTP/1.1
Host: example.com
Accept: application/json, text/javascript
X-API-Key: examplekey
```

Example response:

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
Connection: close
Content-Length: 189
Content-Security-Policy: default-src 'self'; style-src 'self' 'unsafe-
↪inline'
Content-Type: application/json
Server: PowerDNS/4.2.0-alpha1
X-Content-Type-Options: nosniff
X-Frame-Options: deny
X-Permitted-Cross-Domain-Policies: none
X-Xss-Protection: 1; mode=block

{"entries": [[3, "15.219.145.20"], [3, "15.211.192.20"], [2, "15.219.
↪160.20"], [2, "15.203.224.20"], [2, "15.219.145.21"], [2, "15.219.
↪160.21"], [2, "15.211.192.21"], [2, "15.203.224.21"]]}
```

(continues on next page)

(continued from previous page)

CHAPTER

FOURTEEN

SECURITY OF THE POWERDNS RECURSOR

For Security Advisories, see the *dedicated page*.

14.1 PowerDNS Security Policy

If you have a security problem to report, please email us at both peter.van.dijk@powerdns.com and remi.gacogne@powerdns.com. In case you want to encrypt your report using PGP, please use: https://doc.powerdns.com/powerdns-keyblock.asc

Please do not mail security issues to public lists, nor file a ticket, unless we do not get back to you in a timely manner. We fully credit reporters of security issues, and respond quickly, but please allow us a reasonable timeframe to coordinate a response.

We remind PowerDNS and dnsdist users that under the terms of the GNU General Public License, PowerDNS and dnsdist come with ABSOLUTELY NO WARRANTY. This *license* is included in this documentation.

If you believe you have found a security vulnerability that applies to DNS implementations generally, and you want to report this responsibly to a number of implementers, you might consider also using the Open Source DNS Vulnerability mailing list, managed by DNS-OARC.

14.1.1 YesWeHack

Security issues can also be reported on our YesWeHack page and might fetch a bounty. Do note that only the PowerDNS software is in scope for the YesWeHack program, not our websites or other infrastructure.

14.1.2 Disclosure Policy

- Let us know as soon as possible upon discovery of a potential security issue, and we'll make every effort to quickly resolve the issue.
- Provide us a reasonable amount of time to resolve the issue before any disclosure to the public or a third-party.
- We will always credit researchers in our *Security Advisories*.

14.2 Anti-spoofing

The PowerDNS Recursor uses a fresh UDP source port for each outgoing query, making spoofing around 64000 times harder. This raises the bar from 'easily doable given some time' to 'very hard'. Under some circumstances, 'some time' has been measured at 2 seconds. This technique was first used by `dnscache` by Dan J. Bernstein and is standardized in **RFC 5452**

In addition, PowerDNS detects when it is being sent too many unexpected answers, and mistrusts a proper answer if found within a clutch of unexpected ones.

This behaviour can be tuned using the *spoof-nearmiss-max*.

14.3 Throttling

PowerDNS implements a very simple but effective nameserver. Care has been taken not to overload remote servers in case of overly active clients.

This is implemented using the 'throttle'. This accounts all recent traffic and prevents queries that have been sent out recently from going out again.

There are three levels of throttling.

- If a remote server indicates that it is lame for a zone, the exact question won't be repeated in the next 60 seconds.
- After 4 ServFail responses in 60 seconds, the query gets throttled too.
- 5 timeouts in 20 seconds also lead to query suppression.

14.4 Security Polling

PowerDNS products can poll the security status of their respective versions. This polling, naturally, happens over DNS. If the result is that a given version has a security problem, the software will report this at level 'Error' during startup, and repeatedly during operations.

By default, security polling happens on the domain 'secpoll.powerdns.com', but this can be changed with the *security-poll-suffix*. If this setting is made empty, no polling will take place. Organizations wanting to host their own security zones can do so by changing this setting to a domain name under their control.

To make this easier, the zone used to host secpoll.powerdns.com is available.

To enable distributors of PowerDNS to signal that they have backported versions, the PACKAGEVERSION compilation-time macro can be used to set a distributor suffix.

14.4.1 Details

PowerDNS software sadly sometimes has critical security bugs. Even though we send out notifications of these via all channels available, we find that not everybody actually find out about our security releases.

To solve this, PowerDNS software will start polling for security notifications, and log these periodically. Secondly, the security status of the software will be reported using the built-in metrics. This allows operators to poll for the PowerDNS security status and alert on it.

In the implementation of this idea, we have taken the unique role of operating system distributors into account. Specifically, we can deal with backported security fixes.

Finally, this feature can be disabled, or operators can have the automated queries point at their own status service.

Implementation

PowerDNS software periodically tries to resolve 'auth-x.y.z.security-status.secpoll.powerdns.com|TXT' or 'recursor-x.y.z.security-status.secpoll.powerdns.com'.

The data returned is in one of the following forms:

- NXDOMAIN or resolution failure -> 0
- "1 Ok" -> 1
- "2 Upgrade recommended for security reasons, see . . . " -> 2

- "3 Upgrade mandatory for security reasons, see ..." -> 3

In cases 2 or 3, periodic logging commences. Case 2 can also be issued for non-security related upgrade recommendations for pre-releases. The metric security-status is set to 2 or 3 respectively. If at a later date, resolution fails, the security-status is not reset to 1. It could be lowered however if we discover the security status is less urgent than we thought.

If resolution fails, and the previous security-status was 1, the new security-status becomes 0 ('no data'). If the security-status was higher than 1, it will remain that way, and not get set to 0.

In this way, security-status of 0 really means 'no data', and cannot mask a known problem.

Distributions

Distributions frequently backport security fixes to the PowerDNS versions they ship. This might lead to a version number that is known to us to be insecure to be secure in reality.

To solve this issue, PowerDNS can be compiled with a distribution setting which will move the security polls from: 'auth-x.y.z.security-status.secpoll.powerdns.com' to 'auth-x.y.z-n.debian.security-status.secpoll.powerdns.com.

Note two things, one, there is a separate namespace for debian, and secondly, we use the package version of this release. This allows us to know that 4.0.1-1 (say) is insecure, but that 4.0.1-2 is not.

Configuration Details

The configuration setting *security-poll-suffix* is by default set to 'secpoll.powerdns.com'. If empty, nothing is polled. This can be moved to 'secpoll.yourorganization.com'.

If compiled with PACKAGEVERSION=3.1.6-abcde.debian, queries will be sent to "auth-3.1.6-abcde.debian.security-status.security-poll-suffix".

Delegation

If a distribution wants to host its own file with version information, we can delegate dist.security-status.secpoll.powerdns.com to their nameservers directly.

CHAPTER

FIFTEEN

SECURITY ADVISORIES

All security advisories for the PowerDNS Recursor are listed here.

15.1 PowerDNS Security Advisory 2006-01: Malformed TCP queries can lead to a buffer overflow which might be exploitable

- CVE: CVE-2006-4251
- Date: 13th of November 2006
- Affects: PowerDNS Recursor versions 3.1.3 and earlier, on all operating systems.
- Not affected: No versions of the PowerDNS Authoritative Server ('pdns_server') are affected.
- Severity: Critical
- Impact: Potential remote system compromise.
- Exploit: As far as we know, no exploit is available as of 11th of November 2006.
- Solution: Upgrade to PowerDNS Recursor 3.1.4, or apply the patches referred below and recompile
- Workaround: Disable TCP access to the Recursor. This will have slight operational impact, but it is likely that this will not lead to meaningful degradation of service. Disabling access is best performed at packet level, either by configuring a firewall, or instructing the host operating system to drop TCP connections to port 53. Additionally, exposure can be limited by configuring the `allow-from` setting so only trusted users can query your nameserver.

PowerDNS Recursor 3.1.3 and previous miscalculate the length of incoming TCP DNS queries, and will attempt to read up to 4 gigabytes of query into a 65535 byte buffer.

We have not verified if this problem might actually lead to a system compromise, but are acting on the assumption that it might.

For distributors, a minimal patch is available on the PowerDNS wiki. Additionally, those shipping very old versions of the PowerDNS Recursor might benefit from this patch.

The impact of these and other security problems can be lessened by considering the advice in FIXME: security-settings.

15.2 PowerDNS Security Advisory 2006-02: Zero second CNAME TTLs can make PowerDNS exhaust allocated stack space, and crash

- CVE: CVE-2006-4252

- Date: 13th of November 2006
- Affects: PowerDNS Recursor versions 3.1.3 and earlier, on all operating systems.
- Not affected: No versions of the PowerDNS Authoritative Server ('pdns_server') are affected.
- Severity: Moderate
- Impact: Denial of service
- Exploit: This problem can be triggered by sending queries for specifically configured domains
- Solution: Upgrade to PowerDNS Recursor 3.1.4, or apply commit 919.
- Workaround: None known. Exposure can be limited by configuring the **allow-from** setting so only trusted users can query your nameserver.

PowerDNS would recurse endlessly on encountering a CNAME loop consisting entirely of zero second CNAME records, eventually exceeding resources and crashing.

15.3 PowerDNS Security Advisory 2008-01: System random generator can be predicted, leading to the potential to 'spoof' PowerDNS Recursor

- CVE: Not yet assigned
- Date: 31st of March 2008
- Affects: PowerDNS Recursor versions 3.1.4 and earlier, on most operating systems
- Not affected: No versions of the PowerDNS Authoritative Server ('pdns_server') are affected.
- Severity:Moderate
- Impact: Data manipulation; client redirection
- Exploit: This problem can be triggered by sending queries for specifically configured domains, sending spoofed answer packets immediately afterwards.
- Solution: Upgrade to PowerDNS Recursor 3.1.5, or apply changesets 1159, 1160 and 1164.
- Workaround: None known. Exposure can be limited by configuring the **allow-from** setting so only trusted users can query your nameserver.

We would like to thank Amit Klein of Trusteer for bringing a serious vulnerability to our attention which would enable a smart attacker to 'spoof' previous versions of the PowerDNS Recursor into accepting possibly malicious data.

Details can be found on this Trusteer page.

This security problem was announced in this email message.

It is recommended that all users of the PowerDNS Recursor upgrade to 3.1.5 as soon as practicable, while we simultaneously note that busy servers are less susceptible to the attack, but not immune.

The vulnerability is present on all operating systems where the behaviour of the libc random() function can be predicted based on its past output. This includes at least all known versions of Linux, as well as Microsoft Windows, and probably FreeBSD and Solaris.

The magnitude of this vulnerability depends on internal details of the system random() generator. For Linux, the mathematics of the random generator are complex, but well understood and Amit Klein has written and published a proof of concept that can successfully predict its output after uninterrupted observation of 40-50 DNS queries.

Because the observation needs to be uninterrupted, busy PowerDNS Recursor instances are harder to subvert - other data is highly likely to be interleaved with traffic generated by an attacker.

Nevertheless, operators are urged to update at their earliest convenience.

15.4 PowerDNS Security Advisory 2010-01: PowerDNS Recursor up to and including 3.1.7.1 can be brought down and probably exploited

- CVE: CVE-2009-4009
- Date: 6th of January 2010
- Affects: PowerDNS Recursor 3.1.7.1 and earlier
- Not affected: No versions of the PowerDNS Authoritative ('pdns_server') are affected.
- Severity: Critical
- Impact: Denial of Service, possible full system compromise
- Exploit: Withheld
- Solution: Upgrade to PowerDNS Recursor 3.1.7.2 or higher
- Workaround: None. The risk of exploitation or denial of service can be decreased slightly by using the `allow-from` setting to only provide service to known users. The risk of a full system compromise can be reduced by running with a suitable reduced privilege user and group settings, and possibly chroot environment.

Using specially crafted packets, it is possible to force a buffer overflow in the PowerDNS Recursor, leading to a crash.

This vulnerability was discovered by a third party that (for now) prefers not to be named. PowerDNS is very grateful however for their help in improving PowerDNS security.

15.5 PowerDNS Security Advisory 2010-02: PowerDNS Recursor up to and including 3.1.7.1 can be spoofed into accepting bogus data

- CVE: CVE-2009-4010
- Date: 6th of January 2010
- Affects: PowerDNS Recursor 3.1.7.1 and earlier
- Not affected: No versions of the PowerDNS Authoritative ('pdns_server') are affected.
- Severity: High
- Impact: Using smart techniques, it is possible to fool the PowerDNS Recursor into accepting unauthorized data
- Exploit: Withheld
- Solution: Upgrade to PowerDNS Recursor 3.1.7.2 or higher
- Workaround: None.

Using specially crafted zones, it is possible to fool the PowerDNS Recursor into accepting bogus data. This data might be harmful to your users. An attacker would be able to divert data from, say, bigbank.com to an IP address of his choosing.

This vulnerability was discovered by a third party that (for now) prefers not to be named. PowerDNS is very grateful however for their help in improving PowerDNS security.

15.6 PowerDNS Security Advisory 2014-01: PowerDNS Recursor 3.6.0 can be crashed remotely

- CVE: CVE-2014-3614
- Date: 10th of September 2014
- Credit: Dedicated PowerDNS users willing to study a crash that happens once every few months (thanks)
- Affects: Only PowerDNS Recursor version 3.6.0.
- Not affected: No other versions of PowerDNS Recursor, no versions of PowerDNS Authoritative Server
- Severity: High
- Impact: Crash
- Exploit: The sequence of packets required is known
- Risk of system compromise: No
- Solution: Upgrade to PowerDNS Recursor 3.6.1
- Workaround: Restrict service using `allow-from <../recursor/settings.md#allow-from>‘__, install script that restarts PowerDNS

Recently, we’ve discovered that PowerDNS Recursor 3.6.0 (but NOT earlier) can crash when exposed to a specific sequence of malformed packets. This sequence happened spontaneously with one of our largest deployments, and the packets did not appear to have a malicious origin.

Yet, this crash can be triggered remotely, leading to a denial of service attack. There appears to be no way to use this crash for system compromise or stack overflow.

Upgrading to 3.6.1 solves the issue.

In addition, you can apply a minimal fix to your own tree.

As for workarounds, only clients in allow-from are able to trigger the crash, so this should be limited to your userbase. Secondly, this and this can be used to enable Upstart and Systemd to restart the PowerDNS Recursor automatically.

15.7 PowerDNS Security Advisory 2014-02: PowerDNS Recursor 3.6.1 and earlier can be made to provide bad service

- CVE: CVE-2014-8601
- Date: 8th of December 2014
- Credit: Florian Maury (ANSSI)
- Affects: PowerDNS Recursor versions 3.6.1 and earlier
- Not affected: PowerDNS Recursor 3.6.2; no versions of PowerDNS Authoritative Server
- Severity: High
- Impact: Degraded service
- Exploit: This problem can be triggered by sending queries for specifically configured domains
- Risk of system compromise: No
- Solution: Upgrade to PowerDNS Recursor 3.6.2
- Workaround: None known. Exposure can be limited by configuring the **allow-from** setting so only trusted users can query your nameserver.

Recently we released PowerDNS Recursor 3.6.2 with a new feature that strictly limits the amount of work we'll perform to resolve a single query. This feature was inspired by performance degradations noted when resolving domains hosted by 'ezdns.it', which can require thousands of queries to resolve.

During the 3.6.2 release process, we were contacted by a government security agency with news that they had found that all major caching nameservers, including PowerDNS, could be negatively impacted by specially configured, hard to resolve domain names. With their permission, we continued the 3.6.2 release process with the fix for the issue already in there.

We recommend that all users upgrade to 3.6.2 if at all possible. Alternatively, you can apply a minimal fix (including patches for older versions) to your own tree.

As for workarounds, only clients in allow-from are able to trigger the degraded service, so this should be limited to your userbase.

15.8 PowerDNS Security Advisory 2015-01: Label decompression bug can cause crashes or CPU spikes

- CVE: CVE-2015-1868 (original), CVE-2015-5470 (update)
- Date: 23rd of April 2015, updated 7th of July 2015
- Credit: Aki Tuomi, Toshifumi Sakaguchi
- Affects: PowerDNS Recursor versions 3.5 and up; Authoritative Server 3.2 and up
- Not affected: Recursor 3.6.4; Recursor 3.7.3; Auth 3.3.3; Auth 3.4.5
- Severity: High
- Impact: Degraded service
- Exploit: This problem can be triggered by sending queries for specifically configured domains, or by sending specially crafted query packets
- Risk of system compromise: No
- Solution: Upgrade to any of the non-affected versions
- Workaround: Run your Recursor under a supervisor. Exposure can be limited by configuring the `allow-from <../recursor/settings.md#allow-from>'__ setting so only trusted users can query your nameserver. There is no workaround for the Authoritative server.

A bug was discovered in our label decompression code, making it possible for names to refer to themselves, thus causing a loop during decompression. On some platforms, this bug can be abused to cause crashes. On all platforms, this bug can be abused to cause service-affecting CPU spikes.

We recommend that all users upgrade to a corrected version if at all possible. Alternatively, if you want to apply a minimal fix to your own tree, please find patches here.

As for workarounds, for the Recursor: only clients in allow-from are able to trigger the degraded service, so this should be limited to your userbase; further, we recommend running your critical services under supervision such as systemd, supervisord, daemontools, etc.

There is no workaround for the Authoritative Server.

We want to thank Aki Tuomi for noticing this in production, and then digging until he got to the absolute bottom of what at the time appeared to be a random and spurious failure.

We want to thank Toshifumi Sakaguchi for further investigation into the issue after the initial announcement, and for demonstrating to us quite clearly the CPU spike issues.

Update 7th of July 2015: Toshifumi Sakaguchi discovered that the original fix was insufficient in some cases. Updated versions of the Authoritative Server and Recursor were released on the 9th of June. Minimal patches are available. The insufficient fix was assigned CVE-2015-5470.

15.9 PowerDNS Security Advisory 2016-02: Crafted queries can cause abnormal CPU usage

- CVE: CVE-2016-7068
- Date: December 15th 2016
- Credit: Florian Heinz and Martin Kluge
- Affects: PowerDNS Authoritative Server up to and including 3.4.10, 4.0.1, PowerDNS Recursor up to and including 3.7.3, 4.0.3
- Not affected: PowerDNS Authoritative Server 3.4.11, 4.0.2 and PowerDNS Recursor 3.7.4, 4.0.4
- Severity: Medium
- Impact: Degraded service or Denial of service
- Exploit: This issue can be triggered by sending specially crafted query packets
- Risk of system compromise: No
- Solution: Upgrade to a non-affected version
- Workaround: Run dnsdist with the rules provided below in front of potentially affected servers.

An issue has been found in PowerDNS allowing a remote, unauthenticated attacker to cause an abnormal CPU usage load on the PowerDNS server by sending crafted DNS queries, which might result in a partial denial of service if the system becomes overloaded. This issue is based on the fact that the PowerDNS server parses all records present in a query regardless of whether they are needed or even legitimate. A specially crafted query containing a large number of records can be used to take advantage of that behaviour. This issue has been assigned CVE-2016-7068.

PowerDNS Authoritative Server up to and including 3.4.10 and 4.0.1 are affected. PowerDNS Recursor up to and including 3.7.3 and 4.0.3 are affected.

dnsdist can be used to block crafted queries, using `RecordsCountRule()` and `RecordsTypeCountRule()` to block queries with crafted records.

For those unable to upgrade to a new version, a minimal patch is available

We would like to thank Florian Heinz and Martin Kluge for finding and subsequently reporting this issue.

15.10 PowerDNS Security Advisory 2016-04: Insufficient validation of TSIG signatures

- CVE: CVE-2016-7073 CVE-2016-7074
- Date: December 15th 2016
- Credit: Mongo
- Affects: PowerDNS Authoritative Server up to and including 3.4.10, 4.0.1, PowerDNS Recursor from 4.0.0 and up to and including 4.0.3
- Not affected: PowerDNS Authoritative Server 3.4.11, 4.0.2, PowerDNS Recursor < 4.0.0, 4.0.4
- Severity: Medium
- Impact: Zone content alteration
- Exploit: This problem can be triggered by an attacker in position of man-in-the-middle
- Risk of system compromise: No
- Solution: Upgrade to a non-affected version

Two issues have been found in PowerDNS Authoritative Server allowing an attacker in position of man-in-the-middle to alter the content of an AXFR because of insufficient validation of TSIG signatures. The first issue is a missing check of the TSIG time and fudge values in `AXFRRetriever`, leading to a possible replay attack. This issue has been assigned CVE-2016-7073. The second issue is a missing check that the TSIG record is the last one, leading to the possibility of parsing records that are not covered by the TSIG signature. This issue has been assigned CVE-2016-7074.

PowerDNS Authoritative Server up to and including 3.4.10 and 4.0.1 are affected. PowerDNS Recursor from 4.0.0 up to and including 4.0.3 are affected.

For those unable to upgrade to a new version, a minimal patch is available

We would like to thank Mongo for finding and subsequently reporting this issue.

15.11 PowerDNS Security Advisory 2017-03: Insufficient validation of DNSSEC signatures

- CVE: CVE-2017-15090
- Date: November 27th 2017
- Credit: Kees Monshouwer
- Affects: PowerDNS Recursor from 4.0.0 and up to and including 4.0.6
- Not affected: PowerDNS Recursor < 4.0.0, 4.0.7
- Severity: Medium
- Impact: Records manipulation
- Exploit: This problem can be triggered by an attacker in position of man-in-the-middle
- Risk of system compromise: No
- Solution: Upgrade to a non-affected version

An issue has been found in the DNSSEC validation component of PowerDNS Recursor, where the signatures might have been accepted as valid even if the signed data was not in bailiwick of the DNSKEY used to sign it. This allows an attacker in position of man-in-the-middle to alter the content of records by issuing a valid signature for the crafted records. This issue has been assigned CVE-2017-15090.

PowerDNS Recursor from 4.0.0 up to and including 4.0.6 are affected.

For those unable to upgrade to a new version, a minimal patch is available

We would like to thank Kees Monshouwer for finding and subsequently reporting this issue.

15.12 PowerDNS Security Advisory 2017-05: Cross-Site Scripting in the web interface

- CVE: CVE-2017-15092
- Date: November 27th 2017
- Credit: Nixu, Chris Navarrete of Fortinet's Fortiguard Labs
- Affects: PowerDNS Recursor from 4.0.0 up to and including 4.0.6
- Not affected: PowerDNS Recursor 4.0.7, 3.7.x
- Severity: Medium
- Impact: Alteration and denial of service of the web interface

- Exploit: This problem can be triggered by an attacker sending DNS queries to the server
- Risk of system compromise: No
- Solution: Upgrade to a non-affected version

An issue has been found in the web interface of PowerDNS Recursor, where the qname of DNS queries was displayed without any escaping, allowing a remote attacker to inject HTML and JavaScript code into the web interface, altering the content. This issue has been assigned CVE-2017-15092.

PowerDNS Recursor from 4.0.0 up to and including 4.0.6 are affected.

For those unable to upgrade to a new version, a minimal patch is available

We would like to thank Nixu and Chris Navarrete of Fortinet's Fortiguard Labs for independently finding and reporting this issue.

15.13 PowerDNS Security Advisory 2017-06: Configuration file injection in the API

- CVE: CVE-2017-15093
- Date: November 27th 2017
- Credit: Nixu
- Affects: PowerDNS Recursor up to and including 4.0.6, 3.7.4
- Not affected: PowerDNS Recursor 4.0.7
- Severity: Medium
- Impact: Alteration of configuration by an API user
- Exploit: This problem can be triggered by an attacker with valid API credentials
- Risk of system compromise: No
- Solution: Upgrade to a non-affected version
- Workaround: Disable the ability to alter the configuration via the API by setting *api-config-dir* to an empty value (default), or set the API read-only via the *api-readonly* setting.

An issue has been found in the API of PowerDNS Recursor during a source code audit by Nixu. When *api-config-dir* is set to a non-empty value, which is not the case by default, the API allows an authorized user to update the Recursor's ACL by adding and removing netmasks, and to configure forward zones. It was discovered that the new netmask and IP addresses of forwarded zones were not sufficiently validated, allowing an authenticated user to inject new configuration directives into the Recursor's configuration. This issue has been assigned CVE-2017-15093.

PowerDNS Recursor up to and including 4.0.6 and 3.7.4 are affected.

For those unable to upgrade to a new version, a minimal patch is available

We would like to thank Nixu for finding and subsequently reporting this issue.

15.14 PowerDNS Security Advisory 2017-07: Memory leak in DNSSEC parsing

- CVE: CVE-2017-15094
- Date: November 27th 2017
- Credit: Nixu

- Affects: PowerDNS Recursor from 4.0.0 up to and including 4.0.6
- Not affected: PowerDNS Recursor 4.0.7
- Severity: Medium
- Impact: Denial of service
- Exploit: This problem can be triggered by an authoritative server sending crafted ECDSA DNSSEC keys to the Recursor.
- Risk of system compromise: No
- Solution: Upgrade to a non-affected version
- Workaround: Disable DNSSEC validation by setting the *dnssec* parameter to *off* or *process-no-validate* (default).

An issue has been found in the DNSSEC parsing code of PowerDNS Recursor during a code audit by Nixu, leading to a memory leak when parsing specially crafted DNSSEC ECDSA keys. These keys are only parsed when validation is enabled by setting *dnssec* to a value other than *off* or *process-no-validate* (default). This issue has been assigned CVE-2017-15094.

PowerDNS Recursor from 4.0.0 up to and including 4.0.6 are affected.

For those unable to upgrade to a new version, a minimal patch is available

We would like to thank Nixu for finding and subsequently reporting this issue.

15.15 PowerDNS Security Advisory 2017-08: Crafted CNAME answer can cause a denial of service

- CVE: CVE-2017-15120
- Date: December 11th 2017
- Credit: Toshifumi Sakaguchi
- Affects: PowerDNS Recursor from 4.0.0 up to and including 4.0.7
- Not affected: PowerDNS Recursor 3.7.4, 4.0.8, 4.1.0
- Severity: High
- Impact: Denial of service
- Exploit: This problem can be triggered by an authoritative server sending a crafted CNAME answer with a class other than IN to the Recursor.
- Risk of system compromise: No
- Solution: Upgrade to a non-affected version
- Workaround: run the process inside a supervisor like supervisord or systemd

An issue has been found in the parsing of authoritative answers in PowerDNS Recursor, leading to a NULL pointer dereference when parsing a specially crafted answer containing a CNAME of a different class than IN. This issue has been assigned CVE-2017-15120.

When the PowerDNS Recursor is run inside a supervisor like supervisord or systemd, it will be automatically restarted, limiting the impact to somewhat degraded service.

PowerDNS Recursor from 4.0.0 up to and including 4.0.7 are affected.

For those unable to upgrade to a new version, a minimal patch is available

We would like to thank Toshifumi Sakaguchi for finding and subsequently reporting this issue.

15.16 PowerDNS Security Advisory 2018-01: Insufficient validation of DNSSEC signatures

- CVE: CVE-2018-1000003
- Date: January 22nd 2018
- Credit: CZ.NIC
- Affects: PowerDNS Recursor 4.1.0
- Not affected: PowerDNS Recursor < 4.1.0, 4.1.1
- Severity: Low
- Impact: Denial of existence spoofing
- Exploit: This problem can be triggered by an attacker in position of man-in-the-middle
- Risk of system compromise: No
- Solution: Upgrade to a non-affected version

An issue has been found in the DNSSEC validation component of PowerDNS Recursor, allowing an ancestor delegation NSEC or NSEC3 record to be used to wrongfully prove the non-existence of a RR below the owner name of that record. This would allow an attacker in position of man-in-the-middle to send a NXDOMAIN answer for a name that does exist. This issue has been assigned CVE-2018-1000003.

PowerDNS Recursor 4.1.0 is affected.

For those unable to upgrade to a new version, a minimal patch is available

We would like to thank CZ.NIC for finding and subsequently reporting this issue! Please also see https://lists.nic.cz/pipermail/knot-dns-users/2018-January/001309.html

15.17 PowerDNS Security Advisory 2018-04: Crafted answer can cause a denial of service

- CVE: CVE-2018-10851
- Date: November 6th 2018
- Affects: PowerDNS Recursor from 3.2 up to and including 4.1.4
- Not affected: 4.1.5, 4.0.9
- Severity: Medium
- Impact: Denial of service
- Exploit: This problem can be triggered by an authoritative server
- Risk of system compromise: No
- Solution: Upgrade to a non-affected version
- Workaround: run the process inside a supervisor

An issue has been found in PowerDNS Recursor allowing a malicious authoritative server to cause a memory leak by sending specially crafted records. The issue is due to the fact that some memory is allocated before the parsing and is not always properly released if the record is malformed.

This issue has been assigned CVE-2018-10851.

When the PowerDNS Recursor is run inside a supervisor like supervisord or systemd, an out-of-memory crash will lead to an automatic restart, limiting the impact to a somewhat degraded service.

PowerDNS Recursor from 3.2 up to and including 4.1.4 is affected. Please note that at the time of writing, PowerDNS Recursor 3.7 and below are no longer supported, as described in *End of life statements*.

15.18 PowerDNS Security Advisory 2018-06: Packet cache pollution via crafted query

- CVE: CVE-2018-14626
- Date: November 6th 2018
- Affects: PowerDNS Recursor from 4.0.0 up to and including 4.1.4
- Not affected: 4.1.5, 4.0.9
- Severity: Medium
- Impact: Denial of service
- Exploit: This problem can be triggered via crafted queries
- Risk of system compromise: No
- Solution: Upgrade to a non-affected version

An issue has been found in PowerDNS Recursor allowing a remote user to craft a DNS query that will cause an answer without DNSSEC records to be inserted into the packet cache and be returned to clients asking for DNSSEC records, thus hiding the presence of DNSSEC signatures for a specific qname and qtype. For a DNSSEC-signed domain, this means that clients performing DNSSEC validation by themselves might consider the answer to be bogus until it expires from the packet cache, leading to a denial of service.

This issue has been assigned CVE-2018-14626.

PowerDNS Recursor from 4.0.0 up to and including 4.1.4 is affected.

We would like to thank Kees Monshouwer for finding and subsequently reporting this issue.

15.19 PowerDNS Security Advisory 2018-07: Crafted query for meta-types can cause a denial of service

- CVE: CVE-2018-14644
- Date: November 6th 2018
- Affects: PowerDNS Recursor from 4.0.0 up to and including 4.1.4
- Not affected: 4.0.9, 4.1.5
- Severity: Medium
- Impact: Denial of service
- Exploit: This problem can be triggered via crafted queries for some domains
- Risk of system compromise: No
- Solution: Upgrade to a non-affected version

An issue has been found in PowerDNS Recursor where a remote attacker sending a DNS query for a meta-type like OPT can lead to a zone being wrongly cached as failing DNSSEC validation. It only arises if the parent zone is signed, and all the authoritative servers for that parent zone answer with FORMERR to a query for at least one of the meta-types. As a result, subsequent queries from clients requesting DNSSEC validation will be answered with a ServFail.

This issue has been assigned CVE-2018-14644 by Red Hat.

PowerDNS Recursor from 4.0.0 up to and including 4.1.4 is affected.

We would like to thank Toshifumi Sakaguchi for finding and subsequently reporting this issue.

15.20 PowerDNS Security Advisory 2018-09: Crafted query can cause a denial of service

- CVE: CVE-2018-16855
- Date: 26th of November 2018
- Affects: PowerDNS Recursor from 4.1.0 up to and including 4.1.7
- Not affected: 4.0.x, 4.1.8
- Severity: Medium
- Impact: Denial of service
- Exploit: This problem can be triggered via crafted queries
- Risk of system compromise: No
- Solution: Upgrade to a non-affected version

An issue has been found in PowerDNS Recursor where a remote attacker sending a DNS query can trigger an out-of-bounds memory read while computing the hash of the query for a packet cache lookup, possibly leading to a crash.

This issue has been assigned CVE-2018-16855 by Red Hat.

When the PowerDNS Recursor is run inside a supervisor like supervisord or systemd, a crash will lead to an automatic restart, limiting the impact to a somewhat degraded service.

PowerDNS Recursor from 4.1.0 up to and including 4.1.7 is affected.

15.21 PowerDNS Security Advisory 2019-01: Lua hooks are not applied in certain configurations

- CVE: CVE-2019-3806
- Date: 21st of January 2019
- Affects: PowerDNS Recursor from 4.1.4 up to and including 4.1.8
- Not affected: 4.0.x, 4.1.0 up to and including 4.1.3, 4.1.9
- Severity: Low
- Impact: Access restriction bypass
- Exploit: This problem can be triggered via TCP queries
- Risk of system compromise: No
- Solution: Upgrade to a non-affected version
- Workaround: Switch to pdns-distributes-queries=no

An issue has been found in PowerDNS Recursor where Lua hooks are not properly applied to queries received over TCP in some specific combination of settings, possibly bypassing security policies enforced using Lua.

When the recursor is configured to run with more than one thread (threads=X) and to do the distribution of incoming queries to the worker threads itself (pdns-distributes-queries=yes), the Lua script is not properly loaded in the thread handling incoming TCP queries, causing the Lua hooks to not be properly applied.

This issue has been assigned CVE-2019-3806 by Red Hat.

PowerDNS Recursor from 4.1.4 up to and including 4.1.8 is affected.

15.22 PowerDNS Security Advisory 2019-02: Insufficient validation of DNSSEC signatures

- CVE: CVE-2019-3807
- Date: 21st of January 2019
- Affects: PowerDNS Recursor from 4.1.0 up to and including 4.1.8
- Not affected: 4.0.x, 4.1.9
- Severity: Medium
- Impact: Insufficient validation
- Exploit: This problem can be triggered via crafted responses
- Risk of system compromise: No
- Solution: Upgrade to a non-affected version

An issue has been found in PowerDNS Recursor where records in the answer section of responses received from authoritative servers with the AA flag not set were not properly validated, allowing an attacker to bypass DNSSEC validation.

This issue has been assigned CVE-2019-3807 by Red Hat.

PowerDNS Recursor from 4.1.0 up to and including 4.1.8 is affected.

We would like to thank Ralph Dolmans and George Thessalonikefs of NLNetLabs for finding and subsequently reporting this issue!

15.23 PowerDNS Security Advisory 2020-01: Denial of Service

- CVE: CVE-2020-10995
- Date: May 19th 2020
- Affects: PowerDNS Recursor from 4.1.0 up to and including 4.3.0
- Not affected: 4.1.16, 4.2.2, 4.3.1
- Severity: Medium
- Impact: Degraded Service
- Exploit: This problem can be triggered via a crafted reply
- Risk of system compromise: No
- Solution: Upgrade to a non-affected version
- Workaround: None

An issue in the DNS protocol has been found that allow malicious parties to use recursive DNS services to attack third party authoritative name servers. The attack uses a crafted reply by an authoritative name server to amplify the resulting traffic between the recursive and other authoritative name servers. Both types of service can suffer degraded performance as an effect.

This issue has been assigned CVE-2020-10995.

PowerDNS Recursor from 4.1.0 up to and including 4.3.0 is affected. PowerDNS Recursor 4.1.16, 4.2.2 and 4.3.1 contain a mitigation to limit the impact of this DNS protocol issue.

Please note that at the time of writing, PowerDNS Recursor 4.0 and below are no longer supported, as described in https://doc.powerdns.com/recursor/appendices/EOL.html.

We would like to thank Lior Shafir, Yehuda Afek and Anat Bremler-Barr for finding and subsequently reporting this issue!

15.24 PowerDNS Security Advisory 2020-02: Insufficient validation of DNSSEC signatures

- CVE: CVE-2020-12244
- Date: May 19th 2020
- Affects: PowerDNS Recursor from 4.1.0 up to and including 4.3.0
- Not affected: 4.3.1, 4.2.2, 4.1.16
- Severity: Medium
- Impact: Denial of existence spoofing
- Exploit: This problem can be triggered by an attacker in position of man-in-the-middle
- Risk of system compromise: No
- Solution: Upgrade to a non-affected version
- Workaround: None

An issue has been found in PowerDNS Recursor 4.1.0 through 4.3.0 where records in the answer section of a NXDOMAIN response lacking an SOA were not properly validated in SyncRes::processAnswer. This would allow an attacker in position of man-in-the-middle to send a NXDOMAIN answer for a name that does exist, bypassing DNSSEC validation.

This issue has been assigned CVE-2020-12244.

PowerDNS Recursor from 4.1.0 up to and including 4.3.0 is affected.

Please note that at the time of writing, PowerDNS Authoritative 4.0 and below are no longer supported, as described in https://doc.powerdns.com/authoritative/appendices/EOL.html.

We would like to thank Matt Nordhoff for finding and subsequently reporting this issue!

15.25 PowerDNS Security Advisory 2020-03: Information disclosure

- CVE: CVE-2020-10030
- Date: May 19th 2020
- Affects: PowerDNS Recursor from 4.1.0 up to and including 4.3.0
- Not affected: 4.3.1, 4.2.2, 4.1.16
- Severity: Low
- Impact: Information Disclosure, Denial of Service
- Exploit: This problem can be triggered via a crafted hostname
- Risk of system compromise: No
- Solution: Upgrade to a non-affected version
- Workaround: None

An issue has been found in PowerDNS Recursor allowing an attacker with enough privileges to change the system's hostname to cause disclosure of uninitialized memory content via a stack-based out-of-bounds read. It only occurs on systems where gethostname() does not null-terminate the returned string if the hostname is larger than the supplied buffer. Linux systems are not affected because the buffer is always large enough. OpenBSD systems are not affected because the returned hostname is always null-terminated. Under some conditions this issue can lead to the writing of one null-byte out-of-bounds on the stack, causing a denial of service or possibly arbitrary code execution.

This issue has been assigned CVE-2020-10030.

PowerDNS Recursor from 4.1.0 up to and including 4.3.0 is affected.

Please note that at the time of writing, PowerDNS Recursor 4.0 and below are no longer supported, as described in https://doc.powerdns.com/recursor/appendices/EOL.html.

We would like to thank Valentei Sergey for finding and subsequently reporting this issue!

15.26 PowerDNS Security Advisory 2020-04: Access restriction bypass

- CVE: CVE-2020-14196
- Date: July 1st 2020
- Affects: PowerDNS Recursor up to and including 4.3.1, 4.2.2 and 4.1.16
- Not affected: 4.3.2, 4.2.3, 4.1.17
- Severity: Low
- Impact: Access restriction bypass
- Exploit: This problem can be triggered by sending HTTP queries
- Risk of system compromise: No
- Solution: Upgrade to a non-affected version
- Workaround: Disable the webserver, set a password or an API key. Additionally, restrict the binding address using the *webserver-address* setting to local addresses only and/or use a firewall to disallow web requests from untrusted sources reaching the webserver listening address.

An issue has been found in PowerDNS Recursor where the ACL applied to the internal web server via *webserver-allow-from* is not properly enforced, allowing a remote attacker to send HTTP queries to the internal web server, bypassing the restriction.

In the default configuration the API webserver is not enabled. Only installations using a non-default value for *webserver* and *webserver-address* are affected.

15.27 PowerDNS Security Advisory 2020-07: Cache pollution

- CVE: CVE-2020-25829
- Date: 13th of October 2020
- Affects: PowerDNS Recursor up to and including 4.3.4, 4.2.4 and 4.1.17
- Not affected: 4.3.5, 4.2.5, 4.1.18
- Severity: High
- Impact: Denial of service
- Exploit: This problem can be triggered by sending DNS queries

- Risk of system compromise: No
- Solution: Upgrade to a non-affected version
- Workaround: Filter ANY queries to prevent them from reaching the recursor.

An issue has been found in PowerDNS Recursor where a remote attacker can cause the cached records for a given name to be updated to the 'Bogus' DNSSEC validation state, instead of their actual DNSSEC 'Secure' state, via a DNS ANY query. This results in a denial of service for installations that always validate (dnssec=validate) and for clients requesting validation when on-demand validation is enabled (dnssec=process).

15.28 PowerDNS Security Advisory 2022-01: incomplete validation of incoming IXFR transfer in Authoritative Server and Recursor

- CVE: CVE-2022-27227
- Date: 25th of March 2022.
- Affects: PowerDNS Authoritative version 4.4.2, 4.5.3, 4.6.0 and PowerDNS Recursor 4.4.7, 4.5.7 and 4.6.0
- Not affected: PowerDNS Authoritative Server 4.4.3, 4.5.4, 4.6.1 and PowerDNS Recursor 4.4.8, 4.5.8 and 4.6.1
- Severity: Low
- Impact: Denial of service
- Exploit: This problem can be triggered by an attacker controlling the network path for IXFR transfers
- Risk of system compromise: None
- Solution: Upgrade to patched version, do not use IXFR in Authoritative Server
- In the Authoritative server this issue only applies to secondary zones for which IXFR transfers have been enabled and the network path to the primary server is not trusted. Note that IXFR transfers are not enabled by default.
- In the Recursor it applies to setups retrieving one or more RPZ zones from a remote server if the network path to the server is not trusted.

IXFR usually exchanges only the modifications between two versions of a zone, but sometimes needs to fall back to a full transfer of the current version. When IXFR falls back to a full zone transfer, an attacker in position of man-in-the-middle can cause the transfer to be prematurely interrupted. This interrupted transfer is mistakenly interpreted as a complete transfer, causing an incomplete zone to be processed. For the Authoritative Server, IXFR transfers are not enabled by default. The Recursor only uses IXFR for retrieving RPZ zones. An incomplete RPZ transfer results in missing policy entries, potentially causing some DNS names and IP addresses to not be properly intercepted.

We would like to thank Nicolas Dehaine and Dmitry Shabanov from ThreatSTOP for reporting and initial analysis of this issue.

15.29 PowerDNS Security Advisory 2022-02: incomplete exception handling related to protobuf message generation

- CVE: CVE-2022-37428
- Date: 23th of August 2022.
- Affects: PowerDNS Recursor up to and including 4.5.9, 4.6.2 and 4.7.1
- Not affected: PowerDNS Recursor 4.5.10, 4.6.3 and 4.7.2

- Severity: Medium
- Impact: Denial of service
- Exploit: This problem can be triggered by a remote attacker with access to the recursor if protobuf logging is enabled
- Risk of system compromise: None
- Solution: Upgrade to patched version, disable protobuf logging of responses

This issue only affects recursors which have protobuf logging enabled using the

- `protobufServer` function with `logResponses=true` or
- `outgoingProtobufServer` function with `logResponses=true`

If either of these functions is used without specifying `logResponses`, its value is `true`. An attacker needs to have access to the recursor, i.e. the remote IP must be in the access control list. If an attacker queries a name that leads to an answer with specific properties, a protobuf message might be generated that causes an exception. The code does not handle this exception correctly, causing a denial of service.

15.30 PowerDNS Security Advisory 2023-01: unbounded recursion results in program termination

- CVE: CVE-2023-22617
- Date: 20th of January 2023
- Affects: PowerDNS Recursor 4.8.0
- Not affected: PowerDNS Recursor < 4.8.0, PowerDNS Recursor 4.8.1
- Severity: High
- Impact: Denial of service
- Exploit: This problem can be triggered by a remote attacker with access to the recursor by querying names from specific mis-configured domains
- Risk of system compromise: None
- Solution: Upgrade to patched version

An issue in the processing of queries for misconfigured domains has been found in PowerDNS Recursor 4.8.0, allowing a remote attacker to crash the recursor by sending a DNS query for one of these domains. The issue happens because the recursor enters a unbounded loop, exceeding its stack memory. Because of the specific way in which this issue happens, we do not believe this issue to be exploitable for code execution.

PowerDNS Recursor versions before 4.8.0 are not affected.

Note that when the PowerDNS Recursor is run inside a supervisor like supervisord or systemd, a crash will lead to an automatic restart, limiting the impact to a somewhat degraded service.

CVSS 3.0 score: 8.2 (High) https://www.first.org/cvss/calculator/3.0#CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:L/A:H/E:H/RL:U/RC:C

Thanks to applied-privacy.net for reporting this issue and their assistance in diagnosing it.

15.31 PowerDNS Security Advisory 2023-02: Deterred spoofing attempts can lead to authoritative servers being marked unavailable

- CVE: CVE-2023-26437

- Date: 29th of March 2023
- Affects: PowerDNS Recursor up to and including 4.6.5, 4.7.4 and 4.8.3
- Not affected: PowerDNS Recursor 4.6.6, 4.7.5 and 4.8.4
- Severity: Low
- Impact: Denial of service
- Exploit: Successful spoofing may lead to authoritative servers being marked unavailable
- Risk of system compromise: None
- Solution: Upgrade to patched version

When the recursor detects and deters a spoofing attempt or receives certain malformed DNS packets, it throttles the server that was the target of the impersonation attempt so that other authoritative servers for the same zone will be more likely to be used in the future, in case the attacker controls the path to one server only. Unfortunately this mechanism can be used by an attacker with the ability to send queries to the recursor, guess the correct source port of the corresponding outgoing query and inject packets with a spoofed IP address to force the recursor to mark specific authoritative servers as not available, leading a denial of service for the zones served by those servers.

CVSS 3.0 score: 3.7 (Low) https://www.first.org/cvss/calculator/3.0#CVSS:3.0/AV:N/AC:H/PR:N/UI:R/S:C/C:N/I:N/A:L

Thanks to Xiang Li from Network and Information Security Laboratory, Tsinghua University for reporting this issue.

15.32 PowerDNS Security Advisory 2024-01: crafted DNSSEC records in a zone can lead to a denial of service in Recursor

- CVE: CVE-2023-50387 and CVE-2023-50868
- Date: 13th of February 2024.
- Affects: PowerDNS Recursor up to and including 4.8.5, 4.9.2 and 5.0.1
- Not affected: PowerDNS Recursor 4.8.6, 4.9.3 and 5.0.2
- Severity: High
- Impact: Denial of service
- Exploit: This problem can be triggered by an attacker publishing a crafted zone
- Risk of system compromise: None
- Solution: Upgrade to patched version or disable DNSSEC validation

An attacker can publish a zone that contains crafted DNSSEC related records. While validating results from queries to that zone using the RFC mandated algorithms, the Recursor's resource usage can become so high that processing of other queries is impacted, resulting in a denial of service. Note that any resolver following the RFCs can be impacted, this is not a problem of this particular implementation.

CVSS Score: 7.5, see https://nvd.nist.gov/vuln-metrics/cvss/v3-calculator?vector=AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:N/A:H&version=3.1

The remedies are one of:

- upgrade to a patched version
- disable DNSSEC validation by setting `dnssec=off` or `process-no-validate`; when using YAML settings: `dnssec.validate: off` or `process-no-validate`. Note that this will affect clients depending on DNSSEC validation.

We would like to thank Elias Heftrig, Haya Schulmann, Niklas Vogel, and Michael Waidner from the German National Research Center for Applied Cybersecurity ATHENE for bringing this issue to the attention of the DNS community and especially Niklas Vogel for his assistance in validating the patches. We would also like to thank Petr Špaček from ISC for discovering and responsibly disclosing CVE-2023-50868.

15.33 PowerDNS Security Advisory 2024-02: if recursive forwarding is configured, crafted responses can lead to a denial of service in Recursor

- CVE: CVE-2024-25583
- Date: 24th of April 2024.
- Affects: PowerDNS Recursor 4.8.7, 4.9.4 and 5.0.3; earlier versions are not affected
- Not affected: PowerDNS Recursor 4.8.8, 4.9.5 and 5.0.4
- Severity: High (only when using recursive forwarding)
- Impact: Denial of service
- Exploit: This problem can be triggered by an attacker publishing a crafted zone
- Risk of system compromise: None
- Solution: Upgrade to patched version

A crafted response from an upstream server the recursor has been configured to forward-recurse to can cause a Denial of Service in the Recursor. The default configuration of the Recursor does not use recursive forwarding and is not affected.

CVSS Score: 7.5, only for configurations using recursive forwarding, see https://nvd.nist.gov/vuln-metrics/cvss/v3-calculator?vector=AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:N/A:H&version=3.1

The remedy is to update to a patched version.

15.34 Older security advisories

Version 3.0 of the PowerDNS recursor contains a denial of service bug which can be exploited remotely. This bug, which we believe to only lead to a crash, has been fixed in 3.0.1. There are no guarantees however, so an upgrade from 3.0 is highly recommended.

All versions of PowerDNS before 2.9.21.1 do not respond to certain queries. This in itself is not a problem, but since the discovery by Dan Kaminsky of a new spoofing technique, this silence for queries PowerDNS considers invalid, within a valid domain, allows attackers more chances to feed *other* resolvers bad data.

All versions of PowerDNS before 2.9.18 contain the following two bugs, which only apply to installations running with the LDAP backend, or installations providing recursion to a limited range of IP addresses. If any of these apply to you, an upgrade is highly advised:

- The LDAP backend did not properly escape all queries, allowing it to fail and not answer questions. We have not investigated further risks involved, but we advise LDAP users to update as quickly as possible (Norbert Sendetzky, Jan de Groot)
- Questions from clients denied recursion could blank out answers to clients who are allowed recursion services, temporarily. Reported by Wilco Baan. This would've made it possible for outsiders to blank out a domain temporarily to your users. Luckily PowerDNS would send out SERVFAIL or Refused, and not a denial of a domain's existence.

All versions of PowerDNS before 2.9.17 are known to suffer from remote denial of service problems which can disrupt operation. Please upgrade to 2.9.17 as this page will only contain detailed security information from 2.9.17 onwards.

CHAPTER

SIXTEEN

UPGRADE GUIDE

Before upgrading, it is advised to read the *Changelogs*. When upgrading several versions, please read **all** notes applying to the upgrade.

16.1 5.0.6 to 5.1.0 and master

The recursor.conf configuration file may contain YAML configuration syntax and new installs using our packages from repo.powerdns.com will install a configuration file using YAML syntax. Note to third-party package maintainers: please start doing the same.

16.1.1 New settings

- All settings that can be set in the Lua config now can alternatively be set in YAML. See *PowerDNS Recursor New Style (YAML) Settings*.
- The *new-domain-db-snapshot-interval* settings has been introduced to set the interval of NOD DB snapshots taken.
- The *proxy-protocol-exceptions* setting has been introduced to exempt addresses from using the proxy protocol.
- The *system-resolver-ttl* setting has been introduced to set the TTL of the system resolver. The system resolver can be used to resolve forwarding names.
- The *system-resolver-interval* setting has been introduced to set the interval of resolve checks done by the system resolver.
- The *system-resolver-self-resolve-check* setting has been introduced to disable to discovery of self-resolving configurations.
- The *max-chain-length* setting has been introduced to limit the maximum number of queries that can be attached to an outgoing request chain.
- The *max-cnames-followed* setting has been introduced to limit the length of CNAME chains followed. Previously this limit was fixed at 10.
- The *new-domain-ignore-list-file*, *unique-response-ignore-list* and *unique-response-ignore-list-file* settings have been introduced to filter names reported by the NOD and UDR subsystems.

16.1.2 Changed settings

- The *max-qperq* default value has been lowered to 50, and the qname-minimization special case has been removed.
- Disabling *structured-logging* is no longer supported.
- The *structured-logging-backend* setting has gained the possibility to request JSON formatted output of structured logging information.

16.2 5.0.5 to 5.0.6

16.2.1 Changed settings

- The *max-mthreads* setting will be adjusted to a lower value if the value of `sysctl vm.max_map_count` is too low to support the maximum number of mthread stacks. In this case **`Recursor`** logs an error message including the suggested value of `vm.max_map_count` to not cause lowering of *max-mthreads*.

16.3 5.0.4 to 5.0.5

16.4 Changed settings

- For YAML settings only: the type of the *incoming.edns_padding_from* and *incoming.proxy_protocol_from* has been changed from `String` to `Sequence of Subnet`.

16.5 5.0.2 to 5.0.3, 4.9.3 to 4.9.4 and 4.8.6 to 4.8.7

16.5.1 Known issue solved

The DNSSEC validation issue with the *`zoneToCache()`* function has been resolved and workarounds can be removed.

16.6 5.0.1 to 5.0.2, 4.9.2 to 4.9.3 and 4.8.5 to 4.8.6

16.6.1 Known issues

The *`zoneToCache()`* function fails to perform DNSSEC validation if the zone has more than *max-rrsigs-per-record* RRSIG records at its apex. There are two workarounds: either increase the *max-rrsigs-per-record* to the number of RRSIGs in the zone's apex, or tell *`zoneToCache()`* to skip DNSSEC validation. by adding `dnssec="ignore"`, e.g.:

```
zoneToCache(".", "url", "https://www.internic.net/domain/root.zone", {dnssec=
↪"ignore"})
```

16.6.2 New settings

- The *max-rrsigs-per-record*, *max-nsec3s-per-record*, *max-signature-validations-per-query*, *max-nsec3-hash-computations-per-query*, *aggressive-cache-max-nsec3-hash-cost*, *max-ds-per-zone* and *max-dnskeys* settings have been introduced to limit the amount of work done for DNSSEC validation.

16.7 4.9.0 to 5.0.0

16.7.1 YAML settings

Starting with version 5.0.0-alpha1 the settings file(s) can be specified using YAML syntax. The old-style settings files are still accepted but will be unsupported in a future release. When a `recursor.yml` settings file is encountered it will be processed instead of a `recursor.conf` file. Refer to *PowerDNS Recursor New Style*

(YAML) Settings for details and the *Conversion of old-style settings to YAML format* guide for how to convert old-style settings to the new YAML format.

16.7.2 Rust

Some parts of the Recursor code are now written in Rust. This has impact if you do local builds or are a third-party package maintainer. According to *cargo msrv* the minimum version to compile the Rust code and its dependencies is 1.64. Some distributions ship with an older Rust compiler, see Rustup for a way to install a more recent one. For our package builds, we install a Rust compiler from the `Standalone` section of Other Rust Installation Methods.

16.7.3 New settings

- The *bypass-server-throttling-probability* setting has been introduced to try throttled servers once in a while.
- The *tcp-threads* setting has been introduced to set the number of threads dedicated to processing incoming queries over TCP. Previously either the distributor thread(s) or the general worker threads would process TCP queries.
- The *qname-max-minimize-count* and *qname-minimize-one-label* have been introduced to allow tuning of the parameters specified in **RFC 9156**.
- The *allow-no-rd* has been introduced, default disabled, *disallowing* queries that do not have the `Recursion Desired (RD)` flag set. This is a change in behavior compared to previous releases.
- The setting `ignoreDuplicates` was added to the RPZ loading Lua functions `rpzPrimary()` and `rpzFile()`. If set, duplicate records in RPZs will be allowed but ignored. The default is to fail loading an RPZ with duplicate records.

16.7.4 Changed settings

- The *loglevel* can now be set to a level below 3 (error).
- The *extended-resolution-errors* now defaults to enabled.
- The *nsec3-max-iterations* now defaults to 50.
- Disabling *structured-logging* has been deprecated and will be removed in a future release.

16.8 4.8.0 to 4.9.0

16.8.1 Metrics

The way metrics are collected has been changed to increase performance, especially when many thread are used. This allows for solving a long standing issue that some statistics were not updated on packet cache hits. This is now resolved, but has the consequence that some metrics (in particular response related ones) changed behaviour as they now also reflect packet cache hits, while they did not before. This affects the results shown by `rec_control get-qtypelist` and the `response-by-qtype`, `response-sizes` and `response-by-rcode` items returned by the `/api/v1/servers/localhost/statistics` API endpoint. Additionally, most `RCodes` and `QTypes` that are marked `Unassigned`, `Reserved` or `Obsolete` by IANA are not accounted, to reduce the memory consumed by these metrics.

16.8.2 New settings

- The *packetcache-negative-ttl* settings to control the TTL of negative (NxDomain or NoData) answers in the packet cache has been introduced.
- The *stack-cache-size* setting to control the number of allocated mthread stacks has been introduced.

- The *packetcache-shards* settings to control the number of shards in the packet cache has been introduced.
- The *aggressive-cache-min-nsec3-hit-ratio* setting to control which NSEC3 records are stored in the aggressive NSEC cache has been introduced. This setting can be used to switch off aggressive caching for NSEC3 only.
- The *dnssec-disabled-algorithms* has been introduced to not use DNSSEC algorithms disabled by the platform's security policy. This applies specifically to Red Hat Enterprise Linux 9 and derivatives. The default value (automatically determine the algorithms that are disabled) should work for many cases.
- The setting `includeSOA` was added to the `rpzPrimary()` and `rpzFile()` Lua functions to include the SOA of the RPZ the responses modified by the RPZ.

16.8.3 Changed settings

The first two settings below have effect on the way the recursor distributes queries over threads. In some cases, this can lead to imbalance of the number of queries process per thread. See *Performance Guide*, in particular the *Imbalance* section.

- The *pdns-distributes-queries* default has been changed to `no`.
- The *reuseport* default has been changed to `yes`.
- The *packetcache-ttl* default has been changed to 24 hours.
- The *max-recursion-depth* default has been changed to 16. Before it was, 40, but effectively the CNAME length chain limit (fixed at 16) took precedence. If you increase *max-recursion-depth*, you also have to increase *stack-size*. A starting point of 5k per recursion depth is suggested. Add some extra safety margin to avoid running out of stack.
- The *hint-file* setting gained a new special value to disable refreshing of root hints completely. See *Handling of root hints*.

16.8.4 `rec_control`

The `trace_regex` subcommand has been changed to take a file argument. Refer to *rec_control trace-regex* and *Tracing Queries* for details and example use.

16.9 4.8.1 to 4.8.2

16.9.1 Cache eviction policy

The cache eviction policy for the record and the negative caches has been improved to reduce imbalance between shards. The maximum size of the negative cache is now 1/8th of the size of the record cache and its number of shards is 1/8th of the *record-cache-shards* setting. Previously the size was 1/10th of the record cache size and the number of shards was equal to the number of shards of the record cache. The `rec_control dump-cache` command now prints more information about shards.

16.10 4.7.0 to 4.8.0

16.10.1 Structured logging

All logging (except query tracing) has been converted to structured logging. Switch to old style logging by setting the *structured-logging* setting to `no`. When using `systemd`, structured logging information will be sent to `journald` using formatted text strings that list the key-value pairs and are human readable. Switch to native key-value pair logging (more suitable for automated log processing) by setting *structured-logging-backend* on the command line to `systemd-journal`.

16.10.2 New settings

- The *max-ns-per-resolve* setting to limit the number of NS records processed to resolve a name has been introduced.
- The *serve-stale-extensions* setting to control the new `Serve Stale` feature has been introduced.
- The *record-cache-locked-ttl-perc* setting to control locking of record sets in the record cache has been introduced.
- The *edns-padding-out* setting to control EDNS padding for outgoing DoT has been introduced.
- The *structured-logging-backend* setting to control the type of structured logging to `journald` has been introduced.

16.10.3 `pdns_recursor` changes

THe `--config` command line option now implements the `check`, `default` and `diff` keywords.

16.10.4 `rec_control` changes

The `dump-throttle` and `dump-edns` subcommands no longer produces a table per thread, as the corresponding tables are now shared by all threads. Additionally, the `dump-edns` command now only lists IPs that have a not OK status. The `dump-nsspeeds` command has changed format to make it more readable and lists the last round trip time recorded for each address. The `get-proxymapping-stats` and `get-remotelogger-stats` subcommands have been added.

16.11 4.7.2 to 4.7.3

16.11.1 New settings

- The *max-ns-per-resolve* setting to limit the number of NS records processed to resolve a name has been introduced.

16.12 4.6.2 to 4.7.0

16.12.1 Zone to Cache Changes

The *Zone to Cache* feature now validates `ZONEMD` records. This means that zones containing invalid `ZONEMD` records will be rejected by default, while previously the `ZONEMD` records would be ignored. For more detail, refer to *Zone to Cache*.

16.12.2 Asynchronous retrieval of `AAAA` records for nameservers

If `IPv6` is enabled for outgoing queries using *query-local-address*, the **`Recursor`** will schedule an asynchronous task to resolve `IPv6` addresses of nameservers it did not otherwise learn. These addresses will then be used (in addition to `IPv4` addresses) for future queries to authoritative nameservers. This has the consequence that authoritative nameservers will be contacted over `IPv6` in more case than before.

16.12.3 New Lua Configuration Functions

- The `addAllowedAdditionalQType()` Lua configuration function was added to make the **Recursor** add additional records to answers for specific query types.
- The `addProxyMapping()` Lua configuration function was added to map source addresses to alternative addresses.

16.12.4 Post Resolve FFI Function

A new `postresolve_ffi()` Lua callback function has been introduced.

16.12.5 New settings

- The *save-parent-ns-set* setting has been introduced, enabling fallback cases if the parent `NS` set contains names not in the child `NS` set.
- The *max-busy-dot-probes* settings has been introduced, enabling the **Recursor** probe for `DoT` support of authoritative servers. This is an experimental function, use with care.

16.12.6 `rec_control` changes

The `dump-nsspeeds`, `dump-failedservers` and `dump-non-resolving` subcommands no longer produce a table per thread, as the corresponding tables are now shared by all threads. They also use a better readable and sortable timestamp format.

16.13 4.6.3 to 4.6.4

16.13.1 New settings

- The *max-ns-per-resolve* setting to limit the number of NS records processed to resolve a name has been introduced.

16.14 4.6.1 to 4.6.2

16.14.1 Deprecated and changed settings

- The *hint-file* gained a special value `no` to indicate that no hint file should be processed. The hint processing code is also made less verbose.

16.15 4.5.x to 4.6.1

16.15.1 Offensive language

Using the settings mentioned in *Offensive language* now generates a warning. Please start using the new names.

16.15.2 File descriptor usage

The number of file descriptors used by the Recursor has increased because the Recursor now keeps idle outgoing TCP/DoT connections open for a while. The extra file descriptors used in comparison to previous versions of the Recursor is *tcp-out-max-idle-per-thread* times the number of worker threads (*threads*).

16.15.3 New settings

- The *dot-to-auth-names* setting to list nameservers that should be contacted over DoT has been introduced.
- The *dot-to-port-853* setting to specify that nameservers or forwarders using port 853 should be contacted over DoT has been introduced.
- The *ignore-unknown-settings* setting has been introduced to make it easier to switch between recursor versions supporting different settings.
- The *webserver-hash-plaintext-credentials* has been introduced to avoid keeping cleartext sensitive information in memory.
- The *tcp-out-max-idle-ms*, *tcp-out-max-idle-per-auth*, *tcp-out-max-queries* and *tcp-out-max-idle-per-thread* settings have been introduced to control the new TCP/DoT outgoing connections pooling. This mechanism keeps connections to authoritative servers or forwarders open for later re-use.
- The *structured-logging* setting has been introduced to prefer structured logging (the default) when both an old style and a structured log messages is available.
- The *max-include-depth* setting has been introduced to limit the number of nested `$include` directives while processing a zone file.
- The *allow-notify-for*, *allow-notify-for-file*, *allow-notify-from* and *allow-notify-from-file* settings have been introduced, allowing incoming notify queries to clear cache entries.

16.15.4 Deprecated and changed settings

- The *api-key* and *webserver-password* settings now accept a hashed and salted version (if the support is available in the openssl library used).

16.15.5 Privileged port binding in Docker

In our Docker image, our binaries are no longer granted the `net_bind_service` capability, as this is unnecessary in many deployments. For more information, see the section "Privileged ports" in Docker-README.

16.16 4.5.10 to 4.5.11

16.16.1 New settings

- The *max-ns-per-resolve* setting to limit the number of NS records processed to resolve a name has been introduced.

16.17 4.5.1 to 4.5.2

16.17.1 Deprecated and changed settings

- The *nsec3-max-iterations* default value has been changed from 2500 to 150.

16.18 4.4.x to 4.5.1

16.18.1 Offensive language

Synonyms for various settings names containing `master`, `slave`, `whitelist` and `blacklist` have been introduced.

- For *stats-api-blacklist* use *stats-api-disabled-list*.
- For *stats-carbon-blacklist* use *stats-carbon-disabled-list*.
- For *stats-rec-control-blacklist* use *stats-rec-control-disabled-list*.
- For *stats-snmp-blacklist* use *stats-snmp-disabled-list*.
- For *edns-subnet-whitelist* use *edns-subnet-allow-list*.
- For *new-domain-whitelist* use *new-domain-ignore-list*.
- For *snmp-master-socket* use *snmp-daemon-socket*.
- For the LUA config function `rpzMaster()` use *`rpzPrimary()`*.

Currently, the older setting names are also accepted and used. The next release will start deprecating them. Users are advised to start using the new names to avoid future trouble.

16.18.2 Special domains

Queries for all names in the `.localhost` domain will answer in accordance with **RFC 6761** section 6.3 point 4. That means that they will be answered with `127.0.0.1`, `::1` or a negative response.

16.18.3 `rec_control` command writing to a file

For the commands that write to a file, the file to be dumped to is now opened by the **`rec_control`** command itself using the credentials and the current working directory of the user running **`rec_control`**. A single minus - can be used as a filename to write the data to the standard output stream. Previously the file was opened by the recursor, possibly in its chroot environment.

16.18.4 New settings

- The *extended-resolution-errors* setting has been added, enabling adding EDNS Extended Errors to responses.
- The *refresh-on-ttl-perc* setting has been added, enabling an automatic cache-refresh mechanism.
- The *ecs-ipv4-never-cache* and *ecs-ipv6-never-cache* settings have been added, allowing an overrule of the existing decision whether to cache EDNS responses carrying subnet information.
- The *aggressive-nsec-cache-size* setting has been added, enabling the functionality described in **RFC 8198**.
- The *x-dnssec-names* setting has been added, allowing DNSSEC metrics to be recorded in a different set of counter for given domains.
- The *non-resolving-ns-max-fails* and *non-resolving-ns-throttle-time* settings have been added, allowing the control of the cache of nameservers failing to resolve.
- The *edns-padding-from* and *edns-padding-mode* and *edns-padding-tag* settings have been added, to control how padding is applied to answers sent to clients.
- The *tcp-fast-open-connect* setting has been added, it enables TCP Fast Connect for outgoing connections. Please read *TCP Fast Open Support* before enabling this feature.

16.18.5 Deprecated and changed settings

- The *minimum-ttl-override* and *ecs-minimum-ttl-override* defaults have ben changed from 0 to 1.
- The *spoof-nearmiss-max* default has been changed from 20 to 1.
- The *dnssec* default has changed from `process-no-validate` to `process`.
- The meaning of the *max-packetcache-entries* has changed: previously there was one packet cache instance per worker thread. Since queries incoming over TCP are now also using the packet cache, there is now also one packet cache instance per distributor thread. Each cache instance has a size of *max-packetcache-entries* divided by (*threads* + *distributor-threads*).

16.18.6 Removed settings

- The `query-local-address6` setting has been removed. It already was deprecated.

16.19 4.3.x to 4.4.0

16.19.1 Response Policy Zones (RPZ)

To conform better to the standard, RPZ processing has been modified. This has consequences for the points in the resolving process where matches are checked and callbacks are called. See *Response Policy Zones (RPZ)* for details. Additionally a new type of callback has been introduced: `policyEventFilter()`.

16.19.2 Dropping queries from Lua callbacks

The method to drop a query from a Lua callback has been changed. Previously, you could set *rcode* to *pdns.DROP*. See *Callback Semantics* for the new method.

16.19.3 Parsing of unknown record types

The parsing (from zone files) of unknown records types (of the form `\# <length> <hex data>`) has been made more strict. Previously, invalid formatted records could produce inconsistent results.

16.19.4 Deprecated and changed settings

- The *query-local-address* setting has been modified to be able to include both IPv4 and IPv6 addresses.
- The `query-local-address6` setting is now deprecated.

16.19.5 New settings

- The *dns64-prefix* setting has been added, enabling common cases of DNS64 handling without having to write Lua code.
- The *proxy-protocol-from* and *proxy-protocol-maximum-size* settings have been added to allow for passing of Proxy Protocol Version 2 headers between a client and the recursor.
- The *record-cache-shards* setting has been added, enabling the administrator to change the number of shards in the records cache. The value of the metric `record-cache-contended` divided by `record-cache-acquired` indicates if the record cache locks are contended. If so, increasing the number of shards can help reducing the contention.

16.20 4.2.x to 4.3.0

16.20.1 Lua Netmask class methods changed

- Netmask class methods `isIpv4` and `isIpv6` have been deprecated in Lua, use `Netmask.isIPv4()` and `Netmask.isIPv6()` instead. In C++ API these methods have been removed.

16.20.2 `socket-dir` changed

The default *socket-dir* has changed to include `pdns-recursor` in the path. For non-chrooted setups, it is now whatever is passed to `--with-socketdir` during configure (`/var/run` by default) plus `pdns-recursor`. The systemd unit-file is updated to reflect this change and systemd will automatically create the directory with the proper permissions. The packaged sysV init-script also creates this directory. For other operating systems, update your init-scripts accordingly.

16.20.3 Systemd service and permissions

The systemd service-file that is installed no longer uses the `root` user to start. It uses the user and group set with the `--with-service-user` and `--with-service-group` switches during configuration, "pdns" on Debian and "pdns-recursor" on CentOS by default. This could mean that PowerDNS Recursor cannot read its configuration, lua scripts, auth-zones or other data. It is recommended to recursively `chown` directories used by PowerDNS Recursor:

```
# For Debian-based systems
chown -R root:pdns /etc/powerdns

# For CentOS and RHEL based systems
chown -R root:pdns-recursor /etc/pdns-recursor
```

Packages provided on the PowerDNS Repository will `chown` directories created by them accordingly in the post-installation steps.

16.20.4 New settings

- The *allow-trust-anchor-query* setting has been added. This setting controls if negative trust anchors can be queried. The default is *no*.
- The *max-concurrent-requests-per-tcp-connection* has been added. This setting controls how many requests are handled concurrently per incoming TCP connection. The default is 10.
- The *max-generate-steps* setting has been added. This sets the maximum number of steps that will be performed when loading a BIND zone with the `$GENERATE` directive. The default is 0, which is unlimited.
- The *nothing-below-nxdomain* setting has been added. This setting controls the way cached NXDOMAIN replies imply non-existence of a whole subtree. The default is *dnssec* which means that only DNSSEC validated NXDOMAINS results are used.
- The *qname-minimization* setting has been added. This options controls if QName Minimization is used. The default is *yes*.

16.21 4.1.x to 4.2.0

Two new settings have been added:

- `xpf-allow-from` can contain a list of IP addresses ranges from which XPF (X-Proxied-For) records will be trusted.

- `setting-xpf-rr-code` should list the number of the XPF record to use (in lieu of an assigned code).

16.22 4.0.x to 4.1.0

loglevel defaulted to 4 but was always overridden to 6 during the startup. The issue has been fixed and the default value set to 6 to keep the behavior consistent.

The `--with-libsodium` configure flag has changed from 'no' to 'auto'. This means that if libsodium and its development header are installed, it will be linked in.

16.23 4.0.3 to 4.0.4

One setting has been added to limit the risk of overflowing the stack:

- *max-recursion-depth*: defaults to 40 and was unlimited before

16.24 4.0.0 to 4.0.1

Two settings have changed defaults, these new defaults decrease CPU usage:

- *root-nx-trust* changed from "no" to "yes"
- *log-common-errors* changed from "yes" to "no"

CHAPTER

SEVENTEEN

CHANGELOGS

The changelogs for the recursor are split between release trains.

Before upgrading, it is advised to read the *Upgrade Guide*.

17.1 Changelogs for 5.1.X

Before upgrading, it is advised to read the *Upgrade Guide*.

17.1.1 5.1.1

Released: 23rd of July 2024

Improvements

- Limit the number of async tasks pushed to resolve NS names and optimizer processing of additionals. ¶ References: #14499, pull request 14501
- Move default Docker config to YAML. ¶ References: #14459, pull request 14477

Bug Fixes

- Fix maintenanceCalls vs maintenanceCount in SNMP MIB. ¶ References: #14514, pull request 14516
- Dump right SOA into dumpFile and report non-relative SOA for includeSOA=true. ¶ References: #14471, pull request 14481
- Yahttp router: avoid unsigned underflow in match(). ¶ References: #14404, pull request 14478

17.1.2 5.1.0

Released: 10th of July 2024

Bug Fixes

- Fix typo in log message. ¶ References: pull request 14435
- Switch el7 builds to Oracle Linux 7 ¶ References: #14400, pull request 14402
- Keep Lua config in Debian/Ubuntu package as existing setups might use it, even though a fresh one does not. ¶ References: #14384, pull request 14389

17.1.3 5.1.0-rc1

Released: 25th of June 2024

Improvements

- Allow recursor.conf file to contain YAML to ease transition to YAML config. ¶ References: #13935, pull request 14265, pull request 14374
- Add nsName into outgoing protobuf request/response messages. ¶ References: pull request 14318
- Do not add UDR field to outgoingProtobuf answer messages ¶ References: pull request 14312
- Add options for ignoring domains for UDR purposes (Ensar Sarajčić). ¶ References: pull request 14275
- Make max CNAME chain length handled settable, previously fixed at 10. ¶ References: pull request 14309

Bug Fixes

- Don't send double SOA record in the case of a dns64 CNAME that does not resolve. ¶ References: #14362, pull request 14373
- dns.cc: use pdns::views::UnsignedCharView. ¶ References: #14356, pull request 14359
- Fix TCP case for policy tags set by gettag(_ffi). ¶ References: #13021, pull request 14346
- Fix client remotes count when using proxy protocol. ¶ References: pull request 14340

17.1.4 5.1.0-beta1

Released: 6th of June 2024

Improvements

- Add a few more fields to the protobuf messages. ¶ References: #13020, pull request 14257
- Handle authoritative servers slow to respond when load is high better. ¶ References: pull request 14221, pull request 14258
- Be a bit more strict with respect to positive answers expanded from a wildcard. ¶ References: pull request 14206
- Extra export types for protobuf messages. ¶ References: pull request 14111
- Various code cleanups and Coverity prompted fixes. ¶ References: pull request 14259, pull request 14260, pull request 14262, pull request 14268

17.1.5 5.1.0-alpha1

Released: 15th of May 2024

Improvements

- Add possibility to set existing Lua config in YAML settings. ¶ References: pull request 13819
- Tidy iputils.hh and iputils.cc ¶ References: pull request 14097, pull request 14139
- Add interface (not subject to proxy protocol substitutions) addresses in Lua DNSQuestion and corresponding FFI. ¶ References: #13730, pull request 14023

- Add setting to exclude specific listen socket addresses from requiring proxy protocol. ¶ References: #13948, pull request 14018
- Use shared NOD (and/or UDR) DB, to avoid multiple copies in memory and on disk. ¶ References: #13677, pull request 13969
- Add feature to allow names (resolved by system resolver) in forwarding config. ¶ References: #11393, pull request 13921
- Enable 64-bit time_t on 32-bit systems with glibc-2.34 (Sven Wegener). ¶ References: pull request 10933
- Remove the possibility to disable structured logging. ¶ References: pull request 13844
- Add structured logging backend that uses JSON representation. ¶ References: pull request 13842
- Tidy recursor-lua4.cc and recursor-lua4.hh. ¶ References: pull request 13889
- Support v6 in FrameStreamLogger, including tidy. ¶ References: pull request 13864
- Tidy rpzloader.cc and .hh. ¶ References: pull request 13861
- Log if a dnssec related limit was hit (if log_bogus is set). ¶ References: pull request 13824
- Tidy ResolveContext class. ¶ References: pull request 13746
- Tidy filterpo.?? (reaching into iputils.hh as well). ¶ References: pull request 13744
- Introduce command to set aggressive NSEC cache size. ¶ References: #13265, pull request 13504
- RPZ from primary refactor and allow notifies for RPZs ¶ References: #12777, pull request 13701
- Use ref wrapper instead of raw pointer in variant. ¶ References: pull request 13702
- Fix a few coverity reports. ¶ References: pull request 13706, pull request 13719
- Cleanup of code doing SNMP OID handling. ¶ References: pull request 13711
- Allow out-of-tree builds (Chris Hofstaedtler) ¶ References: pull request 13654
- Fix country()/countryCode() mixup in example Lua Record documentation (Edward Dore) ¶ References: pull request 13714
- MTasker cleanup and move to recursordist. ¶ References: pull request 13652
- Lower default max-qperq limit. ¶ References: #8646, pull request 13566

Bug Fixes

- Configure.ac fixup: do not require bash (Eli Schwartz) ¶ References: pull request 13596
- FDWrapper: Do not try to close negative file descriptors. ¶ References: pull request 14006
- Fixup res-system-resolve.cc on FreeBSD: resolve.h needs netinet/in.h. ¶ References: pull request 13985
- Don't throttle lame servers if they are marked as dontThrottle. ¶ References: pull request 13919
- Fix Coverity 1534473 Unintended sign extension. ¶ References: pull request 13894
- Don't enter wildcard qnames into the cache in the ZoneToCache function. ¶ References: pull request 13866
- Fix Coverity issues in new RPZ code. ¶ References: pull request 13741
- Fix a potential null deref in *MTasker::schedule()*. ¶ References: pull request 13680

17.2 Changelogs for 5.0.X

Before upgrading, it is advised to read the *Upgrade Guide*.

17.2.1 5.0.8

Released: 23rd of July 2024

Improvements

- Limit the number of async tasks pushed to resolve NS names and optimize processing of additionals. ¶ References: #14499, pull request 14502
- dns.cc: use pdns::views::UnsignedCharView. ¶ References: #14359, pull request 14415

Bug Fixes

- Dump right SOA into dumpFile and report non-relative SOA for includeSOA=true. ¶ References: #14471, pull request 14482
- Yahttp router: avoid unsigned underflow in route(). ¶ References: #14404, pull request 14479

misc

- Switch el7 builds to Oracle Linux 7. ¶ References: #134400, pull request 14412

17.2.2 5.0.7

Released: 3rd of July 2024

Bug Fixes

- Remove potential double SOA records if the target of a dns64 name is NODATA. ¶ References: #14373, pull request 14379
- Fix TCP case for policy tags to not produce cached tags in protobuf messages. ¶ References: #14346, pull request 14351
- Count substituted remote in case of proxy protocol. ¶ References: #14340, pull request 14348

17.2.3 5.0.6

Released: 5th of June 2024

Improvements

- YaHTTP: Enforce max # of request fields and max request line size. ¶ References: #14197, pull request 14223
- Report error and adjust max-mthreads when linux map limit (vm.max_map_count) is too low to accomodate resource usage under load. ¶ References: #14185, pull request 14222

17.2.4 5.0.5

Released: 14th of May 2024

Improvements

- Only print Docker config if debug flag is set. ¶ References: #13849, pull request 13988

Bug Fixes

- Do not count RRSIGs using unsupported algorithms toward RRSIGs limit. ¶ References: #14049, pull request 14091
- Correctly count NSEC3s considered when chasing the closest encloser. ¶ References: #13984, pull request 13992
- Let NetmaskGroup parse dont-throttle-netmasks, allowing negations. ¶ References: #13966, pull request 13991
- Fix types of two YAML settings (incoming.edns_padding_from, incoming.proxy_protocol_from) that should be sequences of subnets. ¶ References: #13947, pull request 13990
- Fix trace=fail regression and add regression test for it. ¶ References: #13926, pull request 13989

17.2.5 5.0.4

Released: 24th of April 2024

Bug Fixes

- Security advisory 2024-02: CVE-2024-25583 ¶ References: pull request 14108

17.2.6 5.0.3

Released: 7th of March 2024

Improvements

- Log if a DNSSEC related limit was hit if log_bogus is set. ¶ References: #13824, pull request 13845
- Reduce RPZ memory usage by not keeping the initially loaded RPZs in memory. ¶ References: #13830, pull request 13846

Bug Fixes

- Fix gathering of denial of existence proof for wildcard-expanded names. ¶ References: #13847, pull request 13852
- Fix the zoneToCache regression introduced by SA 2024-01. ¶ References: #13788, pull request 13791

17.2.7 5.0.2

Released: 13th of February 2024

Bug Fixes

- Security advisory 2024-01: CVE-2023-50387 and CVE-2023-50868 ¶ References: pull request 13782

17.2.8 5.0.1

Released: 10th of January 2024, with no changes compared to the second release candidate. Version 5.0.0 was never released publicly.

17.2.9 5.0.0-rc2

Released: 20th of December 2023

Improvements

- Warn that disabling structured logging is now deprecated. ¶ References: #13567, pull request 13645

Bug Fixes

- Fix handling of RUNTIME_DIRECTORY and NOD dirs. ¶ References: #13588, #13612, pull request 13646

17.2.10 5.0.0-rc1

Released: 6th of December 2023

Improvements

- Remove experimental warnings for YAML. ¶ References: pull request 13557
- Disallow (by answering Refused) RD=0 by default. ¶ References: #13386, pull request 13507
- Make syncres code clang-tidy. ¶ References: pull request 13434
- Introduce a setting to allow RPZ duplicates, including a dup handling fix. ¶ References: #12842, pull request 13501
- Update new b-root-server.net addresses in built-in hints. ¶ References: pull request 13387
- Change default of nsec3-max-iterations to 50. ¶ References: pull request 13478
- Warn if truncation occurred dumping the trace. ¶ References: pull request 13477

Bug Fixes

- A single NSEC3 record covering everything is a special case. ¶ References: #13542, pull request 13543
- Document outgoing query counts better, including a small fix. ¶ References: #13463, pull request 13511
- Take into account throttled queries when determining if we had a cache hit. ¶ References: #13483, pull request 13497
- Correctly apply outgoing.tcp_max_queries bound. ¶ References: #13467, pull request 13480

17.2.11 5.0.0-beta1

Released: 10th of November 2023

Improvements

- Be more memory efficient handling RPZ updates. ¶ References: pull request 13462
- Change default of extended-resolution-errors setting to true. ¶ References: pull request 13464
- Move a few settings from recursor to outgoing section. ¶ References: pull request 13455
- For structured logging always log addresses including port. ¶ References: pull request 13446
- Teach configure to check for cargo version and require >= 1.64. ¶ References: pull request 13438
- Tidy cache and only copy values if non-expired entry was found. ¶ References: #12612, pull request 13410
- Add endbr64 instructions in the right spots for OpenBSD/amd64. ¶ References: #13430, pull request 13430, pull request 13432
- Handle stack memory on NetBSD as on OpenBSD (Tom Ivar Helbekkmo) ¶ References: pull request 13408

Bug Fixes

- Fix ubsan error: using a value of 80 for bool. ¶ References: pull request 13468
- Handle serve stale logic in getRootNXTrust(). ¶ References: #13383, pull request 13409

17.2.12 5.0.0-alpha2

Released: 17th of October 2023

Improvements

- Convert API managed config from old style to YAML if YAML settings are active. ¶ References: #12679, #13233, pull request 13362
- If we miss glue–but not for all NS records–try to resolve the missing address records. ¶ References: pull request 13364
- Make QName Minimization parameters from **RFC 9156** settable. ¶ References: pull request 13296
- Conform to **RFC 2181** 10.3: don't allow NS records to point to aliases. ¶ References: pull request 13312
- Do not use Qname Minimization for infra-queries. ¶ References: #8646, pull request 13295
- Implement probabilistic un-throttle. ¶ References: pull request 13289
- Put files generated by settings/generate.py into tarball so package builds do not have to run it. ¶ References: pull request 13290
- Fix packetcache submit refresh task logic. ¶ References: #13266, pull request 13278
- Allow loglevel to be set to levels < 3. ¶ References: #13264, pull request 13277
- Move tcp-in processing to dedicated thread(s). ¶ References: #8394, pull request 13195

Bug Fixes

- If serving stale, wipe CNAME records from cache when we get a NODATA negative response for them. ¶ References: #12395, pull request 13353
- Fix Coverity 1522436 potential dereference of null return value. ¶ References: pull request 13363
- Fix log messages text and levels. ¶ References: pull request 13303, pull request 13311
- Fix sysconfdir handling in new settings code. ¶ References: #13259, pull request 13276
- Fix Coverity 1519054: Using invalid iterator. ¶ References: pull request 13250

17.2.13 5.0.0-alpha1

Released: 13th of September 2023

Improvements

- Rewrite settings code, introducing YAML settings file, using Rust and generated code to implement YAML processing ¶ References: pull request 13008
- Make aggressive cache pruning more effective and more fair. ¶ References: pull request 13209
- Remove make_tuple and make_pair (Rosen Penev). ¶ References: pull request 13208
- Rec: fix a few unused argument warnings (depending on features enabled). ¶ References: pull request 13190
- Change the default for building with net-snmp from *auto* to *no*. ¶ References: pull request 13168
- Channel: Make the blocking parameters of the object queue explicit. ¶ References: #13147, pull request 13155
- Do not assume the records are in a particular order when determining if an answer is NODATA. ¶ References: pull request 13102
- Document default for *webserver-loglevel* (Frank Louwers). ¶ References: pull request 13111
- Remove unused sysv init files. ¶ References: pull request 13087
- Fixes a few performance issues reported by Coverity. ¶ References: pull request 13092
- Highlight why regression tests failed with github annotation (Josh Soref) ¶ References: pull request 13074
- Switch from deprecated ::set-output (Josh Soref). ¶ References: pull request 13073
- Use backticks in rec_control(1) (Josh Soref). ¶ References: pull request 13067
- Clarify why bulktest is failing (Josh Soref). ¶ References: pull request 13068
- Set TTL in getFakePTRRecords. ¶ References: #13011, pull request 13043
- Update settings.rst – clarify edns-subnet-allow-list (Seth Arnold). ¶ References: pull request 13032
- Dnsheader: Switch from bitfield to uint16_t whenever possible. ¶ References: pull request 13026
- Clarify log message for NODATA/NXDOMAIN without AA (Håkan Lindqvist). ¶ References: pull request 12805
- Use arc4random only for random values. ¶ References: pull request 12913, pull request 12931, pull request 12999, pull request 13001, pull request 13022, pull request 13175, pull request 15197
- Update base Debian version in Docker docs (Italo Cunha). ¶ References: pull request 12851
- Delint pdns recursor.cc. ¶ References: pull request 12917
- Include qname when logging skip of step 4 of qname minimization (Doug Freed). ¶ References: pull request 12957
- Fix a set of move optimizations, as suggested by Coverity. ¶ References: pull request 12952
- Silence Coverity 1462719 Unchecked return value from library. ¶ References: pull request 12934
- Fix compile warnings. ¶ References: pull request 12930
- Dns random: add method to get full 32-bits of randomness. ¶ References: pull request 12913
- Reformat and delint arguments.cc and arguments.hh. ¶ References: pull request 12808

Bug Fixes

- Remove Before=nss-lookup.target line from unit file. ¶ References: pull request 13210
- TCPIOHandler: Fix a race when creating the first TLS connections. ¶ References: pull request 13167
- Rec: Include cstdint in mtasker_ucontext.cc, noted by @zeha. ¶ References: pull request 13174

17.3 Changelogs for 4.9.X

17.3.1 4.9.8

Released: 23rd of July 2024

Improvements

- Optimize processing of additionals. ¶ References: #14499, pull request 14503
- Switch el7 builds to Oracle Linux 7. ¶ References: #14400, pull request 14413
- dns.cc: use pdns::views::UnsignedCharView. ¶ References: #14359, pull request 14416

Bug Fixes

- Dump right SOA into dumpFile and report non-relative SOA for includeSOA=true. ¶ References: #14471, pull request 14483
- Yahttp router: avoid unsigned underflow in route(). ¶ References: #14404, pull request 14480

17.3.2 4.9.7

Released: 3rd of July 2024

Bug Fixes

- Remove potential double SOA records if the target of a dns64 name is NODATA. ¶ References: #14373, pull request 14380
- Fix TCP case for policy tags to not produce cached tags in protobuf messages. ¶ References: #14346, pull request 14352

17.3.3 4.9.6

Released: 14th of May 2024

Improvements

- Only print Docker config if debug flag is set. ¶ References: pull request 13993

Bug Fixes

- Do not count RRSIGs using unsupported algorithms toward RRSIGs limit. ¶ References: #14049, pull request 14093
- Correctly count NSEC3s considered when chasing the closest encloser. ¶ References: #13984, pull request 13995
- Fix trace=fail regression and add regression test for it. ¶ References: #13926, pull request 13994

17.3.4 4.9.5

Released: 24th of April 2024

Bug Fixes

- Security advisory 2024-02: CVE-2024-25583 ¶ References: pull request 14109

17.3.5 4.9.4

Released: 7th of March 2024

Improvements

- Update new b-root-server.net addresses in built-in hints. ¶ References: #12897, #13387, pull request 13793

Bug Fixes

- Fix gathering of denial of existence proof for wildcard-expanded names. ¶ References: pull request 13853
- Fix the zoneToCache regression introduced by SA 2024-01. ¶ References: #13788, pull request 13795
- A single NSEC3 record covering everything is a special case. ¶ References: #13543, pull request 13792

17.3.6 4.9.3

Released: 13th of February 2024

Bug Fixes

- Security advisory 2024-01: CVE-2023-50387 and CVE-2023-50868 ¶ References: pull request 13783

17.3.7 4.9.2

Released: 8th of November 2023

Improvements

- Handle stack memory on NetBSD as on OpenBSD. ¶ References: #13408, pull request 13412
- Prevent two cases of copy of data that can be moved. ¶ References: #13092, pull request 13286
- Implement a more fair way to prune the aggressive cache. ¶ References: #13209, pull request 13282

Bug Fixes

- Handle serve stale logic in getRootNXTrust(). ¶ References: #13383, #13409, pull request 13449
- If serving stale, wipe CNAME records from cache when we get a NODATA negative response for them. ¶ References: #13353, pull request 13411
- Remove Before=nss-lookup.target line from systemd unit file. ¶ References: #13210, pull request 13284
- Prevent lookups for unsupported qtypes or rcode != 0 to submit refresh tasks. ¶ References: #13278, pull request 13283
- Do not assume the records are in a particular order when determining if an answer is NODATA. ¶ References: #13102, pull request 13176

17.3.8 4.9.1

Released: 25th of August 2023

Bug Fixes

- Fix code producing json. ¶ References: #13071, pull request 13163
- Replace data in the aggressive cache if new data becomes available. ¶ References: #13106, pull request 13161
- Fix a few typos in log messages. ¶ References: #13151, pull request 13160
- (I)XFR: handle partial read of len prefix. ¶ References: #13105, pull request 13159
- Fix setting of policy tags on packet cache hits. ¶ References: #13021, pull request 13057
- Work around Red Hat 8 misfeature OpenSSL's headers. ¶ References: #12961, pull request 12995
- Stop using the now deprecated ERR_load_CRYPTO_strings() to detect OpenSSL. ¶ References: #12935, pull request 12994

17.3.9 4.9.0

Released: 30th of June 2023

Please review the *Upgrade Guide* before upgrading from versions < 4.9.x.

Bug Fixes

- Fix qname length getting out-of-sync with qname-minimization iteration count. ¶ References: #12963, pull request 12968
- Rewrite and fix loop that checks if algorithms are available. ¶ References: #12933, pull request 12936
- Fix daemonize() to properly background the process. ¶ References: #12928, pull request 12932

17.3.10 4.9.0-rc1

Released: 15nd of June 2023

Please review the *Upgrade Guide* before upgrading from versions < 4.9.x.

Improvements

- Escape key names that are special in the systemd-journal structured logging backend. ¶ References: #12468, pull request 12906
- Add feature to switch off unsupported DNSSEC algos, either automatically or manually. ¶ References: #12890, pull request 12893
- Expose NOD/UDR metrics. ¶ References: #12855, pull request 12896
- Add SOA to RPZ modified answers if configured to do so. ¶ References: #8232, pull request 12883
- Keep track of max depth reached and report it if !quiet. ¶ References: pull request 12898
- Another set of fixes for clang-tidy reports. ¶ References: pull request 12793, pull request 12904

Bug Fixes

- Prevent duplicate C/DNAMEs being included when doing serve-stale. ¶ References: pull request 12900

17.3.11 4.9.0-beta1

Released: 2nd of June 2023

Please review the *Upgrade Guide* before upgrading from versions < 4.9.x.

Improvements

- Introduce a way to completely disable root-refresh. ¶ References: #12848, pull request 12861
- Delint some files to make clang-tidy not report any issue. ¶ References: pull request 12790, pull request 12836, pull request 12837, pull request 12838
- Distinguish between recursion depth and CNAME chain length. ¶ References: pull request 12779, pull request 12862
- Log if the answer was marked variable by SyncRes and if it was stored into the packet cache (if !quiet). ¶ References: pull request 12750

Bug Fixes

- Sanitize d_orig_ttl stored in record cache. ¶ References: pull request 12673
- Fix clang-tidy botch with respect to spelling of "log-fail". ¶ References: #12790, pull request 12829

17.3.12 4.9.0-alpha1

Released: 14th of April 2023

Please review the *Upgrade Guide* before upgrading from versions < 4.9.x.

Improvements

- Cleanup rcode enums: base one is 8 bit unsigned, extended one 16 bit unsigned ¶ References: pull request 12710
- Sharded and shared packet cache. ¶ References: pull request 12594
- More fine-grained capping of packet cache TTL. ¶ References: pull request 12709

- Update Debian packaging for Recursor, including removal of sysv init script (Chris Hofstaedtler). ¶ References: pull request 10072, pull request 12716
- Unify shorthands for seconds in log messages (Josh Soref). ¶ References: pull request 12497
- Validate: Stop passing shared pointers all the way down. ¶ References: pull request 12674
- Re-establish "recursion depth is always increasing" invariant. ¶ References: pull request 12688
- OpenSSL 3.0 compatibility. ¶ References: pull request 12401, pull request 12412, pull request 12462, pull request 12501, pull request 12502, pull request 12513, pull request 12515, pull request 12516, pull request 12524, pull request 12540, pull request 12550
- Only store NSEC3 records in aggressive cache if we expect them to be effective. ¶ References: pull request 12493
- rec_control trace-regex: trace to a file or stdout instead of the general log. ¶ References: pull request 11777
- Unify trace logging code in syncres and validator. ¶ References: pull request 12434
- Stack protector for mthread stacks. ¶ References: pull request 12446, pull request 12695
- Change the way RD=0 forwarded queries are handled. ¶ References: pull request 12425
- Enable FORTIFY_SOURCE=3 when supported by the compiler. ¶ References: pull request 12381
- Introduce a thread-safe version of stringerror(). ¶ References: pull request 12396
- Name recursor threads consistently with a "rec/" prefix. ¶ References: #11138, pull request 12399
- Rec: Warn on high (90%) mthread stack usage. ¶ References: pull request 12373
- Rec: Generate EDE in more cases, specifically on unreachable auths or synthesized results. ¶ References: pull request 12334, pull request 12691, pull request 12698
- Wrap the CURL raw pointers in smart pointers. ¶ References: pull request 12292
- Reorganization: move recursor specific files to recursordist. ¶ References: #12241, pull request 12318
- Introducing TCounters. ¶ References: pull request 12193, pull request 12323, pull request 12348
- If we encounter a loop in QM, continue with the next iteration. ¶ References: #12090, pull request 12120
- More clear trace message for cache-only lookups. ¶ References: #12080, pull request 12121

Bug Fixes

- Rework root priming code to allow multiple addresses per NS. ¶ References: #12486, pull request 12655
- Fix a dnsheader unaligned case. ¶ References: pull request 12672
- Serve-stale-extensions works on 30s so an hour should be 120. (Andreas Jakum) ¶ References: pull request 12554
- Fix doc typo (Matt Nordhoff). ¶ References: pull request 12539
- Logging tweaks (Josh Soref). ¶ References: pull request 12495
- Negcache dump code: close fd on fdopen fail. ¶ References: #12374, pull request 12419
- Be more careful saving errno in makeClientSocket() and closesocket() ¶ References: pull request 12392
- Add the 'parse packet from auth' error message to structured logging. ¶ References: pull request 12368

17.4 Changelogs for 4.8.X

17.4.1 4.8.9

Released: 14 of May 2024

Bug Fixes

- Do not count RRSIGs using unsupported algorithms toward RRSIGs limit. ¶ References: #14049, pull request 14095
- Correctly count NSEC3s considered when chasing the closest encloser. ¶ References: #13984, pull request 13996

17.4.2 4.8.8

Released: 24th of April 2024

Bug Fixes

- Security advisory 2024-02: CVE-2024-25583 ¶ References: pull request 14110

17.4.3 4.8.7

Released: 7th of March 2024

Improvements

- Update new b-root-server.net addresses in built-in hints. ¶ References: #13387, pull request 13796

Bug Fixes

- If serving stale, wipe CNAME records from cache when we get a NODATA negative response for them. ¶ References: #13353, pull request 13797
- Fix the zoneToCache regression introduced by SA 2024-01. ¶ References: pull request 13799
- Fix gathering of denial of existence proof for wildcard-expanded names. ¶ References: #13847, pull request 13854

17.4.4 4.8.6

Released: 13th of February 2024

Bug Fixes

- Security advisory 2024-01: CVE-2023-50387 and CVE-2023-50868 ¶ References: pull request 13784

17.4.5 4.8.5

Released: 25th of August 2023

Bug Fixes

- (I)XFR: handle partial read of len prefix. ¶ References: #13105, pull request 13158
- YaHTTP: Prevent integer overflow on very large chunks. ¶ References: #12892, pull request 13078
- Stop using the now deprecated ERR_load_CRYPTO_strings() to detect OpenSSL. ¶ References: #12935, pull request 13077
- Work around Red Hat 8 misfeature in OpenSSL's headers. ¶ References: #12961, pull request 13076
- Fix setting of policy tags for packet cache hits. ¶ References: #13021, pull request 13056

17.4.6 4.8.4

Released: 29th of March 2023

Bug Fixes

- PowerDNS Security Advisory 2023-02: Deterred spoofing attempts can lead to authoritative servers being marked unavailable. ¶ References: pull request 12700

17.4.7 4.8.3

Released: 7th of March 2023

Improvements

- Change a few logging urgency levels ¶ References: #12495, pull request 12608
- Use correct name for isEntryUsable(). Existing code used the right logic but wrong name. ¶ References: #12347, pull request 12607

Bug Fixes

- Fix serve-stale logic to not cause intermittent high CPU load by:
  - correcting the removal of a negative cache entry,
  - correcting the serve-stale main loop with respect to exception handling and
  - correctly handle negcache entries with serve-state status.

  ¶ References: #12595, #12610, #12611, pull request 12613
- Update validation state after a missing negative indication. ¶ References: #12598, pull request 12609

17.4.8 4.8.2

Released: 31th of January 2023

Improvements

- Make cache cleaning of record an negative cache more fair when under pressure. ¶ References: #12374, pull request 12418
- Do not report "not decreasing socket buf size" as an error. ¶ References: #12333, pull request 12345

Bug Fixes

- Do not use "message" as key, it has a special meaning to systemd-journal. ¶ References: #12467, pull request 12475
- When using serve-stale, wrong data can be returned from negative cache and record cache (zjs604381586). ¶ References: #12395, pull request 12457
- Add the 'parse packet from auth' error message to structured logging. ¶ References: #12368, pull request 12456
- Refresh of negcache stale entry might use wrong qtype (zjs604381586). ¶ References: #12352, pull request 12455
- Do not chain ECS enabled queries, it can cause the wrong scope to be used for outgoing queries. ¶ References: #12407, pull request 12408
- Fix compilation on FreeBSD. Reported by HellSpawn. ¶ References: #12317, pull request 12346
- Properly encode json string containing binary data. ¶ References: #12260, pull request 12344

17.4.9 4.8.1

Released: 20th of January 2023

Bug Fixes

- Avoid unbounded recursion when retrieving DS records from some misconfigured domains. CVE-2023-22617. ¶ References: pull request 12442

17.4.10 4.8.0

Released: 12th of December 2022

Bug Fixes

- Refactor unsupported qtype code and make sure we ServFail on all unsupported qtypes. ¶ References: #12289, pull request 12293
- Infra queries should not use refresh mode. ¶ References: #11376, #11776, #12078, #12219, pull request 12221

17.4.11 4.8.0-rc1

Released: 18th of November 2022

Bug Fixes

- Also consider recursive forward in the "forwarded DS should not end up in negCache" code. ¶ References: #12189, #12199, pull request 12201
- Correct skip record condition in processRecords. ¶ References: #12198, pull request 12200
- Get DS records with QName Minimization switched on. ¶ References: #12175, pull request 12197
- Fix typo in structured logging key. ¶ References: #12194, pull request 12196

17.4.12 4.8.0-beta2

Released: 7th of November 2022

Improvements

- Only replace protobuf logger config objects if the reload changed them. ¶ References: #12063, pull request 12146
- Be more lenient replacing auth by non-auth records in cache. ¶ References: #12140, pull request 12150

Bug Fixes

- Fix SNMP OID numbers for rcode stats. ¶ References: #12155, pull request 12163
- Implement output operator for QTypes, avoids numeric qtypes in trace logs. ¶ References: #12122, pull request 12162
- Handle IXFR connect and transfer timeouts. ¶ References: #12125, pull request 12161
- Log invalid RPZ content when obtained via IXFR. ¶ References: #12081, pull request 12145
- Detect invalid bytes in makeBytesFromHex(). ¶ References: #12066, pull request 12147

17.4.13 4.8.0-beta1

Released: 5th of October 2022

Improvements

- Add support for NOD/UDR notifications using dnstap. ¶ References: pull request 12047
- Protobuf and dnstap metrics, including rec_control subcommand to show them. ¶ References: #11841, pull request 11903, pull request 12049
- Provide metrics for rcode received from authoritative servers. ¶ References: #7164, pull request 11949
- Proxymapping metrics, including rec_control subcommand to show them. ¶ References: #11648, pull request 11866
- Add querytime attribute to Lua DNSQuestion object, to see the time a query was received. ¶ References: pull request 11909
- Enable include-dir by default in RPM builds, to be in line with DEB builds (Frank Louwers). ¶ References: #11766, pull request 11768
- Improve error message when invalid values for *local-address* are provided in recursor config file. ¶ References: pull request 11989
- Enable SNMP support for debian and ubuntu builds. ¶ References: #11999, pull request 12011
- Warn if snmp-agent is set but SNMP support is not available. ¶ References: #11998, pull request 12009
- A few tweaks to structured logging calls. ¶ References: pull request 11959

Bug Fixes

- Fix –config (should be equal to –config=default), followup to #11907. ¶ References: pull request 12048
- Fix compilation of the event ports multiplexer. ¶ References: #12044, pull request 12046

- When an expired NSEC3 entry is seen move it to the front of the expiry queue. ¶ References: pull request 12038
- If new data is auth and existing data is not, replace even if cache locking is active. ¶ References: #11958, pull request 12027

Removals

- Remove XPF support. ¶ References: pull request 11856

17.4.14 4.8.0-alpha1

Released: 23rd of September 2022

Improvements

- Lock record cache entries if enabled by *record-cache-locked-ttl-perc*. ¶ References: pull request 11958
- Use `nullptr` in `getNSEC3PARAM` + init `bool` at call site (Axel Viala). ¶ References: pull request 11957
- Axfr-retriever: abort on chunk with TC set. ¶ References: #11804, pull request 11953
- Clarify return codes for the Lua hooks in the Recursor (Frank Louwers). ¶ References: pull request 11955
- Recursor: Add `--config[=check|=diff|=default]`. ¶ References: pull request 11907
- Implement optional Serve stale functionality, enabled by *serve-stale-extensions*.. ¶ References: pull request 11776
- Implement padding of (DoT) messages to authoritative servers, if set by *edns-padding-out* (default `yes`). ¶ References: pull request 11906
- Log socket directory path if there is a problem. ¶ References: pull request 11800
- Handle Lua script loading errors. ¶ References: pull request 11823
- Stop sending Server: header (Chris Hofstaedtler). ¶ References: #4979, pull request 11813
- Keep time and count metrics when maintenance is called. ¶ References: #6981, pull request 11869
- Consider dns64 processing in more cases than `Rcode == NoError`. ¶ References: pull request 11849
- Set `rec_control_LDFLAGS`, needed for macOS or any platforms where libcrypto is not in default lib path. ¶ References: #11855, pull request 11857
- Replace/remove jQuery (Chris Hofstaedtler) ¶ References: pull request 11812
- Remove unused `jsrender.js` (Chris Hofstaedtler). ¶ References: pull request 11811
- Save the last nameserver speed recorded plus output it in `rec_control dump-nsspeeds`. ¶ References: #11736, pull request 11780
- Set `TCP_NODELAY` on in and outgoing TCP. ¶ References: #11734, pull request 11754
- Remove > 5 check on TTL of glue from the cache. ¶ References: pull request 11744
- Structured logging for various subsystems. ¶ References: pull request 11631, pull request 11642, pull request 11654, pull request 11662, pull request 11681, pull request 11693, pull request 11710, pull request 11714, pull request 11854
- Make edns table a sparse table. ¶ References: pull request 11704, pull request 11779
- Shared ednsmap. ¶ References: pull request 11601
- Load IPv6 entries from etc-hosts file. ¶ References: #2248, pull request 11682

- Use `systemd-journal` for structured logging if it is available and set by *structured-logging-backend*. ¶ References: #11705, #11706, pull request 11660, pull request 11709
- Fix typos in stats log messages (Matt Nordhoff). ¶ References: #11654, #11671, pull request 11671, pull request 11680
- Shared throttle map. ¶ References: pull request 11598
- Adaptive root refresh interval, normally at 80% of *max-cache-ttl*. ¶ References: pull request 11381

Bug Fixes

- Libssl: Properly load ciphers and digests with OpenSSL 3.0. ¶ References: #11853, pull request 11862
- rec_control: test for `--version` before requiring an argument. ¶ References: #11864, pull request 11867
- Make rec zone files with trailing dot (phonedph1). ¶ References: pull request 11672
- Handle file related errors initially loading Lua script. ¶ References: #10079, #11818, pull request 11820

17.5 Changelogs for 4.7.X

17.5.1 4.7.6

Released: 25th of August 2023

Bug Fixes

- (I)XFR: handle partial read of len prefix. ¶ References: #13105, pull request 13157
- YaHTTP: Prevent integer overflow on very large chunks. ¶ References: #12892, pull request 13079
- Work around Red Hat 8 misfeature in OpenSSL's headers. ¶ References: #12961, pull request 13075
- Fix setting of policy tags for packet cache hits. ¶ References: #13021, pull request 13058

17.5.2 4.7.5

Released: 29th of March 2023

Bug Fixes

- PowerDNS Security Advisory 2023-02: Deterred spoofing attempts can lead to authoritative servers being marked unavailable. ¶ References: pull request 12701

17.5.3 4.7.4

Released: 25th of November 2022

Bug Fixes

- Fix compilation of the event ports multiplexer. ¶ References: #12046, pull request 12231
- Correct skip record condition in processRecords. ¶ References: #12198, pull request 12230
- Also consider recursive forward in the "forwarded DS should not end up in negCache code." ¶ References: #12189, #12199, pull request 12227

- Timeout handling for IXFRs as a client. ¶ References: #12125, pull request 12190
- Detect invalid bytes in makeBytesFromHex(). ¶ References: #12066, pull request 12173
- Log invalid RPZ content when obtained via IXFR. ¶ References: #12081, pull request 12171
- When an expired NSEC3 entry is seen, move it to the front of the expiry queue. ¶ References: #12038, pull request 12168

17.5.4 4.7.3

Released: 20th of September 2022

Improvements

- For zones having many NS records, we are not interested in all so take a sample. ¶ References: #11904, pull request 11936
- Also check qperq limit if throttling happened, as it increases counters. ¶ References: #11848, pull request 11897

Bug Fixes

- Failure to retrieve DNSKEYs of an Insecure zone should not be fatal. ¶ References: #11890, pull request 11940
- Fix recursor not responsive after Lua config reload. ¶ References: #11850, pull request 11879
- Clear the caches *after* loading authzones. ¶ References: #11843, pull request 11847
- Resize answer length to actual received length in udpQueryResponse. ¶ References: #11773, pull request 11774

17.5.5 4.7.2

Released: 23th of August 2022

Bug Fixes

- PowerDNS Security Advisory 2022-02: incomplete exception handling related to protobuf message generation. ¶ References: pull request 11874, pull request 11877

17.5.6 4.7.1

Released: 8th of July 2022

Improvements

- Allow generic format while parsing zone files for ZoneToCache. ¶ References: #11724, #11726, pull request 11750
- Force gzip compression for debian packages (Zash). ¶ References: #11735, pull request 11740

Bug Fixes

- Run tasks from housekeeping thread in the proper way, causing queued DoT probes to run more promptly. Thanks to Jerry Lundström! ¶ References: #11692, pull request 11748

17.5.7 4.7.0

Released: 30th of May 2022

Bug Fixes

- Fix API issue when asking config values for allow-from or allow-notify-from. ¶ References: #11609, pull request 11632

17.5.8 4.7.0-rc1

Released: 6th of May 2022

Bug Fixes

- Prometheus #HELP texts: DNSSEC counters track responses sent, not actual validations performed. ¶ References: #11539, pull request 11559
- Fix DoT port and protocol used for probed authoritative servers. ¶ References: #11541, pull request 11560
- Fix Coverity 1487923 Out-of-bounds read (wrong use of sizeof). ¶ References: #11536, pull request 11538

17.5.9 4.7.0-beta1

Released: 14th of April 2022

Improvements

- Probe authoritative servers for DoT support (experimental). ¶ References: pull request 11487
- Add deferred mode for retrieving additional records. ¶ References: pull request 11492
- Use boost::mult-index for nsspeed table and make it shared. ¶ References: pull request 11484
- Packet cache improvements: do not fill beyond limit and use strict LRU eviction method. ¶ References: pull request 11312
- Use nice format for timestamp printing. ¶ References: pull request 11444
- Only log "Unable to send NOD lookup" if log-common-errors is set. ¶ References: #11440, pull request 11445
- Remember parent NS set, to be able to fallback to it if needed. ¶ References: pull request 11443
- Proxy by table: allow a table based mapping of source address. ¶ References: pull request 11396, pull request 11507

Bug Fixes

- Update moment.min.js (path traversal fix; we are unaffected). ¶ References: pull request 11524
- Prevent segfault with empty allow-from-file and allow-from options (Sven Wegener). ¶ References: pull request 11496
- In the handler thread, call sd_notify() just before entering the main loop in RecursorThread. ¶ References: pull request 11471
- Distinguish between unreachable and timeout for throttling. ¶ References: pull request 11405
- Use correct task to clean outgoing TCP. ¶ References: pull request 11397

17.5.10 4.7.0-alpha1

Released: 28th of February 2022

Improvements

- Add Additional records to query results if appropriate and configured. ¶ References: #11294, pull request 11302
- Resolve AAAA for NS in an async task if applicable. ¶ References: pull request 11294
- Read the base Lua definitions into the Lua context for reading the Lua config. ¶ References: pull request 11319
- Add SNI information to outgoing DoT if available. ¶ References: pull request 11307
- Detect a malformed question early so we can drop it as soon as possible. ¶ References: pull request 11305
- Thread management re-factoring. ¶ References: pull request 11252
- Document changes to policy.DROP better and warn on using the now unsupported way. ¶ References: #11287, pull request 11288
- Allow disabling of processing root hints and lower log level of some related messages. ¶ References: pull request 11283
- Move two maps (failed servers and non-resolving nameservers) from thread_local to shared. ¶ References: pull request 11269
- A CNAME answer on DS query should abort DS retrieval. ¶ References: pull request 11245
- ZONEMD validation for Zone to Cache function. ¶ References: pull request 11100, pull request 11189
- By default, build with symbol visibility hidden. ¶ References: #11178, pull request 11186
- Update protozero to 1.7.1. ¶ References: pull request 11164
- Add Lua postresolve_ffi hook. ¶ References: pull request 11074
- Compute step sizes for Query Minimization according to RFC 9156. ¶ References: pull request 11036

Bug Fixes

- QType ADDR is supposed to be used internally only. ¶ References: #11337, pull request 11338, pull request 11349
- Fix unaligned access in murmur hash code used by the Newly Observed Domain feature. ¶ References: pull request 11347
- A Lua followCNAME result might need native dns64 processing. ¶ References: #11320, pull request 11327
- Use the Lua context stored in SyncRes when calling hooks. ¶ References: #11289, pull request 11300

- Make incoming TCP bookkeeping more correct. ¶ References: #11021, pull request 11030

17.6 Changelogs for 4.6.X

17.6.1 4.6.6

Released: 29th of March 2023

Bug Fixes

- PowerDNS Security Advisory 2023-02: Deterred spoofing attempts can lead to authoritative servers being marked unavailable. ¶ References: pull request 12702

17.6.2 4.6.5

Released: 25th of November 2022

Bug Fixes

- Correct skip record condition in processRecords. ¶ References: #12198, pull request 12229
- Also consider recursive forward in the "forwarded DS should not end up in negCache code." ¶ References: #12189, #12199, pull request 12226
- Timeout handling for IXFRs as a client. ¶ References: #12125, pull request 12191
- Detect invalid bytes in makeBytesFromHex(). ¶ References: #12066, pull request 12172
- Log invalid RPZ content when obtained via IXFR. ¶ References: #12081, pull request 12170
- When an expired NSEC3 entry is seen, move it to the front of the expiry queue. ¶ References: #12038, pull request 12167

17.6.3 4.6.4

Released: 20th of September 2022

Improvements

- For zones having many NS records, we are not interested in all so take a sample. ¶ References: #11904, pull request 11937
- Also check qperq limit if throttling happened, as it increases counters. ¶ References: #11848, pull request 11898

Bug Fixes

- Failure to retrieve DNSKEYs of an Insecure zone should not be fatal. ¶ References: #11890, pull request 11941
- Resize answer length to actual received length in udpQueryResponse. ¶ References: #11773, pull request 11775

17.6.4 4.6.3

Released: 23th of August 2022

Bug Fixes

- PowerDNS Security Advisory 2022-02: incomplete exception handling related to protobuf message generation. ¶ References: pull request 11874, pull request 11876
- Fix API issue when asking config values for allow-from or allow-notify-from. ¶ References: pull request 11609, pull request 11633

17.6.5 4.6.2

Released: 4th of April 2022

Improvements

- Allow disabling of processing the root hints. ¶ References: #11283, pull request 11360
- Log an error if pdns.DROP is used as rcode in Lua callbacks. ¶ References: #11288, pull request 11361
- A CNAME answer on DS query should abort DS retrieval. ¶ References: #11245, pull request 11358
- Reject non-apex NSEC(3)s that have both the NS and SOA bits set. ¶ References: #11225, pull request 11357
- Fix build with OpenSSL 3.0.0. ¶ References: pull request 11260
- Shorter thread names. ¶ References: #11137, pull request 11170
- Two more features to print (DoT and scrypt). ¶ References: #11109, pull request 11169

Bug Fixes

- Be more careful using refresh mode only for the record asked. ¶ References: #11371, pull request 11418
- Use the Lua context stored in SyncRes when calling hooks. ¶ References: #11300, pull request 11380
- QType ADDR is supposed to be used internally only. ¶ References: #11338, pull request 11363
- If we get NODATA on an AAAA in followCNAMERecords, try native dns64. ¶ References: #11327, pull request 11362
- Initialize isNew before calling a exception throwing function. ¶ References: #11257, pull request 11359

17.6.6 4.6.1

Released: 25th of March 2022

This is a security fix release for *PowerDNS Security Advisory 2022-01*. Additionally, because CentOS 8 is End Of Life now, we have switched those builds to Oracle Linux 8. The resulting packages are compatible with RHEL and all derivatives.

Bug Fixes

- Fix validation of incremental zone transfers (IXFRs). ¶ References: pull request 11458

17.6.7 4.6.0

Released: 17th of December 2021

Improvements

- Do not generate event trace records for Lua hooks if no Lua hook is defined. ¶ References: pull request 11091
- Remove capability requirements from Docker images. ¶ References: pull request 11092

17.6.8 4.6.0-rc1

Released: 3rd of December 2021

Bug Fixes

- Condition to HAVE_SYSTEMD_WITH_RUNTIME_DIR_ENV is reversed. During build, the runtime directory in the service files for virtual-hosting are now correctly generated. ¶ References: #10982, pull request 11055
- Do cache negative answers, even when the response was ECS-scoped. ¶ References: #10994, #11010, pull request 11025
- Fix logic botch in TCP code introduced by notify handling in 4.6.0-beta2. ¶ References: #11018, pull request 11022
- Include sys/time.h; needed on musl. ¶ References: #11005, pull request 11016

17.6.9 4.6.0-beta2

Released: 17th of November 2021

Improvements

- Add support for NOTIFY queries to wipe cache entries (Kevin P. Fleming). ¶ References: #7014, pull request 10751

Bug Fixes

- Return the proper extended error code on specific validation failures. ¶ References: #10936, pull request 10980
- We need a libcurl dev lib for the zone-to-cache function. ¶ References: pull request 10971

17.6.10 4.6.0-beta1

Released: 9th of November 2021

Improvements

- Return documented reply on /api/v1 access. ¶ References: pull request 10865
- Add more UDP error metrics (checksum, IPv6). ¶ References: #10852, pull request 10919
- Move to a stream based socket for the control channel. ¶ References: pull request 10930, pull request 10965
- ZoneParserTNG: Stricter checks when loading a zone file. ¶ References: pull request 10901
- Implement fd-usage metric for OpenBSD. ¶ References: pull request 10891

Bug Fixes

- Credentials: EVP_PKEY_CTX_set1_scrypt_salt() takes an *unsigned char**. ¶ References: #10938, pull request 10943
- Fix regression of carbon-ourname. ¶ References: pull request 10926

17.6.11 4.6.0-alpha2

Released: 25th of October 2021

Improvements

- Move to modern C++ constructs (Rosen Penev). ¶ References: pull request 10646, pull request 10868, pull request 10870
- NOD - use structured logging API. ¶ References: pull request 10843
- Sync dnsmessage.proto. ¶ References: pull request 10847
- Introduce experimental Event Trace function to get a more detailed view the work done by the Recursor. ¶ References: #7420, #7558, pull request 10567
- Use packetcache-servfail-ttl for all packet cache entries considered an error reply. ¶ References: #9135, pull request 10797
- Add a periodic zones-to-cache function. ¶ References: pull request 10505, pull request 10794, pull request 10799

Bug Fixes

- Correct appliedPolicyTrigger value for IP matches. ¶ References: pull request 10842
- Use the correct RPZ policy name when loading via XFR. ¶ References: pull request 10768
- Don't create file with wide permissions. ¶ References: pull request 10760
- Update the stats (serial, number of records, timestamp) for RPZ files. ¶ References: pull request 10757

17.6.12 4.6.0-alpha1

Released: 29th of September 2021

Improvements

- TCP/DoT outgoing connection pooling. ¶ References: pull request 10669
- Be more strict when validating DS with respect to parent/child NSEC(3)s. ¶ References: pull request 10599
- Keep a count of per RPZ (or filter) hits. ¶ References: #10554, pull request 10605
- Modify per-thread cpu usage stats to be Prometheus-friendly. ¶ References: #10735, pull request 10554, pull request 10738
- Refactor almost-expired code and add more detailed stats. ¶ References: pull request 10598
- Add dns64 metrics. ¶ References: pull request 10546
- Move macOS to kqueue event handler and assorted compile fixes. ¶ References: #10631, pull request 10634
- Cumulative and Prometheus friendly histograms. ¶ References: #10122, #9077, pull request 10122, pull request 10663
- Rewrite of outgoing TCP code and implement DoT to auth or forwarders. ¶ References: pull request 10428, pull request 10533, pull request 10659
- Switch OpenBSD to kqueue event handler. ¶ References: pull request 10467
- Take into account g_quiet when determining loglevel and change a few loglevels. ¶ References: #10395, pull request 10396
- Move to tcpiohandler for outgoing TCP, sharing much more code with dnsdist. ¶ References: pull request 10349, pull request 10623
- Deprecate offensive setting names. ¶ References: pull request 10288
- Implement structured logging API. ¶ References: pull request 10160
- Disable PMTU for IPv6. ¶ References: pull request 10264
- Move to hashed passwords for the web interface. ¶ References: pull request 10157
- Rec: Add bindings to set arbitrary key-value metadata in logged messages ¶ References: pull request 10491

Bug Fixes

- Only the DNAME records are authoritative in DNAME answers. ¶ References: #10713, pull request 10718
- Pass the Lua context to follow up queries (follow CNAME, dns64). ¶ References: #10632, pull request 10633
- Detect a loop when the denial of the DS comes from the child zone. ¶ References: #10621, pull request 10622
- Process policy and potential Drop action after Lua hooks. ¶ References: pull request 10602
- Do not use DNSKEYs found below an apex for validation. ¶ References: pull request 10565

17.7 Changelogs for 4.5.X

17.7.1 4.5.12

Released: 25th of November 2022

Bug Fixes

- Correct skip record condition in processRecords. ¶ References: #12198, pull request 12228
- Also consider recursive forward in the "forwarded DS should not end up in negCache code." ¶ References: #12189, #12199, pull request 12225
- Timeout handling for IXFRs as a client. ¶ References: #12125, pull request 12192
- Log invalid RPZ content when obtained via IXFR. ¶ References: #12081, pull request 12169
- When an expired NSEC3 entry is seen, move it to the front of the expiry queue. ¶ References: #12038, pull request 12166
- QType ADDR is supposed to be used internally only. ¶ References: #11337, #11338, pull request 12165

17.7.2 4.5.11

Released: 20th of September 2022

Improvements

- For zones having many NS records, we are not interested in all so take a sample. ¶ References: #11904, pull request 11939
- Also check qperq limit if throttling happened, as it increases counters. ¶ References: #11848, pull request 11899

Bug Fixes

- Failure to retrieve DNSKEYs of an Insecure zone should not be fatal. ¶ References: #11890, pull request 11942

17.7.3 4.5.10

Released: 23rd of August 2022

Bug Fixes

- PowerDNS Security Advisory 2022-02: incomplete exception handling related to protobuf message generation. ¶ References: pull request 11874, pull request 11875
- Fix API issue when asking config values for allow-from or allow-notify-from. ¶ References: pull request 11609, pull request 11634

17.7.4 4.5.9

Released: 4th of April 2022

Improvements

- Do cache negative results, even when wasVariable() is true. ¶ References: #10994, #11010, pull request 11024

Bug Fixes

- Be more careful using refresh mode only for the record asked. ¶ References: #11371, pull request 11419
- Use the Lua context stored in SyncRes when calling hooks. ¶ References: #11300, pull request 11384

17.7.5 4.5.8

Released: 25th of March 2022

This is a security fix release for *PowerDNS Security Advisory 2022-01*. Additionally, because CentOS 8 is End Of Life now, we have switched those builds to Oracle Linux 8. The resulting packages are compatible with RHEL and all derivatives.

Bug Fixes

- Fix validation of incremental zone transfers (IXFRs). ¶ References: pull request 11457

17.7.6 4.5.7

Released: 5th of November 2021

Bug Fixes

- A SHA-384 DS should not trump a SHA-256 one, only potentially ignore SHA-1 DS records. ¶ References: #10908, pull request 10912
- rec_control wipe-cache-typed should check if a qtype arg is present and valid. ¶ References: #10905, pull request 10911
- Put the correct string into appliedPolicyTrigger for Netmask matching rules. ¶ References: #10842, pull request 10863

17.7.7 4.5.6

Released: 11th of October 2021

Bug Fixes

- Do not use DNSKEYs found below an apex for validation. ¶ References: #10565, pull request 10806
- Detect a loop when the denial of the DS comes from the child zone. ¶ References: #10622, pull request 10807
- Match ordering of PacketID using the Birthday vs non-Birthday comparator. ¶ References: #10632, pull request 10809
- Pass the Lua context to follow up queries (follow CNAME, dns64). ¶ References: #10633, pull request 10811
- Only the DNAME records are authoritative in DNAME answers. ¶ References: #10718, pull request 10813
- Use the correct RPZ policy name for statistics when loading via XFR. ¶ References: #10768, pull request 10803
- Fix the aggressive cache returning duplicated NSEC3 records. ¶ References: #10701, pull request 10717
- NS from the cache could be a forwarder, take that into account for throttling decision. ¶ References: #10643, pull request 10655

- Check in more places if the policy has been updated before using or modifying it. ¶ References: #10627, pull request 10629

17.7.8 4.5.5

Released: 30th of July 2021

Improvements

- Work around clueless servers sending AA=0 answers. ¶ References: #10555, pull request 10564

Bug Fixes

- Ancestor NSEC3s can only deny the existence of a DS. ¶ References: #10587, pull request 10593
- Make really sure we did not miss a cut on validation failure. ¶ References: #10570, pull request 10575
- Clear the current proxy protocol values each iteration. ¶ References: #10515, pull request 10573

17.7.9 4.5.4

Released: 2nd of July 2021, 4.5.3 was never released publicly.

Bug Fixes

- Make sure that we pass the SOA along the NSEC(3) proof for DS queries. ¶ References: pull request 10519

17.7.10 4.5.2

Released: 9th of June 2021

Improvements

- Change nsec3-max-iterations default to 150. ¶ References: #10440, pull request 10477
- For the NOD lookup case, we don't want QName Minimization. ¶ References: #10420, pull request 10422

Bug Fixes

- Don't follow referral from the parent to the child for DS queries. ¶ References: #10460, pull request 10476
- When refreshing, do not consider root almost expired. ¶ References: #10426, pull request 10475
- Take into account q_quiet when determining loglevel and change a few loglevels. ¶ References: #10396, pull request 10474
- Only add the NSEC and RRSIG records once in wildcard NODATA answers. ¶ References: #10350, pull request 10473

17.7.11 4.5.1

Released: 11th of May 2021

Bug Fixes

- Prevent a race in the aggressive NSEC cache. ¶ References: pull request 10377

17.7.12 4.5.0

Released: Never released publicly.

Bug Fixes

- Apply dns64 on RPZ hits generated after a gettag_ffi hit. ¶ References: pull request 10353

17.7.13 4.5.0-rc1

Released: 28th of April 2021

Improvements

- Boost 1.76 containers: use standard exceptions. ¶ References: #10329, pull request 10335
- Fix wording in edns-padding-tag help. ¶ References: #10318, pull request 10334
- Improve packet cache size computation now that TCP answers are also cached. ¶ References: #10312, pull request 10333
- Print the covering NSEC in tracing log. ¶ References: #10298, pull request 10307

Bug Fixes

- Do not put results of DS query for auth or forward domains in negcache. ¶ References: #10317, pull request 10320
- Use the correct ECS address when proxy-protocol is enabled. ¶ References: #10303, pull request 10319
- Exception loading the RPZ seed file is not fatal. ¶ References: #10291, pull request 10306
- RPZ dumper: stop generating double zz labels on networks that start with zeroes. ¶ References: #10286, pull request 10305

17.7.14 4.5.0-beta2

Released: 14th of April 2021

Improvements

- Log local IP in dnstap messages. ¶ References: #10268, pull request 10280
- Also disable PMTU for IPv6. ¶ References: #10264, pull request 10279

Bug Fixes

- Clear "from" in record cache if we don't know where the update came from. ¶ References: #10232, pull request 10278
- Better handling of stranded DNSKeys. ¶ References: #10223, pull request 10277

17.7.15 4.5.0-beta1

Released: 26th of March 2021

Improvements

- Support TCP FastOpen connect on outgoing connections. ¶ References: #7982, pull request 9995
- Implement EDNS0 padding (rfc7830) for outgoing responses. ¶ References: pull request 8918
- Get rid of early zone cut computation when doing DNSSEC validation. ¶ References: pull request 10057
- Insert hints as non-auth into cache. ¶ References: #10177, pull request 10182
- Don't pick up random root NS records from AUTHORITY sections. ¶ References: #10125, pull request 10178
- Using DATA to report memory usage is unreliable, start using RES instead, as it seems reliable and relevant. ¶ References: #7591, pull request 10161

Bug Fixes

- Make sure we take the right minimum for the packet cache TTL data. ¶ References: pull request 10185

17.7.16 4.5.0-alpha3

Released: 9th of March 2021

Improvements

- Check sizeof(time_t) to be at least 8. ¶ References: pull request 10010
- Change dnssec default to *process*. ¶ References: pull request 10118
- Implement rfc 8198 - Aggressive Use of DNSSEC-Validated Cache. ¶ References: pull request 10047
- Be less verbose telling we are looking up CNAMEs or DNAMEs while tracing. ¶ References: pull request 10112
- Add validation state to protobuf message. ¶ References: #8587, pull request 10113
- Add Policy Kind / RPZ action to Protobuf messages. ¶ References: #9653, #9654, pull request 10109
- Count DNSSEC stats for given names in a different set of counters. ¶ References: #10058, pull request 10089
- Remember non-resolving nameservers. ¶ References: pull request 10096
- Pass an fd to dump to from rec_control to the recursor. ¶ References: pull request 9468
- Introduce settings to never cache EDNS Client (v4/v6) Subnet carrying replies. ¶ References: pull request 10075
- Change spoof-nearmiss-max default to 1. ¶ References: #9845, pull request 10077
- Add missing entries to Prometheus metrics. ¶ References: #10021, pull request 10022
- Also use packetcache for tcp queries. ¶ References: pull request 9990
- Document taskqueue metrics and add them to SNMP MIB. ¶ References: #10009, pull request 10020
- Treat the .localhost domain as special. ¶ References: pull request 9996

Bug Fixes

- Handle policy (if needed) after postresolve and document the hooks better. ¶ References: #10080, pull request 10111
- Return current rcode instead of 0 if there are no CNAME records to follow. ¶ References: #9547, pull request 10064

17.7.17 4.5.0-alpha2

Released: This release was never made public.

17.7.18 4.5.0-alpha1

Released: 15th of January 2021

Improvements

- Introduce "Refresh almost expired" a mechanism to keep the record cache warm. ¶ References: #440, pull request 9699
- Use protozero for Protocol Buffer operations in dnsdist, and dnstap/outgoing for the recursor. ¶ References: #9780, #9781, pull request 9630, pull request 9843
- Use a short-lived NSEC3 hashes cache for denial validation. ¶ References: pull request 9856
- Introduce synonyms for offensive language in settings and docs. ¶ References: pull request 9670
- Handle failure to start the web server more gracefully. ¶ References: #9808, pull request 9812
- Switch default TTL override to 1. ¶ References: pull request 9720
- Log the exact Bogus state when 'dnssec-log-bogus' is enabled. ¶ References: pull request 9806 9828
- Switch to TCP in case of spoofing (near-miss) attempts. ¶ References: pull request 9744
- Add support for rfc8914: Extended DNS Errors. ¶ References: pull request 9673
- Two OpenBSD improvements for UDP sockets: port randomization and EAGAIN errors. ¶ References: pull request 9633
- Cleanup of RPZ refresh handling. ¶ References: pull request 9594
- Refactor the percentage computation and use rounding. ¶ References: pull request 9629
- Throttle servers sending invalid data and rcodes. ¶ References: pull request 9571
- Terminate TCP connections instead of 'ignoring' errors. ¶ References: pull request 9572
- Don't parse any config with *–version*. ¶ References: pull request 9569
- Expose typed cache flush via Web API. ¶ References: pull request 9562
- Remove query-local-address6. ¶ References: pull request 9554
- Lua: add backtraces to errors. ¶ References: pull request 8942
- Log the line received from rec_control. ¶ References: pull request 9493
- Shared and sharded neg cache. ¶ References: pull request 9475

Bug Fixes

- Lookup DS entries before CNAME entries. ¶ References: #9621, pull request 9883
- Fix the gathering of denial proof for wildcard-expanded answers. ¶ References: pull request 9793
- Actually discard invalid RRSIGs with too high labels count. ¶ References: pull request 9789
- x-our-latency is a gauge. ¶ References: #9638, pull request 9686
- Make parse ip:port a bit smarter. ¶ References: #7743, pull request 9432
- Fix wipe-cache-typed. ¶ References: pull request 9515
- Detach snmp thread to avoid trouble when trying to quit nicely. ¶ References: pull request 9492

17.8 Changelogs for 4.4.x

17.8.1 4.4.8

Released: 25th of March 2022

This is a security fix release for *PowerDNS Security Advisory 2022-01*. Additionally, because CentOS 8 is End Of Life now, we have switched those builds to Oracle Linux 8. The resulting packages are compatible with RHEL and all derivatives.

Bug Fixes

- Fix validation of incremental zone transfers (IXFRs). ¶ References: pull request 11456

17.8.2 4.4.7

Released: 5th of November 2021

Bug Fixes

- A SHA-384 DS should not trump a SHA-256 one, only potentially ignore SHA-1 DS records. ¶ References: #10908, pull request 10910
- rec_control wipe-cache-typed should check if a qtype argument is present and valid. ¶ References: #10905, pull request 10909

17.8.3 4.4.6

Released: 8th of October 2021

Bug Fixes

- Use the correct RPZ policy name for statistics when loading via XFR. ¶ References: #10768, pull request 10802
- NS from the cache could be a forwarder, take that into account for throttling decision. ¶ References: #10643, pull request 10654
- Check in more places if the policy has been updated before using or modifying it. ¶ References: #10627, pull request 10628

17.8.4 4.4.5

Released: 30th of July 2021

Improvements

- Work around clueless servers sending AA=0 answers. ¶ References: #10555, pull request 10580

17.8.5 4.4.4

Released: 9th of June 2021

Bug Fixes

- Check if we have room before adding zero ECS scope ENDS value. ¶ References: pull request 10390
- Use the correct ECS address when proxy-protocol is enabled. ¶ References: #10303, pull request 10383
- Apply dns64 on RPZ hits generated after a gettag_ffi hit. ¶ References: pull request 10385
- RPZ dumper: stop generating double zz labels on networks that start with zeroes. ¶ References: #10286, pull request 10314
- Exception loading the RPZ seed file is not fatal. ¶ References: #10291, pull request 10313

17.8.6 4.4.3

Released: 31st of March 2021

Improvements

- Use a short-lived NSEC3 hashes cache for denial validation. ¶ References: #9856, pull request 10221
- Pull in libfstrm for el8 build. ¶ References: pull request 10062

Bug Fixes

- More fail-safe handling of Newly Discovered Domain files. ¶ References: #10238, pull request 10240
- Handle policy (if needed) after postresolve. ¶ References: #10111, pull request 10227
- Return current rcode instead of 0 if there are no CNAME records to follow. ¶ References: #10064, pull request 10226
- Lookup DS entries before CNAME entries. ¶ References: #9883, pull request 10224
- Handle failure to start the web server more gracefully. ¶ References: #9812, pull request 10199
- Test that we correctly cap the answer's TTL in expanded wildcard cases. ¶ References: #9970, pull request 10197
- Fix the gathering of denial proof for wildcard-expanded answers. ¶ References: #9793, pull request 10194
- Make sure we take the right minimum for the packet cache TTL data in the SERVFAIL case. ¶ References: #10185, pull request 10192

17.8.7 4.4.2

Released: 14th of December 2020

Improvements

- UUID: Use the non-cryptographic variant of the boost::uuid. ¶ References: pull request 9837
- Keep a cached, valid entry over a fresher Bogus one. ¶ References: pull request 9838
- Ensure socket-dir matches runtime directory on old systemd ¶ References: #9574, pull request 9799
- Move to several distinct Bogus states, for easier debugging. ¶ References: #9597, pull request 9821
- Do not chase CNAME during qname minimization step 4. ¶ References: #9790, pull request 9805

Bug Fixes

- Untangle the validation/resolving qnames and qtypes. ¶ References: #9807, pull request 9825
- APL records: fix endianness problem. ¶ References: #9766, pull request 9774

17.8.8 4.4.1

Released: 25th of November 2020

Improvements

- Allow to specify a name in getMetric() that is used for Prometheus export only. ¶ References: #9651, pull request 9687

Bug Fixes

- Do not add request to a wait chain that's already processed or being processed. ¶ References: #9707, pull request 9719
- Avoid a CNAME loop detection issue with DNS64 ¶ References: #9696, pull request 9710
- Do not send overly long NOD lookups. ¶ References: #9697, pull request 9705
- If a.b.c CNAME x.a.b.c is encountered, switch off QName Minimization. ¶ References: #9680, pull request 9683
- Fix the processing of answers generated from gettag. ¶ References: #9679, pull request 9682

17.8.9 4.4.0

Released: 19th of October 2020

Bug Fixes

- Backport of CVE-2020-25829: Cache pollution. ¶ References: pull request 9605

17.8.10 4.4.0-rc2

Released: 6th of October 2020

Improvements

- Don't parse any config with *–version*. ¶ References: #9569, pull request 9577
- Expose typed cache flush via Web API. ¶ References: #9562, pull request 9576
- Log when going Bogus because of a missing SOA in authority. ¶ References: #9471, pull request 9528
- Raise an exception on invalid content in unknown record. ¶ References: #9497, pull request 9506

Bug Fixes

- When deciding if we are auth in the local auth or forwarding case, DS is special. ¶ References: #9434, pull request 9579
- Fix wipe-cache-typed. ¶ References: #9515, pull request 9557
- Watch the descriptor again after an out-of-order read timeout. ¶ References: #9495, pull request 9526

17.8.11 4.4.0-rc1

Released: 21st of September 2020

Bug Fixes

- Only do QName Minimization for the names inside a forwarded domain. ¶ References: #9448, pull request 9465
- Fix the parsing of *dont-throttle-netmasks* in the presence of *dont-throttle-names*. ¶ References: pull request 9458

17.8.12 4.4.0-beta1

Released: 31st of August 2020

Improvements

- Store RPZ trigger and hit in appliedPolicy and protobuf message and log them in the trace log. ¶ References: pull request 9376
- Apply filtering policies (RPZ) on CNAME chains as well. ¶ References: #9363, pull request 9414
- Fix warning: initialized lambda captures are a C++14 extension. ¶ References: pull request 9411
- Clean some coverity reported cases of exceptions thrown but not caught. ¶ References: pull request 9412
- Export record cache lock (contention) stats via the various channels. ¶ References: pull request 9391
- Allow multiple local data records when doing RPZ IP matching. ¶ References: pull request 9396
- Replace the use of '1' by QClass::IN to improve readability. ¶ References: pull request 9380
- Avoid name clashes on Solaris derived systems. ¶ References: #9279, pull request 9348

Bug Fixes

- Allow some more depth headroom for the no-qname-minimization fallback case. ¶ References: pull request 9375
- If we have an NS in cache, use it in the forwarder case. ¶ References: #9227, pull request 9351
- Disable outgoing v4 when query-local-address has no v4 addresses. ¶ References: pull request 9196
- Resize hostname to final size in getCarbonHostname() (Aki Tuomi). ¶ References: pull request 9343

17.8.13 4.4.0-alpha2

Released: 20th of July 2020

Improvements

- Check that DNSKEYs have the zone flag set. ¶ References: pull request 9308
- Remove redundant toLogString() calls (Chris Hofstaedtler). ¶ References: pull request 9314
- Stop cluttering the global namespace with validation states. ¶ References: pull request 9312
- Use explicit flag for the specific version of c++ we're targeting. ¶ References: pull request 9231
- Use new operator to print states. ¶ References: pull request 9303
- Refuse QType 0 right away, based on rfc6895 section 3.1. ¶ References: pull request 9290
- Specify a storage type for validation states. ¶ References: pull request 9295
- Common TCP write problems should only be logged if wanted. ¶ References: pull request 9289
- Dump the authority records of a negative cache entry as well. ¶ References: pull request 9288
- Alternative way to do "skip cname check" for DS and DNSKEY records ¶ References: #9266, pull request 9272
- Control stack depth when priming. ¶ References: pull request 9267
- Add version 'statistic' to prometheus. ¶ References: pull request 9252
- Cleanup cache cleaner pruneCollection function. ¶ References: pull request 9236
- RPZ policy should override gettag_ffi answer by default. ¶ References: pull request 9203
- Don't copy the records when scanning for CNAME loops. ¶ References: pull request 9216
- Do not use *using namespace std;* . ¶ References: pull request 9213
- More sophisticated CNAME loop detection. ¶ References: #9153, #9194, pull request 9202
- Use std::string_view when available (Rosen Penev). ¶ References: pull request 9207
- Make sure we can install unsigned packages. ¶ References: pull request 9152
- Clarify docs (Josh Soref). ¶ References: pull request 9162
- Ensure runtime dirs for virtual services differ. ¶ References: pull request 9073
- Builder: improve shipped config files (Chris Hofstaedtler). ¶ References: #8094, pull request 9085
- Less negatives in error messages improves readability. ¶ References: pull request 9100
- Boost 1.73 moved boost::bind placeholders to the placeholders namespace. ¶ References: pull request 9070
- Fix useless copies in loop reported by clang++ 10. ¶ References: pull request 9076
- NetmaskTree: do not test node for null, the loop guarantees node is not null. ¶ References: pull request 9078

- Wrap pthread objects ¶ References: pull request 9067
- Get rid of a naked pointer in the /dev/poll event multiplexer. ¶ References: pull request 9053
- Random engine. ¶ References: #9004, pull request 9016

Bug Fixes

- Update proxy-protocol.cc (ihsinme). ¶ References: pull request 9320
- Kill an signed vs unsigned warning on OpenBSD. ¶ References: pull request 9302
- Don't validate a NXD with a NSEC proving that the name is an ENT. ¶ References: pull request 9237
- Fix three shared cache issues. ¶ References: pull request 9226
- Limit the TTL of RRSIG records as well. ¶ References: #9193, pull request 9205
- Avoid throwing an exception in Logger::log(). ¶ References: pull request 9079

17.8.14 4.4.0-alpha1

Released: 22th of April 2020

New Features

- Implement native DNS64 support, without Lua. ¶ References: pull request 8967
- Add custom tags to RPZ hits. ¶ References: pull request 8927
- Allow attaching a 'routing' tag string to a query in lua code and use that tag in the record cache when appropriate. ¶ References: pull request 8910
- Share record cache between threads. ¶ References: pull request 8898
- Add support for Proxy Protocol between dnsdist and the recursor. ¶ References: pull request 8874

Improvements

- Fix warnings with llvm10 and -Wrange-loop-construct (Kirill Ponomarev). ¶ References: pull request 9000
- Fix compilation without deprecated OpenSSL APIs (Rosen Penev). ¶ References: pull request 8985
- Detect {Libre,Open}SSL functions availability during configure. ¶ References: #8739, pull request 8900
- Better handling of reconnections in Remote Logger. ¶ References: pull request 8887
- Add 'queue full' metrics for our remote logger, log at debug only. ¶ References: #8629, pull request 8883
- Update boost.m4 ¶ References: #8875, pull request 8740, pull request 8876
- Keep a masked network in the Netmask class. ¶ References: pull request 8812
- Replace include guard ifdef/define with pragma once (Chris Hofstaedtler). ¶ References: pull request 8631
- YaHTTP: Support bracketed IPv6 addresses ¶ References: pull request 8815
- Rework NetmaskTree for better CPU and memory efficiency (Stephan Bosch). ¶ References: pull request 8355
- RPZ dumpFile/seedFile: store/get SOA refresh on dump/load. ¶ References: pull request 8778
- Add 'IO wait' and 'steal' metrics on Linux. ¶ References: pull request 8783
- DNSName: Don't call strlen() when the length is already known. ¶ References: pull request 8792
- Fix build with gcc-10 (Sander Hoentjen). ¶ References: pull request 8640

Bug Fixes

- Fix compilation of the ports event multiplexer. ¶ References: #9025, pull request 9031
- Init zone's d_priority field. ¶ References: pull request 8830
- QName Minimization sometimes uses 1 label too many. ¶ References: #8697, pull request 8777

17.9 Changelogs for 4.3.x

17.9.1 4.3.7

Released: 22nd of March 2021

Improvements

- Do not chase CNAME during qname minimization step 4. ¶ References: #9790, pull request 9804

Bug Fixes

- Make sure we take the right minimum for the packet cache TTL data in the SERVFAIL case. ¶ References: #10185, pull request 10193

17.9.2 4.3.6

Released: 25th of November 2020

Bug Fixes

- Do not add request to a wait chain that's already processed or being processed. ¶ References: #9707, pull request 9718
- Do not send overly long NOD lookups. ¶ References: #9697, pull request 9706
- Avoid a CNAME loop detection issue with DNS64. ¶ References: #9696, pull request 9702
- If a.b.c CNAME x.a.b.c is encountered, switch off QName Minimization. ¶ References: #9680, pull request 9684
- Previous placeholder fix was incomplete. ¶ References: #9070, pull request 9609

17.9.3 4.3.5

Released: 13th of October 2020

Improvements

- Log when going Bogus because of a missing SOA in authority. ¶ References: pull request 9527

Bug Fixes

- Backport of CVE-2020-25829: Cache pollution. ¶ References: pull request 9604
- Watch the descriptor again after an out-of-order read timeout. ¶ References: #9495, pull request 9525
- Raise an exception on invalid content in unknown records. ¶ References: #9497, pull request 9507
- Boost 1.73 moved boost::bind placeholders to the placeholders namespace.x ¶ References: #9070, pull request 9501
- Fix the parsing of *dont-throttle-netmasks* in the presence of *dont-throttle-names*. ¶ References: #9454, pull request 9457

17.9.4 4.3.4

Released: 8th of September 2020

Improvements

- Ensure runtime dirs for virtual services differ. ¶ References: #9073, pull request 9397

Bug Fixes

- Allow some more depth headroom for the no-qname-minimization fallback case. ¶ References: #9375, pull request 9416
- Resize hostname to final size in getCarbonHostname(). ¶ References: pull request 9367

17.9.5 4.3.3

Released: 17th of July 2020

Bug Fixes

- Validate cached DNSKEYs against the DSs, not the RRSIGs only. ¶ References: #9309, pull request 9330
- Ignore cache-only for DNSKEYs and DS retrieval. ¶ References: #9297, pull request 9329
- A ServFail while retrieving DS/DNSKEY records is just that. ¶ References: #9292, pull request 9328
- Refuse DS records received from child zones. ¶ References: #9188, pull request 9327
- Better exception handling in houseKeeping/handlePolicyHit. ¶ References: #9268, pull request 9305
- Take initial refresh time from loaded zone. ¶ References: #9299, #9301, pull request 9304

17.9.6 4.3.2

Released: 1st of July 2020

Improvements

- Defer the NOD lookup until after the response has been sent. ¶ References: #9142, pull request 9243
- CNAME loop detection. ¶ References: #9194, #9202, #9216, pull request 9248

Bug Fixes

- Backport of CVE-2020-14196: Enforce webserver ACL. ¶ References: pull request 9285
- Copy the negative cache entry before validating it. ¶ References: #9251, pull request 9262
- Fix compilation of the ports event multiplexer. ¶ References: #9031, pull request 9242
- Fix the handling of DS queries for the root. ¶ References: #9151, pull request 9245
- Fix RPZ removals when an update has several deltas. ¶ References: #9172, pull request 9246
- Fix compilation on systems that do not define HOST_NAME_MAX. ¶ References: #9127, pull request 9128
- Fix build with gcc-10. ¶ References: #8640, pull request 9122

misc

- Correct depth increments. ¶ References: #9184, #9192, pull request 9247
- Limit the TTL of RRSIG records as well ¶ References: #9205, pull request 9249

17.9.7 4.3.1

Released: 19th of May 2020

Improvements

- Add ubuntu focal target. ¶ References: pull request 9082

Bug Fixes

- Backport of security fixes for CVE-2020-10995, CVE-2020-12244 and CVE-2020-10030, plus avoid a crash when loading an invalid RPZ. ¶ References: pull request 9115
- RPZ dumpFile/seedFile: store/get SOA refresh on dump/load. ¶ References: #8778, pull request 9048

misc

- Update boost.m4. ¶ References: #8875, pull request 8963

17.9.8 4.3.0

Released: 3rd of March 2020

Improvements

- Only log qname parsing errors when 'log-common-errors' is set. ¶ References: pull request 8870
- Update copyright year. ¶ References: pull request 8863

17.9.9 4.3.0-rc2

Released: 18th of February 2020

Improvements

- Do continue rpz processing if the current policy is passthru. ¶ References: pull request 8827

Bug Fixes

- Refuse NSEC records with a bitmap length > 32. ¶ References: pull request 8831

17.9.10 4.3.0-rc1

Released: 3rd of February 2020

Improvements

- Update boost.m4. ¶ References: pull request 8751
- Explicitly enable dnstap for debian-stretch and buster. ¶ References: pull request 8738
- EPEL 8 now has libfstrm-devel. ¶ References: pull request 8728
- Give an explicit message if something is wrong with socket-dir. ¶ References: pull request 8726

Bug Fixes

- Make `ComboAddress::setPort()` update the current object. ¶ References: pull request 8730
- Fix the evaluation order for filtering policies (RPZ). ¶ References: pull request 8727

17.9.11 4.3.0-beta2

Released: 16th of January 2020

Improvements

- Add the source and destination ports to the protobuf msg. ¶ References: pull request 8704
- Increase default max-qperq. ¶ References: #8646, pull request 8675

Bug Fixes

- Debian postinst / do not fail on user creation if it already exists. ¶ References: pull request 8673
- Parsing *dont-throttle-names* and *dont-throttle-netmasks* as comma separated lists. (costypetrisor) ¶ References: #8676, pull request 8685
- An Opt-Out NSEC3 RR only proves that there is no secure delegation. ¶ References: #8664, pull request 8692
- Fix wrong zoneCuts caused by cache only lookup. ¶ References: #8642, pull request 8670

17.9.12 4.3.0-beta1

Released: 12th of December 2019

Improvements

- Better time based data structures ¶ References: pull request 8571
- QName Minimization is no longer experimental and is now enabled by default. ¶ References: pull request 8477, pull request 8561
- Make threads run until asked to stop. ¶ References: #8518, pull request 8521
- Fix -Wshadow warnings (Aki Tuomi) ¶ References: pull request 8440
- Do RFC 8020 only if cache entry is dnssec validated ¶ References: pull request 8511
- Add a parameter to limit the number of '$GENERATE' steps ¶ References: pull request 8492

Bug Fixes

- Remove duplicate RRs inside a RRSet when computing the signature ¶ References: pull request 8512
- Check return value of dup() and avoid fd leak if if fdopen() fails ¶ References: pull request 8560
- Avoid startup race by setting the state of a thread before starting it. ¶ References: #8558, pull request 8559
- Purge map of failed auths periodically by keeping a last changed timestamp. ¶ References: #7771, pull request 8525
- Avoid mthread race when using the set of rootNSZones. ¶ References: pull request 8510

17.9.13 4.3.0-alpha3

Released: 29th of October 2019

New Features

- Implement RFC 8020 "NXDOMAIN: There Really Is Nothing Underneath" ¶ References: pull request 8367

Improvements

- Update CentOS 6 init script (None) ¶ References: pull request 8463
- Basic validation of $GENERATE parameters ¶ References: pull request 8451
- Add signal handling for SIGTERM and SIGINT in pdns_recursor, if we are PID1 (Frank Louwers) ¶ References: pull request 8344
- Docs: Add small description for pipe backend about distributor-threads (Donatas Abraitis) ¶ References: pull request 8287
- Improve commandline error reporting for non-opts ¶ References: pull request 8290

misc

- Prime NS records of root-servers.net parent (.net) ¶ References: pull request 8470
- Dns64: stop hiding PTR indirection ¶ References: pull request 8433
- Allow multiple simultaneous incoming TCP queries over a connection ¶ References: #8358, pull request 8391
- Add CentOS 8 as builder target ¶ References: pull request 8400
- Fix chmod paths in rules files ¶ References: pull request 8371

- Build Newly Observed Domain (NOD) support by default. ¶ References: pull request 8366
- Rec: chmod/own recursor.conf for the systemd case ¶ References: #8352, pull request 8360
- Fix #8338: Issue with "zz" abbreviation for IPv6 RPZ triggers ¶ References: #8338, pull request 8340
- Retry getrandom() on EINTR ¶ References: pull request 8317
- Recursor webhandler for prometheus metrics (Greg Cockroft) ¶ References: pull request 7758

17.9.14 4.3.0-alpha2

Released: Never released

17.9.15 4.3.0-alpha1

Released: 5th of September 2019

New Features

- Rec: lua pdns_features table ¶ References: pull request 8210
- Builder: add raspbian-buster target ¶ References: pull request 8075
- Rec: export a protobuf incoming response message for timeouts ¶ References: pull request 8000
- Recursor: add devicename field to protobuf messages ¶ References: pull request 8001
- Recursor: don't start as root in systemd ¶ References: pull request 7879
- Rec experimental qname minimization ¶ References: pull request 7757
- Rec: set the query-zone field in the dnstap messages. ¶ References: pull request 7877
- Allow unix domains sockets for dnstap destinations ¶ References: pull request 7868
- DNSTAP logging for queries to, and responses from, auths ¶ References: pull request 7538

Improvements

- Bail out when no context library is available ¶ References: pull request 8122
- Some unneeded float<->double conversions. ¶ References: pull request 8091
- Rec: document that the special-memory-usage stat is excluded by default ¶ References: pull request 8140
- Update boost.m4 ¶ References: #6942, #8084, pull request 7951
- Rec: small speed improvements in the syncres ¶ References: pull request 8010
- Don't create temporary strings to escape dnsname labels ¶ References: pull request 8013
- Add static assertions for the size of the src address control buffer ¶ References: pull request 8007
- Clear cmsg_space(sizeof(data)) in cmsghdr to appease valgrind. ¶ References: #7981, pull request 7996
- Explicitly align the buffer used for cmsgs ¶ References: #7981, pull request 7990
- Silence unused lambda warning (retry) (fwSmit) ¶ References: #7949, pull request 7967
- Rec: clean ups in the syncres::docnamecachelookup code ¶ References: pull request 7945
- All: dnsname, speeds up tostring() conversion ¶ References: pull request 7699
- rec: optimize for large number of filtering policies, empty sections ¶ References: pull request 7904
- Rec: reuse the outgoing query protobuf for the incoming response ¶ References: pull request 7901

- Rec: compare the cachekey type and place first then the name ¶ References: pull request 7905
- Update boost.m4 to the latest version ¶ References: pull request 7862
- Check if -latomic is needed instead of hardcoding (Rosen Penev) ¶ References: pull request 7861
- Rec: small speedups in the recursion 'slow' path ¶ References: pull request 7843
- Add latomic to arc platform (Rosen Penev) ¶ References: pull request 7857
- Eliminate the loop in syncres::getaddrs() ¶ References: pull request 7548

misc

- Rec: fix two coverity issues ¶ References: pull request 8256
- Add missing inc in rpz findclientpolicy loop. ¶ References: pull request 8236
- Fix inverse handler registration logic for snmp. ¶ References: pull request 8227
- Restore the lua binding for dnsname::wirelength() ¶ References: pull request 8142
- Rec docs: fix versionadded for maintenance() ¶ References: pull request 8152
- Fix the rfc1982lessthan template. ¶ References: pull request 8089
- Ensure debian sysv users get set{g,u}id ¶ References: pull request 8034
- Make sure we always compile with boost_cb_enable_debug set to 0 ¶ References: pull request 8067
- Limit compression pointers to 14 bits ¶ References: pull request 8028
- Another time sensistive test fixed with a fixednow construct. ¶ References: #8008, pull request 8047
- Rec: don't go bogus if the auth zone delegation test takes too long ¶ References: pull request 8008
- Rec: fix the export of only outgoing queries or incoming responses ¶ References: pull request 7997
- Fix a few markup issues in our documentation ¶ References: pull request 7946
- Adapt calidns for openbsd and other systems without rcvmmsg(2) ¶ References: pull request 7871
- Rec: better detection of bogus zone cuts for dnssec validation ¶ References: pull request 7928
- suffixmatchtree: fix root removal, partial match of non-leaf nodes ¶ References: pull request 7886
- Rec: don't mix time() and gettimeofday() in our unit tests (again) ¶ References: #6160, #7235, #7883, pull request 7884
- Stubquery: fix handling of optional type arg. ¶ References: pull request 7870
- Fix warnings reported by coverity ¶ References: pull request 7864
- Recursor: log udp tc bits during trace ¶ References: pull request 7841

17.10 Changelogs for 4.2.x

17.10.1 4.2.5

Released: 13th of October 2020

Bug Fixes

- Backport of CVE-2020-25829: Cache pollution. ¶ References: pull request 9603
- Raise an exception on invalid content in unknown records. ¶ References: #9497, pull request 9508

- Boost 1.73 moved boost::bind placeholders to the placeholders namespace. ¶ References: #9070, pull request 9502
- Fix the parsing of *dont-throttle-netmasks* in the presence of *dont-throttle-names*. ¶ References: #9454, pull request 9456
- Resize hostname to final size in getcarbonhostname(). ¶ References: pull request 9368

17.10.2 4.2.4

Released: 17th of July 2020

Bug Fixes

- Validate cached DNSKEYs against the DSs, not the RRSIGs only. ¶ References: #9309, pull request 9334
- Ignore cache-only for DNSKEYS/DS retrieval. ¶ References: #9297, pull request 9333
- A ServFail while retrieving DS/DNSKEY records is just that. ¶ References: #9292, pull request 9332
- Refuse DS records received from child zones. ¶ References: #9188, pull request 9331
- Better exception handling in housekeeping/handlePolicyHit. ¶ References: #9268, pull request 9306

17.10.3 4.2.3

Released: 1st of July 2020

Improvements

- Fix build with gcc-10 ¶ References: #8640, pull request 9123

Bug Fixes

- Backport of CVE-2020-14196: Enforce webserver ACL. ¶ References: pull request 9284
- Copy the negative cache entry before validating it. ¶ References: #9251, pull request 9261
- Fix compilation on systems that do not define HOST_NAME_MAX. ¶ References: #9127, pull request 9133

17.10.4 4.2.2

Released: 19th of May 2020

Improvements

- Add ubuntu focal target. ¶ References: pull request 9081
- Only log qname parsing errors when 'log-common-errors' is set. ¶ References: pull request 8869

Bug Fixes

- Backport of security fixes for CVE-2020-10995, CVE-2020-12244 and CVE-2020-10030, plus avoid a crash when loading an invalid RPZ. ¶ References: pull request 9116
- Refuse NSEC records with a bitmap length > 32. ¶ References: pull request 8832
- Avoid startup race by setting the state of a tread before starting it. ¶ References: pull request 8802
- Better detection of Bogus zone cuts for DNSSEC validation. ¶ References: pull request 8696
- Fix parsing *dont-throttle-names* and *dont-throttle-netmasks* as comma separated lists. ¶ References: pull request 8686

misc

- Update gen-version to use latest tag for version number. ¶ References: pull request 8988
- Update boost.m4. ¶ References: #8875, pull request 8752, pull request 8964
- Debian postinst / do not fail on user creation if it already exists. ¶ References: pull request 8674

17.10.5 4.2.1

Released: 9th of December 2019

Improvements

- Add CentOS 8 as builder target ¶ References: pull request 8427
- Update boost.m4 ¶ References: pull request 8124
- Add deviceName field to protobuf messages ¶ References: #8101, pull request 8187
- Test improvements (Chris Hofstaedtler) ¶ References: #8008, #8047, pull request 8121
- Builder: add raspbian-buster target ¶ References: pull request 8086

Bug Fixes

- Purge map of failed auths periodically by keeping a last changed timestamp. ¶ References: pull request 8552
- Prime NS records of root-servers.net parent (.net) ¶ References: pull request 8528
- Issue with "zz" abbreviation for IPv6 RPZ triggers ¶ References: pull request 8493
- Basic validation of $GENERATE parameters ¶ References: pull request 8452
- Fix inverse handler registration logic for SNMP. ¶ References: pull request 8230

17.10.6 4.2.0

Released: 16th of July 2019

Improvements

- Clear CMSG_SPACE(sizeof(data)) in cmsghdr to appease valgrind ¶ References: #7981, pull request 8005

Bug Fixes

- Make sure we always compile with BOOST_CB_ENABLE_DEBUG set to 0 ¶ References: pull request 8074
- Limit compression pointers to 14 bits ¶ References: pull request 8052

misc

- Fix the export of only outgoing queries or incoming responses ¶ References: pull request 8009

17.10.7 4.2.0-rc2

Released: 25th of June 2019

Improvements

- Compare the CacheKey type and place first then the name ¶ References: pull request 7939

Bug Fixes

- Handle short reads from our random device ¶ References: pull request 7955
- Check if -latomic is needed instead of hardcoding ¶ References: pull request 7953
- Don't mix time() and gettimeofday() in our unit tests ¶ References: pull request 7931
- SuffixMatchTree fixes ¶ References: pull request 7954

17.10.8 4.2.0-rc1

Released: 23th of May 2019

Improvements

- Use `net-snmp-config --netsnmp-agent-libs` instead of `--agent-libs`. ¶ References: pull request 7818

Bug Fixes

- Fix the detection of `snmp_select_info2()`. ¶ References: pull request 7826
- Ensure a valid range to `string()` in `PacketReader::getUnquotedText()` ¶ References: #7272, pull request 7813

17.10.9 4.2.0-beta1

Released: 7th of May 2019

New Features

- Add a new `max-cache-bogus-ttl` option. ¶ References: #7445, pull request 7478
- Implement a way to disallow throttling of auths. ¶ References: pull request 7480
- ECS cache limit with TTL. ¶ References: pull request 7631
- Use a bounded load balancing algo to distribute queries. ¶ References: pull request 7507

Improvements

- Add a `distribution-pipe-buffer-size` setting. ¶ References: pull request 7571
- Add `protobuf-use-kernel-timestamp` for sharper latencies. ¶ References: pull request 7508
- Ignore path MTU discovery on UDP server socket. ¶ References: pull request 7410
- Set `--enable-option-checking=fatal` on all package builds, enable SNMP in RPMS. ¶ References: #7671, pull request 7669
- This provides cpu usage statistics per thread (worker & distributor). ¶ References: pull request 7649
- Add a new `ecs-minimum-ttl-override` setting. ¶ References: pull request 7574
- `Utility::random()` and `srandom()` are not used anymore. ¶ References: pull request 7484
- Add rec statistics about ECS response sizes, API endpoint to get a specific stat. ¶ References: #7498, pull request 7504
- Move back to malloc on !openbsd. Doing mmap/munmap all the time hurts... ¶ References: pull request 7583
- Set `ip(v6)_recverr` socket option to get notified of more than just port unreachable errors on Linux. ¶ References: pull request 7540
- Change the way `getRealMemUsage()` works on Linux (using `statm`). ¶ References: pull request 7502
- Lua: expose `dns_random` as `pdnsrandom`. ¶ References: #6853, pull request 7492
- Add an option to not override custom RPZ types with the default policy. ¶ References: pull request 7476
- Resync YaHTTP code to cmouse/yahttp@11be77a1fc4032. (Chris Hofstaedtler) ¶ References: pull request 7433

Bug Fixes

- Fix DNSSEC validation of non-expanded wildcards. ¶ References: pull request 7714
- Add DNAME support. ¶ References: #6318, pull request 6341
- Move replaced negcache entries to the back of the expunge queue. ¶ References: pull request 7730
- Fix the cache cleaning code being only run once for workers. ¶ References: pull request 7731
- Alternative solution to the unaligned accesses. ¶ References: pull request 7708
- `ednsoptionview` improvements. ¶ References: pull request 7652
- Add missing `getregisteredname` Lua function. (Aki Tuomi) ¶ References: pull request 7589
- Correctly interpret an empty AXFR response to an IXFR query. ¶ References: pull request 7494

17.10.10 4.2.0-alpha1

Released: 1st of February 2019

Initial 4.2.x release, please see the blog post: https://blog.powerdns.com/2019/02/01/changes-in-the-powerdns-recursor-4-2-0/

17.11 Changelogs for 4.1.x

Note: 4.1.x and earlier releases are End of Life and no longer supported. See *EOL Statements*.

17.11.1 4.1.18

Released: 13th of October 2020

Bug Fixes

- Backport of CVE-2020-25829: Cache pollution. ¶ References: pull request 9601

17.11.2 4.1.17

Released: 1st of July 2020

Bug Fixes

- Backport of CVE-2020-14196: Enforce webserver ACL. ¶ References: pull request 9283
- Fix compilation on systems that do not define HOST_NAME_MAX. ¶ References: #8640, #9127, pull request 9129

17.11.3 4.1.16

Released: 19th of May 2020

Improvements

- Only log qname parsing errors when 'log-common-errors' is set. ¶ References: pull request 8868

Bug Fixes

- Backport of security fixes for CVE-2020-10995, CVE-2020-12244 and CVE-2020-10030, plus avoid a crash when loading an invalid RPZ. ¶ References: pull request 9117

misc

- Update python dependencies for docs generation. ¶ References: pull request 8809
- Update boost.m4. ¶ References: pull request 8753

17.11.4 4.1.15

Released: 6th of December 2019

Bug Fixes

- Backport 8525 to rec 4.1.x: Purge map of failed auths periodically by keeping a last changed timestamp ¶ References: pull request 8554
- Backport 8470 to rec 4.1.x: prime NS records of root-servers.net parent (.net) ¶ References: pull request 8544
- Backport 8340 to rec 4.1.x: issue with "zz" abbreviation for IPv6 RPZ triggers ¶ References: pull request 8543
- Backport 7068 to 4.1.x: Do the edns data dump for all threads ¶ References: pull request 8542

misc

- Backport #7951 to 4.1.x: update boost.m4 ¶ References: pull request 8123

17.11.5 4.1.14

Released: 13th of June 2019

Improvements

- Add statistics counters for AD and CD queries. ¶ References: pull request 7906

Bug Fixes

- Add missing getregisteredname Lua function ¶ References: pull request 7912

17.11.6 4.1.13

Released: 21st of May 2019

Improvements

- Add the `disable-real-memory-usage` setting to skip expensive collection of detailed memory usage info. ¶ References: #7661, pull request 7673

Bug Fixes

- Fix DNSSEC validation of wildcards expanded onto themselves. ¶ References: #7714, pull request 7816

17.11.7 4.1.12

Released: 2nd of April 2019

Improvements

- Provide CPU usage statistics per thread (worker & distributor). ¶ References: pull request 7647
- Use a bounded load-balancing algo to distribute queries. ¶ References: #7507, pull request 7634
- Implement a configurable ECS cache limit so responses with an ECS scope more specific than a certain threshold and a TTL smaller than a specific threshold are not inserted into the records cache at all. ¶ References: #7572, #7631, pull request 7651

Bug Fixes

- Correctly interpret an empty AXFR response to an IXFR query. ¶ References: #7494, pull request 7495

17.11.8 4.1.11

Released: 1st of February 2019

Since Spectre/Meltdown, system calls have become more expensive. This made exporting a very high number of protobuf messages costly, which is addressed in this release by reducing the number of syscalls per message.

Improvements

- Add an option to export only responses over protobuf to the Lua `protobufServer()` directive. ¶ References: pull request 7434
- Reduce systemcall usage in protobuf logging. (See #7428.) ¶ References: #7428, pull request 7430

17.11.9 4.1.10

Released: 24th of January 2019

This release fixes a bug when trying to build PowerDNS Recursor with protobuf support disabled, thus this release is only relevant to people building PowerDNS Recursor from source and not if you're installing it as a package from our repositories.

Bug Fixes

- PowerDNS Recursor release 4.1.9 introduced a call to the Lua `ipfilter()` hook that required access to the DNS header, but the corresponding variable was only declared when protobuf support had been enabled. ¶ References: pull request 7403

17.11.10 4.1.9

Released: 21st of January 2019

This release fixes *Security Advisory 2019-01* and *Security Advisory 2019-02* that were recently discovered, affecting PowerDN

- CVE-2019-3806, 2019-01: from 4.1.4 up to and including 4.1.8 ;
- CVE-2019-3807, 2019-02: from 4.1.0 up to and including 4.1.8.

The issues are:

- CVE-2019-3806, 2019-01: Lua hooks are not properly applied to queries received over TCP in some specific combination of settings, possibly bypassing security policies enforced using Lua ;

- CVE-2019-3807, 2019-02: records in the answer section of responses received from authoritative servers with the AA flag not set were not properly validated, allowing an attacker to bypass DNSSEC validation.

Improvements

- Try another worker before failing if the first pipe was full ¶ References: #7383, pull request 7377

Bug Fixes

- Properly apply Lua hooks to TCP queries, even with pdns-distributes-queries set (CVE-2019-3806, PowerDNS Security Advisory *2018-01*). Validates records in the answer section of responses with AA=0 (CVE-2019-3807, PowerDNS Security Advisory *2019-02*). ¶ References: pull request 7397

17.11.11 4.1.8

Released: 26th of November 2018

This release fixes *Security Advisory 2018-09* that we recently discovered, affecting PowerDNS Recursor up to and including 4.1.7.

The issue is that a remote attacker can trigger an out-of-bounds memory read via a crafted query, while computing the hash of the query for a packet cache lookup, possibly leading to a crash.

When the PowerDNS Recursor is run inside a supervisor like supervisord or systemd, a crash will lead to an automatic restart, limiting the impact to a somewhat degraded service.

Bug Fixes

- Crafted query can cause a denial of service (CVE-2018-16855, PowerDNS Security Advisory *2018-09*) ¶ References: pull request 7221

17.11.12 4.1.7

Released: 9th of November 2018

This release updates the mitigation for *Security Advisory 2018-07*, reverting the EDNS fallback strictness increase. This is necessary because there are a lot of broken name servers on the Internet.

Improvements

- Revert 'Keep the EDNS status of a server on FormErr with EDNS' ¶ References: pull request 7172
- Refuse queries for all meta-types ¶ References: pull request 7174

17.11.13 4.1.6

Released: 7th of November 2018

This release reverts #6980, it could lead to DNSSEC validation issues.

Bug Fixes

- Revert "rec: Authority records in AA=1 CNAME answer are authoritative". ¶ References: #7158, pull request 7159

17.11.14 4.1.5

Released: 6th of November 2018

This release fixes the following security advisories:

- PowerDNS Security Advisory *2018-04* (CVE-2018-10851)
- PowerDNS Security Advisory *2018-06* (CVE-2018-14626)
- PowerDNS Security Advisory *2018-07* (CVE-2018-14644)

Improvements

- Add pdnslog to lua configuration scripts (Chris Hofstaedtler) ¶ References: #6848, pull request 6919
- Fix compilation with libressl 2.7.0+ ¶ References: #6943, pull request 6948
- Export outgoing ECS value and server ID in protobuf (if any) ¶ References: #6989, #6991, pull request 7004
- Switch to devtoolset 7 for el6 ¶ References: #7040, pull request 7122
- Allow the signature inception to be off by a number of seconds. (Kees Monshouwer) ¶ References: #7081, pull request 7125

Bug Fixes

- Delay the creation of rpz threads until we have dropped privileges ¶ References: #6792, pull request 6984
- Crafted answer can cause a denial of service (CVE-2018-10851, PowerDNS Security Advisory *2018-04*) ¶ References: pull request 7151
- Packet cache pollution via crafted query (CVE-2018-14626, PowerDNS Security Advisory *2018-06*) ¶ References: pull request 7151
- Crafted query for meta-types can cause a denial of service (CVE-2018-14644, PowerDNS Security Advisory *2018-07*) ¶ References: pull request 7151
- Cleanup the netmask trees used for the ecs index on removals ¶ References: #6960, pull request 6961
- Make sure that the ECS scope from the auth is < to the source ¶ References: #6605, pull request 6963
- Authority records in aa=1 cname answer are authoritative ¶ References: #6979, pull request 6980
- Avoid a memory leak in catch-all exception handler ¶ References: pull request 7073
- Don't require authoritative answers for forward-recurse zones ¶ References: #6340, pull request 6741
- Release memory in case of error in the openssl ecdsa constructor ¶ References: pull request 6917
- Convert a few uses to toLogString to print DNSName's that may be empty in a safer manner ¶ References: #6924, pull request 6925
- Avoid a crash on DEC Alpha systems ¶ References: pull request 6945
- Clear all caches on (N)TA changes ¶ References: #6949, pull request 6951

17.11.15 4.1.4

Released: 31st of August 2018

Improvements

- Split `pdns_enable_unit_tests`. (Chris Hofstaedtler) ¶ References: pull request 6436
- Add a new *max-udp-queries-per-round* setting. ¶ References: pull request 6518
- Fix warnings reported by gcc 8.1.0. ¶ References: pull request 6590
- Tests: replace awk command by perl. ¶ References: pull request 6809
- Allow the snmp thread to retrieve statistics. ¶ References: pull request 6720

Bug Fixes

- Don't account chained queries more than once. ¶ References: #6462, pull request 6465
- Make *rec_control* respect *include-dir*. ¶ References: #6536, pull request 6557
- Load lua scripts only in worker threads. ¶ References: #6567, pull request 6812
- Purge all auth/forward zone data including subtree. (@phonedph1) ¶ References: pull request 6873

17.11.16 4.1.3

Released: 22nd of May 2018

This release improves the stability and resiliency of the RPZ implementation, prevents metrics gathering from slowing down the processing of DNS queries and fixes an issue related to the cleaning of EDNS Client Subnet entries from the cache.

Improvements

- Move carbon/webserver/control/stats handling to a separate thread. ¶ References: pull request 6567
- Use a separate, non-blocking pipe to distribute queries. ¶ References: pull request 6566
- Add a subtree option to the *API* cache flush endpoint. ¶ References: #6550, pull request 6562
- Update copyright years to 2018 (Matt Nordhoff). ¶ References: #6130, #6610, pull request 6611
- Fix a warning on botan >= 2.5.0. ¶ References: #6474, pull request 6478, pull request 6596
- Add `_raw` versions for `QName` / `ComboAddresses` to the `FFI` API. ¶ References: pull request 6583

Bug Fixes

- Respect the `AXFR` timeout while connecting to the `RPZ` server. ¶ References: pull request 6469
- Don't increase the `DNSSEC` validations counters when running with `process-no-validate`. ¶ References: pull request 6467
- Count a lookup into an internal auth zone as a cache miss. ¶ References: pull request 6313
- Delay the loading of `RPZ` zones until the parsing is done, fixing a race condition. ¶ References: #6237, pull request 6588
- Reorder includes to avoid boost `L` conflict. ¶ References: #6358, #6516, #6517, #6542, pull request 6595
- Use canonical ordering in the `ECS` index.

  ¶ References: #6505, pull request 6586
- Add `-rdynamic` to `C{,XX}FLAGS` when we build with `LuaJIT`. ¶ References: pull request 6514, pull request 6630

- Increase `MTasker` stacksize to avoid crash in exception unwinding (Chris Hofstaedtler). ¶ References: #6179, pull request 6418
- Use the SyncRes time in our unit tests when checking cache validity (Chris Hofstaedtler). ¶ References: #6086, pull request 6419
- Disable only our own tcp listening socket when reuseport is enabled ¶ References: #6849, pull request 6850

17.11.17 4.1.2

Released: 29th of March 2018

This release improves the stability and resiliency of the RPZ implementation and fixes several issues related to EDNS Client Subnet.

New Features

- Add FFI version of `gettag()`. ¶ References: pull request 6344

Improvements

- Add the option to set the AXFR timeout for RPZs. ¶ References: pull request 6268, pull request 6290, pull request 6298, pull request 6303
- IXFR: correct behavior of dealing with DNS Name with multiple records and speed up IXFR transaction (Leon Xu). ¶ References: pull request 6172
- Add *RPZ statistics endpoint* to the *API*. ¶ References: #6225, pull request 6379

Bug Fixes

- Retry loading RPZ zones from server when they fail initially. ¶ References: #6238, pull request 6237, pull request 6293, pull request 6336
- Fix ECS-based cache entry refresh code. ¶ References: pull request 6300
- Fix ECS-specific NS AAAA not being returned from the cache. ¶ References: #6319, pull request 6320

17.11.18 4.1.1

Released: 22nd of January 2018

This is the second release in the 4.1 train.

This release fixes PowerDNS Security Advisory *2018-01*.

The full release notes can be read on the blog.

This is a release on the stable branch, containing a fix for the abovementioned security issue and several bug fixes from the development branch.

Improvements

- Don't process records for another class than IN. We don't use records of another class than IN, but we used to store some of them in the cache which is useless. Just skip them. ¶ References: #6198, pull request 6085

Bug Fixes

- Correctly handle ancestor delegation NSEC{,3} for children. Fixes the DNSSEC validation issue found in Knot Resolver, where a NSEC{3} ancestor delegation is wrongly use to prove the non-existence of a RR below the delegation. We already had the correct check for the exact owner name, but not for RRs below the delegation. (Security Advisory *2018-01*) ¶ References: pull request 6215
- Fix the computation of the closest encloser for positive answers. When the positive answer is expanded from a wildcard with NSEC3, the closest encloser is not always parent of the qname, depending on the number of labels in the initial wildcard. ¶ References: #6199, pull request 6092
- Pass the correct buffer size to `arecvfrom()`. The incorrect size could possibly cause DNSSEC failures. ¶ References: #6200, pull request 6095
- Fix to make `primeHints` threadsafe, otherwise there's a small chance on startup that the root-server IPs will be incorrect. ¶ References: #6212, pull request 6209
- Don't validate signature for "glue" CNAME, since anything else than the initial CNAME can't be considered authoritative. ¶ References: #6201, pull request 6137

17.11.19 4.1.0

Released: 4th of December 2017

This is the first release in the 4.1 train.

The full release notes can be read on the blog.

This is a major release containing significant speedups (both in throughput and latency), enhanced capabilities and a highly conformant and robust DNSSEC validation implementation that is ready for heavy production use. In addition, our EDNS Client Subnet implementation now scales effortlessly to networks needing very fine-grained scopes (as used by some 'country sized' service providers).

- Improved DNSSEC support,
- Improved documentation,
- Improved RPZ support,
- Improved EDNS Client Subnet support,
- Support for Botan 2.x (and removal of support for Botan 1.10),
- SNMP support,
- Lua engine has gained access to more parts of the recursor,
- CPU affinity can now be specified,
- TCP Fast Open support,
- New performance metrics.

Changes since 4.1.0-rc3:

Bug Fixes

- Dump the validation status of negcache entries, fix DNSSEC type. ¶ References: pull request 5972
- Fix DNSSEC validation of DS denial from the negative cache. ¶ References: pull request 5978
- Store additional records as non-auth, even on AA=1 answers. ¶ References: pull request 5997
- Don't leak when the loading a public ECDSA key fails. ¶ References: pull request 6008
- When validating DNSKeys, the zone should be part of the signer. ¶ References: pull request 6009
- Cache Secure validation state when inserting negcache entries. ¶ References: pull request 5980

17.11.20 4.1.0-rc3

Released: 17th of November 2017

The third Release Candidate adds support for Botan 2.x (and removes support for Botan 1.10!), has a lot of DNSSEC fixes, features a cleaned up web UI and has miscellaneous minor improvements.

Improvements

- Add the DNSSEC validation state to the `DNSQuestion` Lua object (although the ability to update the validation state from these hooks is postponed to after 4.1.0). ¶ References: #5888, pull request 5895
- Add support for Botan 2.x and remove support for Botan 1.10. ¶ References: #2250, #5797, pull request 5498
- Print more details of trust anchors. In addition, the *trace* output that mentions if data from authoritative servers gets accepted now also prints the TTL and clarifies the 'place' number previously printed. ¶ References: pull request 5876
- Better support for deleting entries in `NetmaskTree` and `NetmaskGroup`. ¶ References: pull request 5616

Bug Fixes

- Prevent possible downgrade attacks in the recursor. ¶ References: pull request 5889
- Split NODATA / NXDOMAIN NSEC wildcard denial proof of existence. Otherwise there is a very real risk that a NSEC will cover a more specific wildcard and we end up with what looks like a NXDOMAIN proof but is a NODATA one. ¶ References: #5882, pull request 5885
- Fix incomplete validation of cached entries. ¶ References: pull request 5904
- Fix going Insecure on NSEC3 hashes with too many iterations, since we could have gone Bogus on a positive answer synthesized from a wildcard if the corresponding NSEC3 had more iterations that we were willing to accept, while the correct result is Insecure. ¶ References: pull request 5912
- Sort NS addresses by speed and remove old ones. ¶ References: #1066, pull request 5877
- Purge `nsSpeeds` entries even if we get less than 2 new entries. ¶ References: pull request 5896
- Add EDNS to truncated, servfail answers. ¶ References: #5618, pull request 5881
- Use `_exit()` when we *really* want to exit, for example after a fatal error. This stops us dying while we die. A call to `exit()` will trigger destructors, which may paradoxically stop the process from exiting, taking down only one thread, but harming the rest of the process. ¶ References: pull request 5917
- In the recursor secpoll code, we assumed the TXT record would be the first record first record we received. Sometimes it was the RRSIG, leading to a silent error, and no secpoll check. Fixed the assumption, added an error. ¶ References: pull request 5930
- Don't crash when asked to run with zero threads. ¶ References: pull request 5938
- Only accept types not matching the query if we asked for ANY. Even from forward-recurse servers. ¶ References: #5934, pull request 5939
- Allow the use of a 'self-resolving' NS if cached A / AAAA exists. Before this, we could skip a perfectly valid NS for which we had retrieved the A and / or AAAA entries, for example via a glue. ¶ References: #2758, pull request 5937
- Add the config-name argument to the definition of configname. There was a bug where the config-name parameter was not used to change the path of the config file. This meant that some commands via rec_control (e.g. reload-acls) would fail when run against a recursor which had config-name defined. The correct behaviour was present in some, but not all, definitions of configname. (@jake2184) ¶ References: pull request 5961

17.11.21 4.1.0-rc2

Released: 30th of October 2017

The second Release Candidate contains several correctness fixes for DNSSEC, mostly in the area of verifying negative responses.

Improvements

- Don't directly store NSEC3 records in the positive cache. ¶ References: pull request 5834
- Improve logging for the built-in *webserver* and the *Carbon* sender. ¶ References: pull request 5805
- New b.root ipv4 address (Kees Monshouwer). ¶ References: #5663, pull request 5824
- Add *experimental metrics* that track the time spent inside PowerDNS per query. These metrics ignore time spent waiting for the network. ¶ References: pull request 5774
- Add *log-timestamp* setting. This option can be used to disable printing timestamps to stdout, this is useful when using `systemd-journald` or another supervisor that timestamps output by itself. ¶ References: pull request 5842

Bug Fixes

- Check that the NSEC covers an empty non-terminal when looking for NODATA. ¶ References: pull request 5808
- Disable validation for infrastructure queries (e.g. when recursing for a name). Also validate entries from the Negative cache if they were not validated before. ¶ References: #5827, pull request 5835
- Fix DNSSEC validation for denial of wildcards in negative answers and denial of existence proofs in wildcard-expanded positive responses. ¶ References: #5861, pull request 5868
- Fix DNSSEC validation when using `-flto`. ¶ References: pull request 5873
- Lowercase all outgoing qnames when *lowercase-outgoing* is set. ¶ References: pull request 5740
- Create *socket-dir* from the init-script. ¶ References: #5439, pull request 5762
- Fix crashes with uncaught exceptions in MThreads. ¶ References: pull request 5803

17.11.22 4.1.0-rc1

Released: 9th of October 2017

The RC1 release features many fixes to the DNSSEC validation code, reported by different users. Other improvements include: logging, RPZ and the Remote Logger.

While not specifically mentioned in the ChangeLog, also thanks to Winfried Angele for bringing a documentation issue to our attention!

Improvements

- Improve `--quiet=false` output to include DNSSEC and more timing details. ¶ References: pull request 5756
- Add DNSSEC test vectors for RSA, ECDSA, ed25519 and GOST. ¶ References: pull request 5733
- Wrap the webserver's and Resolver::tryGetSOASerial objects into smart pointers (also thanks to Chris Hofstaedtler for reviewing!) ¶ References: pull request 5543
- Add more unit tests for the NetmaskTree and ECS cache index. ¶ References: pull request 5545
- Switch the default webserver's ACL to `127.0.0.1, ::1`. ¶ References: pull request 5588

- Add help text on autodetecting systemd support. (Ruben Kerkhof thanks for reporting!) ¶ References: #5524, pull request 5598
- Add `log-rpz-changes` to log RPZ additions and removals. ¶ References: pull request 5622
- Log the policy type (QName, Client IP, NS IP...) over protobuf. ¶ References: pull request 5621
- Remove unused SortList compare operator for ComboAddress. ¶ References: pull request 5637
- Add support for dumping the in-memory RPZ zones to a file. ¶ References: pull request 5620
- Support for identifying devices by id such as mac address. ¶ References: pull request 5646
- Implement dynamic cache sizing. ¶ References: pull request 5699
- Improve dnsbulktest experience in Travis for more robustness. ¶ References: pull request 5755
- Set `TC=1` if we had to omit part of the AUTHORITY section. ¶ References: pull request 5772
- autoconf: set `--with-libsodium` to `auto`. ¶ References: pull request 5764

Bug Fixes

- Don't fetch the DNSKEY of a zone to validate the DS of the same zone. ¶ References: pull request 5569
- Improve DNSSEC debug logging, ¶ References: pull request 5614
- Add NSEC records on nx-trust cache hits. ¶ References: #5649, pull request 5672
- Handle NSEC wrap-around. ¶ References: #5650, pull request 5671
- Fix erroneous check for section 4.1 of rfc6840. ¶ References: #5648, #5651, pull request 5670
- Handle direct NSEC queries. ¶ References: #5705, pull request 5715
- Detect zone cuts by asking for DS instead of NS. ¶ References: #5681, pull request 5716
- Do not allow direct queries for RRSIG or NSEC3. ¶ References: #5735, pull request 5738
- The target zone being insecure doesn't mean that the denial of the DS is too, if the parent zone is Secure.. ¶ References: pull request 5771
- Add a missing header for PRId64 in the negative cache, required on EL5/EL6. ¶ References: pull request 5530
- Prevent an infinite loop if we need auth and the best match is not. ¶ References: pull request 5549
- Be more careful about the validation of negative answers. ¶ References: pull request 5570
- Fix libatomic detection on ppc64. (Sander Hoentjen) ¶ References: #5456, pull request 5599
- Fix sortlist in the presence of CNAME. (Benoit Perroud thanks for reporting this issue!) ¶ References: #5357, pull request 5615
- Fix cache handling of ECS queries with a source length of 0. ¶ References: pull request 5515
- Handle SNMP alarms so we can reconnect to the master. ¶ References: #5327, pull request 5328
- Fix Recursor 4.1.0 alpha 1 compilation on FreeBSD. (@RvdE) ¶ References: pull request 5662
- Remove pdns.PASS and pdns.TRUNCATE. ¶ References: pull request 5739
- Fix a crash when getting a public GOST key if the private one is not set. ¶ References: pull request 5734
- Don't negcache entries for longer than their RRSIG validity. ¶ References: pull request 5773
- Gracefully handle Socket::accept() returning a null pointer on EAGAIN. ¶ References: pull request 5792

17.11.23 4.1.0-alpha1

Released: 18th of July 2017

This is the first release of the PowerDNS Recursor in the 4.1 release train. This release contains several performance and correctness improvements in the EDNS Client subnet area, as well as better DNSSEC processing.

New Features

- Add support for RPZ wildcarded target names. ¶ References: #5237, pull request 5265
- Add server-side TCP Fast Open support. This adds a new option *tcp-fast-open*. ¶ References: #5128, pull request 5138
- Pass `tcp` to `gettag()` to allow a script to take different actions whether a query came in over TCP or UDP. ¶ References: pull request 4569
- Allow setting the requestor ID field in the `DNSQuestion` from all hooks. ¶ References: pull request 4569
- Implement CNAME wildcards in recursor authoritative component. ¶ References: #2818, pull request 5063
- Allow returning the `DNSQuestion.data` table from `gettag()`. ¶ References: #4981, pull request 4982
- Add *SNMP* support. ¶ References: pull request 4990, pull request 5404
- Allow access to EDNS options from the `gettag()` hook. ¶ References: #5195, pull request 5198
- Pass `tcp` to `gettag()`, allow setting the requestor ID from hooks. ¶ References: pull request 4569
- Allow retrieving stats from Lua via the `getStat()` call. ¶ References: pull request 5293
- Add ECS metrics. ¶ References: pull request 5409
- Add a *cpu-map* directive to set CPU affinity per thread. ¶ References: pull request 5482

Improvements

- Implement "on-the-fly" DNSSEC processing. This places the DNSSEC processing alongside the regular recursion, reducing possible cornercases, adding unit tests and making the code better maintainable. ¶ References: #4254, #4362, #4490, #4994, pull request 5223, pull request 5463, pull request 5486, pull request 5528
- Use ECS when updating the validation state if needed. ¶ References: pull request 5484
- Use the RPZ zone's TTL and add a new *maxTTL* setting. ¶ References: pull request 5057
- RPZ updates are done zone by zone, zones are now shared pointers. ¶ References: #5231, #5236, pull request 5275, pull request 5307
- Split SyncRes::doResolveAt, add const and static whenever possible. Possibly improving performance while making the code easier to maintain. ¶ References: pull request 5106
- Packet cache speedup and cleanup. ¶ References: pull request 5102
- Make Lua mandatory for recursor builds. ¶ References: pull request 5146
- Use one listening socket per thread when reuseport is enabled. ¶ References: pull request 5103, pull request 5487
- Stop (de)serializing `DNSQuestion.data`. ¶ References: pull request 5141
- Refactor the negative cache into a class. ¶ References: pull request 5226
- Only check the netmask for subnet specific cache entries. ¶ References: pull request 5319

- Refactor and split `SyncRes::doResolveAt()`, making it easier to understand. Get rid of `SyncRes::d_nocache`, makes sure we can't get into a root refresh loop. Limit the use of global variables in SyncRes, to make it easier to understand the interaction between components ¶ References: pull request 5236
- Add an ECS index to the cache ¶ References: pull request 5461, pull request 5472
- When dumping the cache, also dump RRSIGs. ¶ References: pull request 5511
- Don't always override *loglevel* to 6. ¶ References: pull request 5485
- Make more specific Netmasks < to less specific ones. ¶ References: pull request 5406, pull request 5530

Bug Fixes

- Fix validation at the exact RRSIG inception or expiration time. ¶ References: pull request 5525
- Fix `remote`/`local` inversion in *`preoutquery()`*. ¶ References: #4969, pull request 4984
- Show a useful error when an invalid *lua-config-file* is configured. ¶ References: #4939, #5075, pull request 5078
- Fix *`DNSQuestion`* members alterations from Lua not being taken into account. ¶ References: pull request 4860
- Ensure locks cannot be copied. ¶ References: pull request 5209
- Only apply *root-nx-trust* if the received SOA is ".". ¶ References: #5246, pull request 5252
- Don't throw an exception when logging to protobuf without a question set. ¶ References: pull request 5312
- Correctly truncate EDNS Client Subnetmasks. ¶ References: pull request 5320
- Clean up auth/recursor code mismatches in the API (Chris Hofstaedtler). ¶ References: #5398, pull request 5466
- Only increase `no-packet-error` on the first read. ¶ References: #5474, pull request 5474

17.12 Changelogs for 4.0.x

This page has all the changelogs for the PowerDNS Recursor 4.0 release train.

Note: 4.0.x and earlier releases are End of Life and no longer supported. See *EOL Statements*.

17.12.1 PowerDNS Recursor 4.0.9

Released 6th of November 2018

This release fixes the following security advisories:

- PowerDNS Security Advisory *2018-04*: Crafted answer can cause a denial of service (CVE-2018-10851)
- PowerDNS Security Advisory *2018-06*: Packet cache pollution via crafted query (CVE-2018-14626)
- PowerDNS Security Advisory *2018-07*: Crafted query for meta-types can cause a denial of service (CVE-2018-14644)

Bug fixes

- #7152: Crafted answer can cause a denial of service (CVE-2018-10851)
- #7152: Packet cache pollution via crafted query (CVE-2018-14626)
- #7152: Crafted query for meta-types can cause a denial of service (CVE-2018-14644)

17.12.2 PowerDNS Recursor 4.0.8

Released 11th of December 2017

This release fixes PowerDNS Security Advisory *2017-08*.

Bug fixes

- #5930: Don't assume TXT record is first record for secpoll
- #6082: Don't add non-IN records to the cache

17.12.3 PowerDNS Recursor 4.0.7

Released 27th of November 2017

This release fixes PowerDNS Security Advisories *2017-03*, *2017-05*, *2017-06* and *2017-07*.

Bug fixes

- #4561: Update rec_control manpage (Winfried Angele)
- #4824: Check in the detected OpenSSL/libcrypto for ECDSA
- #5406: Make more specific Netmasks < to less specific ones
- #5525: Fix validation at the exact RRSIG inception or expiration time
- #5740: Lowercase all outgoing qnames when lowercase-outgoing is set
- #5599: Fix libatomic detection on ppc64
- #5961: Edit configname definition to include the 'config-name' argument (Jake Reynolds)
- #5995: Security Advisories *2017-03*, *2017-05*, *2017-06* and *2017-07*.

Improvements

- #4646: Extract nested exception from Luawrapper
- #4960: Use explicit yes for default-enabled settings (Chris Hofstaedtler)
- #5078: Throw an error when lua-conf-file can't be loaded
- #5261: get-remote-ring's "other" report should only have two items. (Patrick Cloke)
- #5320: PowerDNS sdig does not truncate trailing bits of EDNS Client Subnet mask
- #5488: Only increase *no-packet-error* on the first read
- #5498: Add support for Botan 2.x
- #5511: Add more information to recursor cache dumps
- #5523: Fix typo in two log messages (Ruben Kerkhof)
- #5598: Add help text on autodetecting systemd support
- #5726: Be more resilient with broken auths
- #5739: Remove pdns.PASS and pdns.TRUNCATE
- #5755: Improve dnsbulktest experience in travis for more robustness
- #5762: Create socket-dir from init-script
- #5843: b.root renumbering, effective 2017-10-24

- #5921: Don't retry security polling too often when it fails

17.12.4 PowerDNS Recursor 4.0.6

Released 6th of July 2017

This release features a fix for the ed25519 verifier. This verifier hashed the message before verifying, resulting in unverifiable signatures. Also on the Elliptic Curve front, support was added for ED448 (DNSSEC algorithm 16) by using libdecaf.

Besides that, this release features massive improvements to our edns-client-subnet handling, and some IXFR fixes. Note that this release changes *use-incoming-edns-subnet* to disabled by default.

Bug fixes

- **commit c24288b87:** Use the incoming ECS for cache lookup if *use-incoming-edns-subnet* is set
- **commit b91dc6e92:** when making a netmask from a comboaddress, we neglected to zero the port. This could lead to a proliferation of netmasks.
- **commit 261591b6f:** Don't take the initial ECS source for a scope one if EDNS is off
- **commit 66f894b7a:** also set `d_requestor` without Lua: the ECS logic needs it
- **commit c2086f265:** Fix IXFR skipping the additions part of the last sequence
- **commit a5c9534d0:** Treat requestor's payload size lower than 512 as equal to 512
- **commit 61b1ea2f4:** make URI integers 16 bits, fixes ticket #5443
- **commit 27f9da3c2:** unbreak quoting; fixes ticket #5401

Improvements

- **commit 2325010e6:** with this, EDNS Client Subnet becomes compatible with the packet cache, using the existing variable answer facility.
- **commit 2ec8d8148:** Remove just enough entries from the cache, not one more than asked
- **commit 71df15677:** Move expired cache entries to the front so they are expunged
- **commit d84834c4c:** changed IPv6 addr of b.root-servers.net (Arsen Stasic)
- **commit bcce047bc:** e.root-servers.net has IPv6 now (phonedph1)
- **commit cef8ec7c2:** hello decaf signers (ED25519 and ED448) Testing algorithm 15: 'Decaf ED25519' ->'Decaf ED25519' -> 'Decaf ED25519' Signature & verify ok, signature 68usec, verify 93usec Testing algorithm 16: 'Decaf ED448' ->'Decaf ED448' -> 'Decaf ED448' Signature & verify ok, signature 163usec, verify 252usec (Kees Monshouwer)
- **commit 68490a4b5:** don't use the libdecaf ed25519 signer when libsodium is enabled (Kees Monshouwer)
- **commit 5a88a8ed5:** do not hash the message in the ed25519 signer (Kees Monshouwer)
- **commit 0e7893bf4:** Disable use-incoming-edns-subnet by default

17.12.5 PowerDNS Recursor 4.0.5

Released 13th of June 2017

This release adds ed25519 (algorithm 15) support for DNSSEC and adds the 2017 DNSSEC root key. If you do DNSSEC validation, this upgrade is **mandatory** to continue validating after October 2017.

Bug fixes

- commit af76224: Correctly lowercase the TSIG algorithm name in hash computation, fixes #4942
- commit 86c4ed0: Clear the RPZ NS IP table when clearing the policy, this prevents false positives
- commit 5e660e9: Fix cache-only queries against a forward-zone, fixes #5211
- commit 2875033: Only delegate if NSes are below apex in auth-zones, fixes #4771
- commit e7c183d: Remove hardcoding of port 53 for TCP/IP forwarded zones in recursor, fixes #4799
- commit 5bec36e: Make sure `labelsToAdd` is not empty in `getZoneCuts()`
- commit 0f59e05: Wait until after daemonizing to start the outgoing protobuf thread, prevents hangs when the protobuf server is not available
- commit 233e144: Ensure (re)priming the root never fails
- commit 3642cb3: Don't age the root, fixes a regression from 3.x
- commit 83f9226: Fix exception when sending a protobuf message for an empty question
- commit ffdd813: LuaWrapper: Allow embedded NULs in strings received from Lua
- commit c5ffd90: Fix coredumps on illumos/SmartOS, fixes #4579 (Roman Dayneko)
- commit 651c0e9: StateHolder: Allocate (and copy if needed) before taking the lock
- commit 547d68f: SuffixMatchNode: Fix insertion issue for an existing node
- commit 3ada4e2: Fix negative port detection for IPv6 addresses on 32-bit systems

Additions and Enhancements

- commit 7705e1c: Add support for RPZ wildcarded target names. Fixes #5237
- #5165: Speed up RPZ zone loading and add a `zoneSizeHint` parameter to `rpzFile` and `rpzMaster` for faster reloads
- #4794: Make the RPZ summary consistent (Fixes #4342) and log additions/removals at debug level, not info
- commit 1909556: Add the 2017 root key
- commit abfe671 and commit 7abbb2c: Update Ed25519 algorithm number and mnemonic and hook up to the Recursor (Kees Monshouwer)
- #5355: Add `use-incoming-edns-subnet` option to process and pass along ECS and fix some ECS bugs in the process
- commit dff1a11: Refuse to start with chroot set in a systemd env (Fixes #4848)
- commit 5a38a56: Handle exceptions raised by `closesocket()` to prevent process termination
- #4619: Document missing `top-pub-queries` and `top-pub-servfail-queries` commands for `rec_control` (phonedph1)
- commit 502a850: IPv6 address for g.root-servers.net added (Kevin Otte)
- commit 7a2a645: Log outgoing queries / incoming responses via protobuf

17.12.6 PowerDNS Recursor 4.0.4

Released January 13th 2017

The 4.0.4 version of the PowerDNS Recursor fixes PowerDNS Security Advisories *2016-02* and *2016-04*.

Bug fixes

- commit 658d9e4: Check TSIG signature on IXFR (Security Advisory *2016-04*)
- commit 91acd82: Don't parse spurious RRs in queries when we don't need them (Security Advisory *2016-02*)
- commit 400e28d: Fix incorrect length check in `DNSName` when extracting qtype or qclass
- commit 2168188: rec: Wait until after daemonizing to start the RPZ and protobuf threads
- commit 3beb3b2: On (re-)priming, fetch the root NS records
- commit cfeb109: rec: Fix src/dest inversion in the protobuf message for TCP queries
- commit 46a6666: NSEC3 optout and Bogus insecure forward fixes
- commit bb437d4: On RPZ customPolicy, follow the resulting CNAME
- commit 6b5a8f3: DNSSEC: don't go bogus on zero configured DSs
- commit 1fa6e1b: Don't crash on an empty query ring
- commit bfb7e5d: Set the result to NoError before calling `preresolve`

Additions and Enhancements

- commit 7c3398a: Add `max-recursion-depth` to limit the number of internal recursion
- commit 3d59c6f: Fix building with ECDSA support disabled in libcrypto
- commit 0170a3b: Add requestorId and some comments to the protobuf definition file
- commit d8cd67b: Make the negcache forwarded zones aware
- commit 46ccbd6: Cache records for zones that were delegated to from a forwarded zone
- commit 5aa64e6, commit 5f4242e and commit 0f707cd: DNSSEC: Implement keysearch based on zone-cuts
- commit ddf6fa5: rec: Add support for boost::context >= 1.61
- commit bb6bd6e: Add `getRecursorThreadId()` to Lua, identifying the current thread
- commit d8baf17: Handle CNAMEs at the apex of secure zones to other secure zones

17.12.7 PowerDNS Recursor 4.0.3

Released September 6th 2016

The 4.0.3 version of the PowerDNS Recursor features many improvements to the Policy Engine (RPZ) and the Lua bindings to it. We would like to thank Wim (42wim) for testing and reporting on the RPZ module.

Bug fixes

- #4350: Call `gettag()` for TCP queries
- #4376: Fix the use of an uninitialized filtering policy
- #4381: Parse query-local-address before lua-config-file
- #4383: Fix accessing an empty policyCustom, policyName from Lua
- #4387: ComboAddress: don't allow invalid ports
- #4388: Fix RPZ default policy not being applied over IXFR
- #4391: DNSSEC: Actually follow RFC 7646 §2.1

- #4396: Add boost context ldflags so freebsd builds can find the libs
- #4402: Ignore NS records in a RPZ zone received over IXFR
- #4403: Fix build with OpenSSL 1.1.0 final
- #4404: Don't validate when a Lua hook took the query
- #4425: Fix a protobuf regression (requestor/responder mix-up)

Additions and Enhancements

- #4394: Support Boost 1.61+ fcontext
- #4402: Add Lua binding for DNSRecord::d_place

17.12.8 PowerDNS Recursor 4.0.2

Released August 26th 2016

This release fixes a regression in 4.x where CNAME records for DNSSEC signed domains were not sorted before the final answers, leading to some clients (notably some versions of Chrome) not being able to extract the required answer from the packet. This happened exclusively for DNSSEC signed domains, but the problem happens even for clients not requesting DNSSEC validation.

Further fixes and changes can be found below:

Bug fixes

- #4264: Set `dq.rcode` before calling postresolve
- #4294: Honor PIE flags.
- #4310: Fix build with LibreSSL, for which OPENSSL_VERSION_NUMBER is irrelevant
- #4340: Don't shuffle CNAME records.
- #4354: Fix delegation-only

Additions and enhancements

- #4288: Respect the timeout when connecting to a protobuf server
- #4300: allow newDN to take a DNSName in; document missing methods
- #4301: expose SMN toString to lua
- #4318: Anonymize the protobuf ECS value as well
- #4324: Allow Lua access to the result of the Policy Engine decision, skip RPZ, finish RPZ implementation
- #4349: Remove unused `DNSPacket::d_qlen`
- #4351: RPZ: Use query-local-address(6) by default
- #4357: Move the root DNSSEC data to a header file

17.12.9 PowerDNS Recursor 4.0.1

Released July 29th 2016

This release has several improvements with regards to DNSSEC validation and it improves interoperability with DNSSEC clients that expect an AD-bit on validated data when they query with only the DO-bit set.

Bug fixes

- #4119 Improve DNSSEC record skipping for non dnssec queries (Kees Monshouwer)
- #4162 Don't validate zones from the local auth store, go one level down while validating when there is a CNAME
- #4187:
- Don't go bogus on islands of security
- Check all possible chains for Insecure states
- Don't go Bogus on a CNAME at the apex
- #4215 RPZ: default policy should also override local data RRs
- #4243 Fix a crash when the next name in a chained query is empty and `rec_control current-queries` is invoked

Improvements

- #4056 OpenSSL 1.1.0 support (Chris Hofstaedtler)
- #4133 Add limits to the size of received {A,I}XFR (CVE-2016-6172)
- #4140 Fix warnings with gcc on musl-libc (James Taylor)
- #4160 Also validate on +DO
- #4164 Fail to start when the lua-dns-script does not exist
- #4168 Add more Netmask methods for Lua (Aki Tuomi)
- #4210 Validate DNSSEC for security polling
- #4217 Turn on root-nx-trust by default and log-common-errors=off
- #4207 Allow for multiple trust anchors per zone
- #4242 Fix compilation warning when building without Protobuf

17.12.10 PowerDNS Recursor 4.0.0

Released July 11th 2016

PowerDNS Recursor 4.0.0 is part of the great 4.x "Spring Cleaning" of PowerDNS which lasted through the end of 2015.

As part of the general cleanup, we did the following:

- Moved to C++ 2011, a cleaner more powerful version of C++ that has allowed us to improve the quality of implementation in many places.
- Implemented dedicated infrastructure for dealing with DNS names that is fully "DNS Native" and needs less escaping and unescaping
- Switched to binary storage of DNS records in all places
- Moved ACLs to a dedicated Netmask Tree
- Implemented a version of RCU for configuration changes
- Instrumented our use of the memory allocator, reduced number of malloc calls substantially.
- The Lua hook infrastructure was redone using LuaWrapper; old scripts will no longer work, but new scripts are easier to write under the new interface.

In addition to this cleanup, which has many internal benefits and solves longstanding issues with escaped domain names, 4.0.0 brings the following major new features:

- RPZ aka Response Policy Zone support
- IXFR slaving in the PowerDNS Recursor for RPZ
- DNSSEC processing in Recursor (Authoritative has had this for years)
- DNSSEC validation (without NSEC(3) proof validation)
- EDNS Client Subnet support in PowerDNS Recursor (Authoritative has had this for years)
- Lua asynchronous queries for per-IP/per-domain status
- Caches that can now be wiped per whole zone instead of per name
- Statistics on authoritative server response times (split for IPv4 and IPv6)
- APIs are no longer marked as 'experimental' and had one final URL change
- New metric: tcp-answer-bytes to measure DNS TCP/IP bandwidth, and many other new metrics

Please be aware that beyond the items listed here, there have been heaps of tiny changes. As always, please carefully test a new release before deploying it.

This release features the following fixes compared to rc1:

- #3989 Fix usage of std::distance() in DNSName::isPartOf() (signed/unsigned comparisons)
- #4017 Fix building without Lua. Add `isTcp` to `dq`.
- #4023 Actually log on dnssec=log-fail
- #4028 DNSSEC fixes (NSEC casing, send DO-bit over TCP, DNSSEC trace additions)
- #4052 Don't fail configure on missing fcontext.hpp
- #4096 Don't call `commit()` if we skipped all the records

It has the following improvements:

- #3400 Enable building on OpenIndiana
- #4016 Log protobuf messages for cache hits. Add policy tags in gettag()
- #4040 Allow DNSSEC validation when chrooted
- #4094 Sort included html files for improved reproducibility (Chris Hofstaedtler)

And these additions:

- #3981 Import JavaScript sources for libs shipped with Recursor (Christian Hofstaedtler)
- #4012 add tags support to ProtobufLogger.py
- #4032 Set the existing policy tags in `dq` for `{pre,post}resolve`
- #4077 Add DNSSEC validation statistics
- #4090 Allow reloading the lua-config-file at runtime
- #4097 Allow logging DNSSEC bogus in any mode
- #4125 Add protobuf fields for the query's time in the response

PowerDNS Recursor 4.0.0-rc1

Released June 9th 2016

This first (and hopefully last) Release Candidate contains the finishing touches to the experimental DNSSEC support by adding (Negative) Trust Anchor support and fixing a possible issue with DNSSEC and forwarded domains:

- #3910 Add (Negative) Trust Anchor management
- #3926 Set +CD on forwarded recursive queries

Other changes:

- #3941 Ensure delegations from local auth zones are followed
- #3924 Add a virtual hosting unit-file
- #3929 Set the FDs in the unit file to a sane value

Bug fixes:

- #3961 Fix building on EL6 i386
- #3957 Add error reporting when parsing forward-zones(-recurse) (Aki Tuomi)

PowerDNS Recursor 4.0.0-beta1

Released May 27th 2016

This release fixes a bug in the DNSSEC implementation where a name would we validated as bogus when talking to non-compliant authoritative servers:

- #3875 Disable DNSSEC for domain where the auth responds with FORMERR or NOTIMP

Improvements

- #3866 Increase max FDs in systemd unit file
- #3905 Add a dnssec=process-no-validate option and make it default

Bug fixes

- #3881 Fix the `noEdnsOutQueries` counter
- #3892 support `clock_gettime` for platforms that require -lrt

PowerDNS Recursor 4.0.0-alpha3

Released May 10th 2016

This release features several leaps in the correctness and stability of the DNSSEC implementation.

Notable changes are:

- #3752 Correct handling of query flags in conformance with RFC 6840

Bug fixes

- #3804 Fix a memory leak in DNSSEC validation
- #3785 and #3390 Correctly validate insecure delegations
- #3606 Various DNSSEC fixes, disabling DNSSEC on forward-zones
- #3681 Catch exception with a malformed DNSName in `rec_control wipe-cache`
- #3779, #3768, #3766, #3783 and #3789 DNSName and other hardening improvements

Improvements

- #3801 Add missing Lua rcodes bindings
- #3587 Update L-Root addresses

PowerDNS Recursor 4.0.0-alpha2

Released March 9th 2016

Note that the DNSSEC implementation has several bugs in this release, it is advised to set `dnssec=off` in your recursor.conf.

This release features many low-level performance fixes. Other notable changes since 4.0.0-alpha1 are:

- #3259, #3280 The PowerDNS Recursor now properly uses GNU autoconf and autotools for building and installing
- OpenSSL crypto primitives are now used for DNSSEC validation
- #3313 Implement the logic we need to generate EDNS MAC fields in dnsdist & read them in recursor (blogpost
- #3350 Add lowercase-outgoing feature to Recursor
- #3410 Recuweb is now built-in to the daemon
- #3230 API: drop JSONP, add web security headers (Chris Hofstaedtler)
- #3485 Allow multiple carbon-servers
- #3427, #3479, #3472 MTasker modernization (Andrew Nelless)

Bug fixes

- #3444, #3442 RPZ IXFR fixes
- #3448 Remove edns-subnet-whitelist whitelist pointing to powerdns.com (Christian Hofstaedtler)
- #3293 make asynchronous UDP Lua queries work again in 4.x
- #3365 Apply rcode set in UDPQueryResponse callback (Jan Broers)
- #3244 Fix the forward zones in the recursor
- #3135 Use 56 bits instead of 64 in EDNS Client Subnet option (Winfried Angele)
- #3527 Make the recursor counters atomic

Improvements

- #3435 Add `toStringNoDot` and `chopOff` functions to Lua
- #3437 Add `pdns.now` timeval struct to recursor Lua
- #3352 Cache improvements
- #3502 Make second argument to pdnslog optional (Thiago Farina)
- #3520 Reduce log level of periodic statistics to notice (Jan Broers)

17.12.11 PowerDNS Recursor 4.0.0-alpha1

Released December 24th 2015

17.13 Changelogs for all pre 4.0 releases

Note: Beyond PowerDNS 2.9.20, the Authoritative Server and Recursor are released separately. Hence, this changelog starts at version 3.0.

Note: pre-4.0 releases are End of Life and no longer supported. See *EOL Statements*.

17.13.1 PowerDNS Recursor 3.6.4

Released 9th of June 2015

This is a security release fixing *Security Advisory 2015-01*

Bug fixes:

- commit bccd068: Limit the maximum length of a qname

17.13.2 PowerDNS Recursor 3.7.3

Released 9th of June 2015

Bug fixes:

- commit 92f7b2b: Limit the maximum length of a qname

This is a security release fixing *Security Advisory 2015-01*

Improvements:

- commit 46366a5, commit f318a7d: pdnssec: check for glue and delegations in parent zones (Kees Monshouwer)

17.13.3 PowerDNS Recursor 3.7.2

Released 23rd of April, 2015

Among other bug fixes and improvements (as listed below), this release incorporates a fix for CVE-2015-1868, as detailed in *PowerDNS Security Advisory 2015-01*

Bug fixes:

- commit adb10be commit 3ec3e0f commit dc02ebf Fix handling of forward references in label compressed packets; fixes CVE-2015-1868
- commit a7be3f1: make sure we never call sendmsg with msg_control!=NULL && msg_controllen>0. Fixes ticket #2227
- commit 9d835ed: Improve robustness of root-nx-trust.

Improvements:

- commit 99c595b: Silence warnings that always occur on FreeBSD (Ruben Kerkhof)

17.13.4 PowerDNS Recursor 3.6.3

Released 23rd of April, 2015

The only difference between Recursor 3.6.2 and 3.6.3 is a fix for CVE-2015-1868, as detailed in *PowerDNS Security Advisory 2015-01*

17.13.5 PowerDNS Recursor 3.7.0

Unreleased, please see the 3.7.1 changelog below.

17.13.6 PowerDNS Recursor 3.7.1

Released February 12th, 2015.

This version contains a mix of speedups and improvements, the combined effect of which is vastly improved resilience against traffic spikes and malicious query overloads.

Of further note is the massive community contribution, mostly over Christmas. Especially Ruben Kerkhof, Pieter Lexis, Kees Monshouwer and Aki Tuomi delivered a lot of love. Thanks!

Minor changes:

- Removal of dead code here and there 04dc6d618734fc630122de4c56dff641ebaf0988
- Per-qtype response counters are now 64 bit 297bb6acf7902068693a4aae1443c424d0e8dd52 on 64 bit systems
- Add IPv6 addresses for b and c.root-servers.net hints efc2595423c9a1be6f2d8f4da25445198ceb8b57
- Add IP address to logging about terminated queries 37aa9904d1cc967ba4b5d5e17dbe41485f8cdece
- Improve qtype name logging fab3ed3453e15ae88e29a0e4071b214eb19caad9 (Aki Tuomi)
- Redefine 'BAD_NETS' for dont-query based on newer IANA guidance 12cd44ee0fcde5893f85dccc499bfc35152c5fff (lochiiconnectivity)
- Add documentation links to systemd unit eb154adfdffa5c78624e2ea98e938d7b5787119e (Ruben Kerkhof)

Improvements:

- Upgrade embedded PolarSSL to 1.3.9: d330a2ea1a93d7675ef680311f8aa0306aeefcf1
- yahttp upgrade c290975778942ed1082ca66918695a5bd2d6bac4 c65a57e888ee48eaa948e590c90c51420bffa847 (Aki Tuomi)
- Replace . in hostnames by - for Carbon so as not to confuse Metronome 46541751ed1c3bc051d78217543d5fc76733e212
- Manpages got a lot of love and are now built from Markdown (Pieter Lexis)
- Move to PolarSSL base64 488360551009784ab35c43ee4580e773a2a8a227 (Kees Monshouwer)
- The quiet=no query logging is now more informative 461df9d20c560d240285f772c09b3beb89d46daa
- We can finally bind to 0.0.0.0 and :: and guarantee answers from the correct source b71b60ee73ef3c86f80a2179981eda2e61c4363f
- We use per-packet timestamps to drop ancient traffic in case of overload b71b60ee73ef3c86f80a2179981eda2e61c4363f, non-Linux portability in d63f0d83631c41eff203d30b0b7c475a88f1db59
- Builtin webserver can be queried with the API key in the URL again c89f8cd022c4a9409b95d22ffa3b03e4e98dc400
- Ringbuffers are now available via API c89f8cd022c4a9409b95d22ffa3b03e4e98dc400
- Lua 5.3 compatibility 59c6fc3e3931ca87d484337daee512e716bc4cf4 (Kees Monshouwer)
- No longer leave a stale UNIX domain socket around from rec_control if the recursor was down 524e4f4d81f4ed9eb218715cbc8a59f0b9868234, ticket #2061
- Running with 'quiet=no' would strangely actually prevent debug messages from being logged f48d7b657ec32517f8bfcada3bfe6353ca313314
- Webserver now implements CORS for the API ea89a97e864c43c1cb03f2959ad04c4ebe7580ad, fixing ticket #1984

- Housekeeping thread would sometimes run multiple times simultaneously, which worked, but was odd cc59bce675e62e2b9657b42614ce8be3312cae82

New features:

- New `root-nx-trust` flag makes PowerDNS generalize NXDOMAIN responses from the root-servers 01402d56846a3a61811ebd4e6bc97e53f908e568
- `getregisteredname()` for Lua, which turns 'www.bbc.co.uk' into 'bbc.co.uk' 8cd4851beb78bc6ab320926fb5cb6a09282016b1
- Lua preoutquery filter 3457a2a0ec41d3b3aff7640f30008788e1228a6e
- Lua IP-based filter (ipfilter) before parsing packets 4ea949413c495254acb0bd19335142761c1efc0c
- `iputils` class for Lua, to quickly process IP addresses and netmasks in their native format
- `getregisteredname` function for Lua, to find the registered domain for a given name
- Various new ringbuffers: top-servfail-remotes, top-largeanswer-remotes, top-servfail-queries

Speedups:

- Remove unneeded malloc traffic 93d4a89096e64d53740790f58fadec56f6a0af14 8682c32bc45b6ffa7c0f6da778e1b223ae7f03ce a903b39cfe7364c56324038264d3db50b8cece87
- Our nameserver-loop detection carried around a lot of baggage for complex domain names, plus did not differentiate IPv4 and IPv6 well enough 891fbf888ccac074e3edc38864641ca774f2f03c
- Prioritize new queries over nameserver responses, improving latency under query bursts bf3b0cec366c090af000b066267b6f6bbb3a512a
- Remove escaping in case there was nothing to escape 83b746fd1d94c8742d8bd87a44beb44c154230c7
- Our logging infrastructure had a lot of locking d1449e4d073595e1e1581804f121fc90e37158bf
- Reduce logging level of certain common messages, which locked up synchronously logging systems 854d44e31c76aa650520e6d462dd3a02b5936f7a
- Add limit on total wall-clock time spent on a query 9de3e0340fa066d4c59449e1643a1de8c343f8f2
- Packet cache is now case-insensitive, which increases hitrate 90974597aadaf1096e3fd0dc450be7422ea591a5

Security relevant:

- Check for PIE, RELRO and stack protector during configure 8d0354b189c12e1e14f5309d3b49935c17f9eeb0 (Aki Tuomi)
- Testing for support of PIE etc was improved in b2053c28ccb9609e2ce7bcb6beda83f98a062aa3 and beyond, fixes #2125 (Ruben Kerkhof)
- Max query-per-query limit (max-qperq) is now configurable 173d790ead08f67733010ca4c6fc404a040fe699

Bugs fixed:

- IPv6 outgoing queries had a disproportionate effect on our query load. Fixed in 76f190f2a0877cd79ede2994124c1a58dc69ae49 and beyond.
- rec_control gave incorrect output on a timeout 12997e9d800734da51b808767e1e2477244c30eb
- When using the webserver AND having an error in the Lua script, recursor could crash during startup 62f0ae62984adadab687c23fe1b287c1f219b2cb
- Hugely long version strings would trip up security polling 18b7333828a1275ae5f5574a9c8330290d8557ff (Kees Monshouwer)
- The 'remotes' ringbuffer was sized incorrectly f8f243b01215d6adcb59389f09ef494f1309041f
- Cache sizes had an off-by-one scaling problem, with the wrong number of entries allocated per thread f8f243b01215d6adcb59389f09ef494f1309041f
- Our automatic file descriptor limit raising was attempted *after* setuid, which made it a lot less effective. Found and fixed by Aki Tuomi a6414fdce9b0ec32c340d1f2eea2254f3fedc1c1

- Timestamps used for dropping packets were occasionally wrong 183eb8774e4bc2569f06d5894fec65740f4b70b6 and 4c4765c104bacc146533217bcc843efb244a8086 (RC2) with thanks to Winfried for debugging.
- In RC1, our new DoS protection measures would crash the Recursor if too many root servers were unreachable. 6a6fb05ad81c519b4002ed1db00f3ed9b7bce6b4. Debugging and testing by Fusl.

Various other documentation changes by Chris Hofstaedtler and Ruben Kerkhof. Lots of improvements all over the place by Kees Monshouwer.

17.13.7 PowerDNS Recursor 3.6.2

Note: Version 3.6.2 is a bugfix update to 3.6.1. Released on the 30th of October 2014.

Official download page

A list of changes since 3.6.1 follows.

- commit ab14b4f: expedite servfail generation for ezdns-like failures (fully abort query resolving if we hit more than 50 outqueries). This also prevents the issue documented in *PowerDNS Security Advisory 2014-02* (CVE-2014-8601)
- commit 42025be: PowerDNS now polls the security status of a release at startup and periodically. More detail on this feature, and how to turn it off, can be found in Security polling.
- commit 5027429: We did not transmit the right 'local' socket address to Lua for TCP/IP queries in the recursor. In addition, we would attempt to lookup a filedescriptor that wasn't there in an unlocked map which could conceivably lead to crashes. Closes ticket 1828, thanks Winfried for reporting
- commit 752756c: Sync embedded yahttp copy. API: Replace HTTP Basic auth with static key in custom header
- commit 6fdd40d: add missing `#include <pthread.h>` to rec-channel.hh (this fixes building on OS X).

17.13.8 PowerDNS Recursor 3.6.1

Warning: Version 3.6.1 is a mandatory security upgrade to 3.6.0! Released on the 10th of September 2014.

PowerDNS Recursor 3.6.0 could crash with a specific sequence of packets. For more details, see the advisory. PowerDNS Recursor 3.6.1 was very well tested, and is in full production already, so it should be a safe upgrade.

Downloads

- Official download page

In addition to various fixes related to this potential crash, 3.6.1 fixes a few minor issues and adds a debugging feature:

- We could not encode IPv6 AAAA records that mapped to IPv4 addresses in some cases (:ffff.1.2.3.4). Fixed in commit c90fcbd , closing ticket 1663.
- Improve systemd startup timing with respect to network availability (commit cf86c6a), thanks to Morten Stevens.
- Realtime telemetry can now be enabled at runtime, for example with 'rec_control carbon-server 82.94.213.34 ourname1234'. This ties in to our existing carbon-server and carbon-ourname settings, but now at runtime. This specific invocation will make your stats appear automatically on our public telemetry server.

17.13.9 PowerDNS Recursor version 3.6.0

This is a performance, feature and bugfix update to 3.5/3.5.3. It contains important fixes for slightly broken domain names, which your users expect to work anyhow. It also brings robust resilience against certain classes of attacks.

Downloads

- Official download page
- native RHEL5/6 packages from Kees Monshouwer

Changes between RC1 and release

- commit 30b13ef: do not apply some of our filters to root and gtlds, plus remove some useless {}
- commit cc81d90: fix yahttp copy in dist-recursor for BSD cp
- commit b798618: define __APPLE_USE_RFC_3542 during recursor build on Darwin, fixes ticket 1449
- commit 1d7f863: Merge pull request ticket 1443 from zeha/recursor-nostrip
- commit 5cdeede: remove (non-working) [aaaa-]additional-processing flags from the recursor. Closes ticket 1448
- commit 984d747: Support building recursor on kFreeBSD and Hurd
- commit 79240f1: Allow not stripping of binaries in recursor's make install
- commit e9c2ad3: document pdns.DROP for recursor, add policy-drops metric for it

New features

- commit aadceba: Implement minimum-ttl-override config setting, plus runtime configurability via 'rec_control set-minimum-ttl'.
- Lots of work on the JSON API, which is exposed via Aki Tuomi's 'yahttp'. Massive thanks to Chris Hofstaedtler for delivering this exciting new functionality. Documentation & demo forthcoming, but code to use it is available on GitHub.
- Lua modules can now use 'pdnslog(INFO..'), as described in ticket 1074, implemented in commit 674a305
- Adopt any-to-tcp feature to the recursor. Based on a patch by Winfried Angele. Closes ticket 836, commit 56b4d21 and commit e661a20.
- commit 2c78bd5: implement built-in statistics dumper using the 'carbon' protocol, which is also understood by metronome (our mini-graphite). Use 'carbon-server', 'carbon-ourname' and 'carbon-interval' settings.
- New setting 'udp-truncation-threshold' to configure from how many bytes we should truncate. commit a09a8ce.
- Proper support for CHaos class for CHAOS TXT queries. commit c86e1f2, addition for lua in commit f94c53d, some warnings in commit 438db54 however.
- Added support for Lua scripts to drop queries w/o further processing. commit 0478c54.
- Kevin Holly added qtype statistics to recursor and rec_control (get-qtypelist) (commit 79332bf)
- Add support for include-files in configuration, also reload ACLs and zones defined in them (commit 829849d, commit 242b90e, commit 302df81).
- Paulo Anes contributed server-down-max-fails which helps combat Recursive DNS based amplification attacks. Described in this post. Also comes with new metric 'failed-host-entries' in commit 406f46f.
- commit 21e7976: Implement "followCNAMERecords" feature in the Lua hooks.

Improvements

- commit 06ea901: make pdns-distributes-queries use a hash so related queries get sent to the same thread. Original idea by Winfried Angele. Astoundingly effective, approximately halves CPU usage!
- commit b13e737: –help now writes to stdout instead of stderr. Thanks Winfried Angele.
- To aid in limiting DoS attacks, when truncating a response, we actually truncate all the way so only the question remains. Suggested in ticket 1092, code in commit add935a.
- No longer experimental, the switch 'pdns-distributes-queries' can improve multi-threaded performance on Linux (various cleanup commits).
- Update to embedded PolarSSL, plus remove previous AES implementation and shift to PolarSSL (commit e22d9b4, commit 990ad9a)
- commit 92c0733 moves various Lua magic constants into an enum namespace.
- set group and supplementary groups before chroot (commit 6ee50ce, ticket 1198).
- commit 4e9a20e: raise our socket buffer setting so it no longer generates a warning about lowering it.
- commit 4e9a20e: warn about Linux suboptimal IPv6 settings if we detect them.
- SIGUSR2 turns on a 'trace' of all DNS traffic, a second SIGUSR2 now turns it off again. commit 4f217ce.
- Various fixes for Lua 5.2.
- commit 81859ba: No longer attempt to answer questions coming in from port 0, reply would not reach them anyhow. Thanks to Niels Bakker and 'sid3windr' for insight & debugging. Closes ticket 844.
- commit b1a2d6c: now, I'm not one to get OCD over things, but that log message about stats based on 1801 seconds got to me. 1800 now.

Fixes

- 0c9de4fc: stay away from getaddrinfo unless we really can't help it for ascii ipv6 conversions to binary
- commit 08f3f63: fix average latency calculation, closing ticket 424.
- commit 75ba907: Some of our counters were still 32 bits, now 64.
- commit 2f22827: Fix statistics and stability when running with pdns-distributes-queries.
- commit 6196f90: avoid merging old and new additional data, fixes an issue caused by weird (but probably legal) Akamai behaviour
- commit 3a8a4d6: make sure we don't exceed the number of available filedescriptors for mthreads. Raises performance in case of DoS. See this post for further details.
- commit 7313fe6: implement indexed packet cache wiping for recursor, orders of magnitude faster. Important when reloading all zones, which causes massive cache cleaning.
- rec_control get-all would include 'cache-bytes' and 'packetcache-bytes', which were expensive operations, too expensive for frequent polling. Removed in commit 8e42d27.
- All old workarounds for supporting Windows of the XP era have been removed.
- Fix issues on S390X based systems which have unsigned characters (commit 916a0fd)

17.13.10 PowerDNS Recursor version 3.5.3

Released September 17th, 2013

This is a bugfix and performance update to 3.5.2. It brings serious performance improvements for dual stack users.

Downloads

- Official download page
- native RHEL5/6 packages from Kees Monshouwer

Changes since 3.5.2

- 3.5 replaced our ANY query with A+AAAA for users with IPv6 enabled. Extensive measurements by Darren Gamble showed that this change had a non-trivial performance impact. We now do the ANY query like before, but fall back to the individual A+AAAA queries when necessary. Change in commit 1147a8b.
- The IPv6 address for d.root-servers.net was added in commit 66cf384, thanks Ralf van der Enden.
- We now drop packets with a non-zero opcode (i.e. special packets like DNS UPDATE) earlier on. If the experimental pdns-distributes-queries flag is enabled, this fix avoids a crash. Normal setups were never susceptible to this crash. Code in commit 35bc40d, closes ticket 945.
- TXT handling was somewhat improved in commit 4b57460, closing ticket 795.

17.13.11 PowerDNS Recursor version 3.5.2

Released June 7th, 2013

This is a stability and bugfix update to 3.5.1. It contains important fixes that improve operation for certain domains.

Downloads

- Official download page
- native RHEL5/6 packages from Kees Monshouwer

Changes since 3.5.1

- Responses without the QR bit set now get matched up to an outstanding query, so that resolution can be aborted early instead of waiting for a timeout. Code in commit ee90f02.
- The depth limiter changes in 3.5.1 broke some legal domains with lots of indirection. Improved in commit d393c2d.
- Slightly improved logging to aid debugging. Code in commit 437824d and commit 182005e.

17.13.12 PowerDNS Recursor version 3.5.1

Released May 3rd, 2013

This is a stability and bugfix update to 3.5. It contains important fixes that improve operation for certain domains.

Downloads

- Official download page
- native RHEL5/6 packages from Kees Monshouwer

Changes since 3.5

- We now abort earlier while following endless glue or CNAME chains. Fix in commit 02d1742.
- Some unused code would crash certain gcc versions on ARM. Reported by Morten Stevens, fixed in commit 5b188e8.
- The 3.5 fix for ticket 731 was too strict, causing trouble with at least one domain. Reported by Aki Tuomi, check slightly relaxed in commit 4134690.
- Automake/autoconf now use non-deprecated syntax. Reported by Morten Stevens, change in commit ca17ef2.

17.13.13 PowerDNS Recursor version 3.5

Released April 15th, 2013

This is a stability, security and bugfix update to 3.3/3.3.1. It contains important fixes for slightly broken domain names, which your users expect to work anyhow. **Note**: Because a semi-sanctioned 3.4-pre was distributed for a long time, and people have come to call that 3.4, we are skipping an actual 3.4 release to avoid confusion.

Downloads

- Official download page
- native RHEL5/6 packages from Kees Monshouwer

Changes between RC5 and the final 3.5 release

- Winfried Angele reported that restarting a very busy recursor could lead to crashes. Fixed in r3153, closing ticket 735.

Changes between RC4 and RC5

- Bernd-René Predota of Liberty Global reported that Recursor 3.3 would treat empty non-AA NOERROR responses as authoritative NXDATA responses. This bug turned out to be in 3.5-RC4 too. Fixed in commit 3146, related to ticket 731.

Changes between RC3 (unreleased) and RC4

- Winfried Angele spotted, even before release, that commit 3132 in RC3 broke outgoing IPv6 queries. We are grateful for his attention to detail! Fixed in commit 3141. Changes between RC2 and RC3 (unreleased)
- Use private temp dir when running under systemd, thanks Morten Stevens and Ruben Kerkhof. Change in commit 3105.
- NSD mistakenly compresses labels for RP and other types, violating a MUST in RFC 3597. Recursor does not decompress these labels, violating a SHOULD in RFC3597. We now decompress these labels, and reportedly NSD will stop compressing them. Reported by Jan-Piet Mens, fixed in commit 3109.
- When forwarding to another recursor, we would handle responses to ANY queries incorrectly. Spotted by Jan-Piet Mens, fixed in commit 3116, closes ticket 704.
- Our local-nets definition (used as a default for some settings) now includes the networks from RFC 3927 and RFC 6598. Reported by Maik Zumstrull, fixed in commit 3122.
- The RC1 change to stop using ANY queries to get A+AAAA for name servers in one go had a 5% performance impact. This impact is corrected in commit 3132. Thanks to Winfried Angele for measuring and reporting this. Closes ticket 710.

- New command 'rec_control dump-nsspeeds' will dump our NS speeds (latency) cache. Code in commit 3131.

Changes between RC1 and RC2

- While Recursor 3.3 was not vulnerable to the specific attack noted in 'Ghost Domain Names: Revoked Yet Still Resolvable' (more information at A New DNS Exploitation Technique: Ghost Domain Names), further investigation showed that a variant of the attack could work. This was fixed in commit 3085. This should also close the slightly bogus CVE-2012-1193. Closes ticket 668.
- The auth-can-lower-ttl flag was removed, as it did not have any effect in most situations, and thus did not operate as advertised. We now always comply with the related parts of RFC 2181. Change in commit 3092, closing ticket 88.

New features

- The local zone server now understands wildcards, code in commit 2062.
- The Lua postresolve and nodata hooks, that had been distributed as a '3.3-hooks' snapshot earlier, have been merged. Code in commit 2309.
- A new feature, rec_control trace-regex allows the tracing of lookups for specific names. Code in commit 3044, commit 3073.
- A new setting, export-etc-hosts-search-suffix, adds a configurable suffix to names imported from /etc/hosts. Code in commit 2544, commit 2545.

Improvements

- We now throttle queries that don't work less aggressively, code in commit 1766.
- Various improvements in tolerance against broken auths, code in commit 1996, commit 2188, commit 3074 (thanks Winfried).
- Additional processing is now optional, and disabled by default. Presumably this yields a performance improvement. Change in commit 2542.
- rec_control reload-lua-script now reports errors. Code in commit 2627, closing ticket 278.
- rec_control help now lists commands. Code in commit 2628.
- rec_control wipe-cache now also wipes the recursor's packet cache. Code in commit 2880 from ticket 333.
- Morten Stevens contributed a systemd file. Import in commit 2966, now part of the recursor tarball.
- commit 2990 updates the address of D.root-servers.net.
- Winfried Angele implemented and documented the ipv6-questions metric. Merge in commit 3034, closing ticket 619.
- We no longer use ANY to get A+AAAA for nameservers, because some auth operators have decided to break ANY lookups. As a bonus, we now track v4 and v6 latency separately. Change in commit 3064.

Bugs fixed

- Some unaligned memory access was corrected, code in commit 2060, commit 2122, commit 2123, which would cause problems on UltraSPARC.
- Garbage encountered during reload-acls could cause crashes. Fixed in commit 2323, closing ticket 330.
- The recursor would lose its root hints in a very rare situation. Corrected in commit 2380.

- We did not always drop supplemental groups while dropping privileges. Reported by David Black of Atlassian, fixed in commit 2524.
- Cache aging would sometimes get confused when we had a mix of expired and non-expired records in cache. Spotted and fixed by Winfried Angele in commit 3068, closing ticket 438.
- rec_control reload-acl no longer ignores arguments. Fix in commit 3037, closing ticket 490.
- Since we re-parse our commandline in rec_control we've been doubling the commands on the commandline, causing weird output. Reported by Winfried Angele. Fixed in commit 2992, closing ticket 618. This issue was not present in any officially released versions.
- commit 2879 drops some spurious stderr logging from Lua scripts, and makes sure 'place' is always valid.
- We would sometimes refuse to resolve domains with just one nameserver living at the apex. Fixed in commit 2817.
- We would sometimes stick RRs in the wrong parts of response packets. Fixed in commit 2625.
- The ACL parser was too liberal, sometimes causing recursors to be very open. Fixed in commit 2629, closing ticket 331.
- rec_control now honours socket-dir from recursor.conf. Fixed in commit 2630.
- When traversing CNAME chains, sometimes we would end up with multiple SOAs in the result. Fixed in commit 2633.

17.13.14 Recursor version 3.3.1

Warning:Unreleased

Version 3.3.1 contains a small number of important fixes, adds some memory usage statistics, but no new features.

- Discovered by John J and Robin J, the PowerDNS Recursor did not process packets that were truncated in mid-record, and also did not act on the 'truncated' (TC) flag in that case. This broke a very small number of domains, most of them served by very old versions of the PowerDNS Authoritative Server. Fix in commit 1740.
- PowerDNS emitted a harmless, but irritating, error message on receiving certain very short packets. Discovered by Winfried A and John J, fix in commit 1729.
- PowerDNS could crash on startup if configured to provide service on malformed IPv6 addresses on FreeBSD, or in case when the FreeBSD kernel was compiled without any form of IPv6 support. Debugged by Bryan Seitz, fix in commit 1727.
- Add max-mthread-stack metric to debug rare crashes. Could be used to save memory on constrained systems. Implemented in commit 1745.
- Add cache-bytes and packetcache-bytes metrics to measure our 'pre-malloc' memory utilization. Implemented in commit 1750.

17.13.15 Recursor version 3.3

Released on the 22nd of September 2010.

Warning: Version 3.3 fixes a number of small but persistent issues, rounds off our IPv6 %link-level support and adds an important feature for many users of the Lua scripts.

In addition, scalability on Solaris 10 is improved.

Bug fixes

- 'dist-recursor' script was not compatible with pure POSIX /bin/sh, discovered by Simon Kirby. Fix in commit 1545.

- Simon Bedford, Brad Dameron and Laurient Papier discovered relatively high TCP/IP loads could cause TCP/IP service to shut down over time. Addressed in commits 1546, 1640, 1652, 1685, 1698. Additional information provided by Zwane Mwaikambo, Nicholas Miell and Jeff Roberson. Testing by Chris Hofstaedtler and Michael Renner.
- The PowerDNS Recursor could not read the 'root zone' (this is something else than the root hints) because of an unquoted TXT record. This has now been addressed, allowing operators to hardcode the root zone. This can improve security if the root zone used is kept up to date. Change in commit 1547.
- A return of an old bug, when a domain gets new nameservers, but the old nameservers continue to contain a copy of the domain, PowerDNS could get 'stuck' with the old servers. Fixed in commit 1548.
- Discovered & reported by Alexander Gall of SWITCH, the Recursor used to try to resolve 'AXFR' records over UDP. Fix in commit 1619.
- The Recursor embedded authoritative server messed up parsing a record like '@ IN MX 15 @'. Spotted by Aki Tuomi, fix in commit 1621.
- The Recursor embedded authoritative server messed up parsing really really long lines. Spotted by Marco Davids, fix in commit 1624, commit 1625.
- Packet cache was not DNS class correct. Spotted by "Robin", fix in commit 1688.
- The packet cache would cache some NXDOMAINs for too long. Solving this bug exposed an underlying oddity where the initial NXDOMAIN response had an overly long (untruncated) TTL, whereas all the next ones would be ok. Solved in commit 1679, closing ticket 281. Especially important for RBL operators. Fixed after some nagging by Alex Broens (thanks).

Improvements

- The priming of the root now uses more IPv6 addresses. Change in commit 1550, closes ticket 287. Also, the IPv6 address of I.ROOT-SERVERS.NET was added in commit 1650.
- The `rec_control dump-cache` command now also dumps the 'negative query' cache. Code in commit 1713.
- PowerDNS Recursor can now bind to fe80 IPv6 space with '%eth0' link selection. Suggested by Darren Gamble, implemented with help from Niels Bakker. Change in commit 1620.
- Solaris on x86 has a long standing bug in port_getn(), which we now work around. Spotted by 'Dirk' and 'AS'. Solution suggested by the Apache runtime library, update in commit 1622.
- New runtime statistic: 'tcp-clients' which lists the number of currently active TCP/IP clients. Code in commit 1623.
- Deal better with UltraDNS style CNAME redirects containing SOA records. Spotted by Andy Fletcher from UKDedicated in ticket 303, fix in commit 1628.
- The packet cache, which has 'ready to use' packets containing answers, now artificially ages the ready to use packets. Code in commit 1630.
- Lua scripts can now indicate that certain queries will have 'variable' answers, which means that the packet cache will not touch these answers. This is great for overriding some domains for some users, but not all of them. Use setvariable() in Lua to indicate such domains. Code in commit 1636.
- Add query statistic called 'dont-outqueries', plus add IPv6 address :: and IPv4 address 0.0.0.0 to the default "dont-query" set, preventing the Recursor from talking to itself. Code in commit 1637.
- Work around a gcc 4.1 bug, still in wide use on common platforms. Code in commit 1653.
- Add 'ARCHFLAGS' to PowerDNS Recursor Makefile, easing 64 bit compilation on mainly 32 bit platforms (and vice versa).
- Under rare circumstances, querying the Recursor for statistics under very high load could lead to a crash (although this has never been observed). Bad code removed & good code unified in commit 1675.
- Spotted by Jeff Sipek, the rec_control manpage did not list the new get-all command. commit 1677.

- On some platforms, it may be better to have PowerDNS itself distribute queries over threads (instead of leaving it up to the kernel). This is an experimental feature and can be enabled with the 'pdns-distributes-queries' setting. Code in commit 1678 and beyond. Speeds up Solaris measurably.
- Cache cleaning code was cleaned up, unified and expanded to cover the 'negative cache', which used to be cleaned rather bluntly. Code in commit 1702, further tweaks in commit 1712, spotted by Darren Gamble, Imre Gergely and Christian Kovacic.

Changes between RC1, RC2 and RC3.

- RC2: Fixed linking on RHEL5/CentOS5, which both ship with a gcc compiler that claims to support atomic operations, but doesn't. Code in commit 1714. Spotted by 'Bas' and Imre Gergely.
- RC2: Negative query cache was configured to grow too large, and was not cleaned efficiently. Code in commit 1712, spotted by Imre Gergely.
- RC3: Root failed to be renewed automatically, relied on fallback to make this happen. Code in commit 1716, spotted by Detlef Peeters.

17.13.16 Recursor version 3.2

Released on the 7th of March 2010.

Warning: Lua scripts from version 3.1.7.* are fully compatible with version 3.2. However, scripts written for development snapshot releases, are NOT. Please see Scripting for details!

The 3.2 release is the first major release of the PowerDNS Recursor in a long time. Partly this is because 3.1.7.* functioned very well, and delivered satisfying performance, partly this is because in order to really move forward, some heavy lifting had to be done.

As always, we are grateful for the large PowerDNS community that is actively involved in improving the quality of our software, be it by submitting patches, by testing development versions of our software or helping debug interesting issues. We specifically want to thank Stefan Schmidt and Florian Weimer, who both over the years have helped tremendously in keeping PowerDNS fast, stable and secure.

This version of the PowerDNS Recursor contains a rather novel form of lock-free multithreading, a situation that comes close to the old '–fork' trick, but allows the Recursor to fully utilize multiple CPUs, while delivering unified statistics and operational control.

In effect, this delivers the best of both worlds: near linear scaling, with almost no administrative overhead.

Compared to 'regular multithreading', whereby threads cooperate more closely, more memory is used, since each thread maintains its own DNS cache. However, given the economics, and the relatively limited total amount of memory needed for high performance, this price is well worth it.

In practical numbers, over 40,000 queries/second sustained performance has now been measured by a third party, with a 100.0% packet response rate. This means that the needs of around 400,000 residential connections can now be met by a single commodity server.

In addition to the above, the PowerDNS Recursor is now providing resolver service for many more Internet users than ever before. This has brought with it 24/7 Service Level Agreements, and 24/7 operational monitoring by networking personnel at some of the largest telecommunications companies in the world.

In order to facilitate such operation, more statistics are now provided that allow the visual verification of proper PowerDNS Recursor operation. As an example of this there are now graphs that plot how many queries were dropped by the operating system because of a CPU overload, plus statistics that can be monitored to determine if the PowerDNS deployment is under a spoofing attack. All in all, this is a large and important PowerDNS Release, paving the way for further innovation.

Note: This release removes support for the 'fork' multi-processor option. In addition, the default is now to spawn two threads. This has been done in such a way that total memory usage will remain identical, so each thread will use half of the allocated maximum number of cache entries.

Changes between RC2 and -release

- 'Make install' when an existing configuration file contained a 'fork' statement has been fixed. Spotted by Darren Gamble, code in commit 1534.
- Reloading a nonexistent allow-from-file caused the control thread to stop working. Spotted by Imre Gergely, code in commit 1532.
- Parser got confused by reading en empty line in auth-forward-zones. Spotted by Imre Gergely, code in commit 1533.
- David Gavarret discovered undocumented and not-working settings to set the owner, group and access modes of the control socket. Code by Aki Tuomi and documentation in commit 1535. Fixup in commit 1536 for FreeBSD as found by Ralf van der Enden.
- Tiny improvement possibly solving an issue on Solaris 10's completion port event multiplexer (commit 1537).

Changes between RC1 and RC2

- Compilation on Solaris 10 has been fixed (various patchlevels had different issues), code in commit 1522.
- Compatibility with CentOS4/RHEL4 has been restored, the gcc and glibc versions shipped with this distribution contain a Thread Local Storage bug which we now work around. Thanks to Darren Gamble and Imre Gergely for debugging this issue, code in commit 1527.
- A failed setuid operation, because of misconfiguration, would result in a crash instead of an error message. Fixed in commit 1523.
- Imre Gergely discovered that PowerDNS was doing spurious root repriming when invalidating nssets. Fixed in commit 1531.
- Imre Gergely discovered our rrd graphs had not been changed for the new multithreaded world, and did not allow scaling beyond 200% cpu use. In addition, CPU usage graphs did not add up correctly. Implemented in commit 1524.
- Andreas Jakum discovered the description of 'max-packetcache-entries' and 'forward-zones-recurse' was wrong in the output of '–help' and '–config'. In addition, some stray backup files made it into the RC1 release. Addressed in commit 1529. Full release notes follow, including some overlap with the incremental release notes above. Improvements
- Multithreading, allowing near linear scaling to multiple CPUs or cores. Configured using 'threads=' (many commits). This also deprecates the '–fork' option.
- Added ability to read a configuration item of a running PowerDNS Recursor using 'rec_control get-parameter' (commit 1243), suggested by Wouter de Jong.
- Added ability to read all statistics in one go of a running PowerDNS Recursor using 'rec_control get-all' (commit 1496), suggested by Michael Renner.
- Speedups in packet generation (Commits 1258, 1259, 1262)
- TCP deferred accept() filter is turned on again for slight DoS protection. Code in commit 1414.
- PowerDNS Recursor can now do TCP/IP queries to remote IPv6 addresses (commit 1412).
- Solaris 9 '/dev/poll' support added, Solaris 8 now deprecated. Changes in commit 1421, commit 1422, commit 1424, commit 1413.
- Lua functions can now also see the address _to_ which a question was sent, using getlocaladdress(). Implemented in commit 1309 and commit 1315.
- Maximum cache sizes now default to a sensible value. Suggested by Roel van der Made, implemented in commit 1354.
- Domains can now be forwarded to IPv6 addresses too, using either ::1 syntax or [::1]:25. Thanks to Wijnand Modderman for discovering this issue, fixed in commit 1349.

- Lua scripts can now load libraries at runtime, for example to calculate md5 hashes. Code by Winfried Angele in commit 1405.
- Periodic statistics output now includes average queries per second, as well as packet cache numbers (commit 1493).
- New metrics are available for graphing, plus added to the default graphs (commit 1495, commit 1498, commit 1503)
- Fix errors/crashes on more recent versions of Solaris 10, where the ports functions could return ENOENT under some circumstances. Reported and debugged by Jan Gyselinck, fixed in commit 1372.

New features

- Add pdnslog() function for Lua scripts, so errors or other messages can be logged properly.
- New settings to set the owner, group and access modes of the control socket (socket-owner, socket-group, socket-mode). Code by Aki Tuomi and documentation in commit 1535. Fixup in commit 1536 for FreeBSD as found by Ralf van der Enden.
- rec_control now accepts a –timeout parameter, which can be useful when reloading huge Lua scripts. Implemented in commit 1366.
- Domains can now be forwarded with the 'recursion-desired' bit on or off, using either **forward-zones-recurse** or by prefixing the name of a zone with a '+' in **forward-zones-file**. Feature suggested by Darren Gamble, implemented in commit 1451.
- Access control lists can now be reloaded at runtime (implemented in commit 1457).
- PowerDNS Recursor can now use a pool of query-local-addresses to further increase resilience against spoofing. Suggested by Ad Spelt, implemented in commit 1426.
- PowerDNS Recursor now also has a packet cache, greatly speeding up operations. Implemented in commit 1426, commit 1433 and further.
- Cache can be limited in how long it maximally stores records, for BIND compatibility (TTL limiting), by setting **max-cache-ttl**.Idea by Winfried Angele, implemented in commit 1438.
- Cache cleaning turned out to be scanning more of the cache than necessary for cache maintenance. In addition, far more frequent but smaller cache cleanups improve responsiveness. Thanks to Winfried Angele for discovering this issue. (commits 1501, 1507)
- Performance graphs enhanced with separate CPU load and cache effectiveness plots, plus display of various overload situations (commits 1503)

Compiler/Operating system/Library updates

- PowerDNS Recursor can now compile against newer versions of Boost (verified up to and including 1.42.0). Reported & fixed by Darix in commit 1274. Further fixes in commit 1275, commit 1276, commit 1277, commit 1283.
- Fix compatibility with newer versions of GCC (closes ticket ticket 227, spotted by Ruben Kerkhof, code in commit 1345, more fixes in commit 1394, 1416, 1440).
- Rrdtool update graph is now compatible with FreeBSD out of the box. Thanks to Bryan Seitz (commit 1517).
- Fix up Makefile for older versions of Make (commit 1229).
- Solaris compilation improvements (out of the box, no handwork needed).
- Solaris 9 MTasker compilation fixes, as suggested by John Levon. Changes in commit 1431.

Bug fixes

- Under rare circumstances, the recursor could crash on 64 bit Linux systems running glibc 2.7, as found in Debian Lenny. These circumstances became a lot less rare for the 3.2 release. Discovered by Andreas Jakum and debugged by #powerdns, fix in commit 1519.
- Imre Gergely discovered that PowerDNS was doing spurious root repriming when invalidating nssets. Fixed in commit 1531.
- Configuration parser is now resistant against trailing tabs and other whitespace (commit 1242)
- Fix typo in a Lua error message. Close ticket 210, as reported by Stefan Schmidt (commit 1319).
- Profiled-build instructions were broken, discovered & fixes suggested by Stefan Schmidt. ticket 239, fix in commit 1462.
- Fix up duplicate SOA from a remote authoritative server from showing up in our output (commit 1475).
- All security fixes from 3.1.7.2 are included.
- Under highly exceptional circumstances on FreeBSD the PowerDNS Recursor could crash because of a TCP/IP error. Reported and fixed by Andrei Poelov in ticket 192, fixed in commit 1280.
- PowerDNS Recursor can be a root-server again. Error spotted by the ever vigilant Darren Gamble (ticket 229), fix in commit 1458.
- Rare TCP/IP errors no longer lead to PowerDNS Recursor logging errors or becoming confused. Debugged by Josh Berry of Plusnet PLC. Code in commit 1457.
- Do not hammer parent servers in case child zones are misconfigured, requery at most once every 10 seconds. Reported & investigated by Stefan Schmidt and Andreas Jakum, fixed in commit 1265.
- Properly process answers from remote authoritative servers that send error answers without including the original question (commit 1329, commit 1327).
- No longer spontaneously turn on 'export-etc-hosts' after reloading zones. Discovered by Paul Cairney, reported in ticket 225, addressed in commit 1348.
- Very abrupt server failure of large numbers of high-volume authoritative servers could trigger an out of memory situation. Addressed in commit 1505.
- Make timeouts for queries to remote authoritative servers configurable with millisecond granularity. In addition, the old code turned out to consider the timeout expired when the integral number of seconds since 1970 increased by 1 - which *on average* is after 500ms. This might have caused spurious timeouts! New default timeout is 1500ms. See **network-timeout** setting for more details. Code in commit 1402.

17.13.17 Recursor version 3.1.7.2

Released on the 6th of January 2010.

This release consist of a number of vital security updates. These updates address issues that can in all likelihood lead to a full system compromise. In addition, it is possible for third parties to pollute your cache with dangerous data, exposing your users to possible harm.

This version has been well tested, and at the time of this release is already powering millions of internet connections, and should therefore be a risk-free upgrade from 3.1.7.1 or any earlier version of the PowerDNS Recursor.

All known versions of the PowerDNS Recursor are impacted to a greater or lesser extent, so an immediate update is advised.

These vulnerabilities were discovered by a third party that can't yet be named, but who we thank for their contribution to a more secure PowerDNS Recursor.

For more information, see *PowerDNS Security Advisory 2010-01* and *PowerDNS Security Advisory 2010-02*.

17.13.18 Recursor version 3.1.7.1

Released on the 2nd of August 2009.

This release consists entirely of fixes for tiny bugs that have been reported over the past year. In addition, compatibility has been restored with the latest versions of the gcc compiler and the 'boost' libraries.

No features have been added, but some debugging code that very slightly impacted performance (and polluted the console when operating in the foreground) has been removed.

FreeBSD users may want to upgrade because of a very remote chance of 3.1.7 and previous crashing once every few years. For other operators not currently experiencing problems, there is no reason to upgrade.

- Improved error messages when parsing zones for authoritative serving (commit 1235).
- Better resilience against whitespace in configuration (changesets 1237, 1240, 1242)
- Slight performance increase (commit 1378)
- Fix rare case where timeouts were not being reported to the right query-thread (commit 1260)
- Fix compilation against newer versions of the Boost C++ libraries (commit 1381)
- Close very rare issue with TCP/IP close reporting ECONNRESET on FreeBSD. Reported by Andrei Poelov in ticket 192.
- Silence debugging output (commit 1286).
- Fix compilation against newer versions of gcc (commit 1384)
- No longer set export-etc-hosts to 'on' on reload-zones. Discovered by Paul Cairney, closes ticket 225.
- Sane default for the maximum cache size in the Recursor, suggested by Roel van der Made (commit 1354).
- No longer exit because of the changed behaviour of the Solaris 'completion ports' in more recent versions of Solaris. Fix in commit 1372, reported by Jan Gyselinck.

17.13.19 Recursor version 3.1.7

Released the 25th of June 2008.

This version contains powerful scripting abilities, allowing operators to modify DNS responses in many interesting ways. Among other things, these abilities can be used to filter out malware domains, to perform load balancing, to comply with legal and other requirements and finally, to implement 'NXDOMAIN' redirection.

It is hoped that the addition of Lua scripting will enable responsible DNS modification for those that need it.

For more details about the Lua scripting, which can be modified, loaded and unloaded at runtime, see Scripting. Many thanks are due to the #lua irc channel, for excellent near-realtime Lua support. In addition, a number of PowerDNS users have been enthusiastically testing prereleases of the scripting support, and have found and solved many issues.

In addition, 3.1.7 fixes a number of bugs

- In 3.1.5 and 3.1.6, an authoritative server could continue to renew its authority, even though a domain had been delegated to other servers in the meantime.

  In the rare cases where this happened, and the old servers were not shut down, the observed effect is that users were fed outdated data. Bug spotted and analysed by Darren Gamble, fix in commit 1182 and commit 1183.
- Thanks to long time PowerDNS contributor Stefan Arentz, for the first time, Mac OS X 10.5 users can compile and run the PowerDNS Recursor! Patch in commit 1185.
- Sten Spans spotted that for outgoing TCP/IP queries, the **query-local-address** setting was not honored. Fixed in commit 1190.
- **rec_control wipe-cache** now also wipes domains from the negative cache, hurrying up the expiry of negatively cached records. Suggested by Simon Kirby, implemented in commit 1204.

- When a forwarder server is configured for a domain, using the **forward-zones** setting, this server IP address was filtered using the **dont-query** setting, which is generally not what is desired: the server to which queries are forwarded will often live in private IP space, and the operator should be trusted to know what he is doing. Reported and argued by Simon Kirby, fix in commit 1211.
- Marcus Rueckert of OpenSUSE reported that very recent gcc versions emitted a (correct) warning on an overly complicated line in syncres.cc, fixed in commit 1189.
- Stefan Schmidt discovered that the netmask matching code, used by the new Lua scripts, but also by all other parts of PowerDNS, had problems with explicit '/32' matches. Fixed in commit 1205.

17.13.20 Recursor version 3.1.6

Released on the 1st of May 2008.

This version fixes two important problems, each on its own important enough to justify a quick upgrade.

- Version 3.1.5 had problems resolving several slightly misconfigured domains, including for a time 'juniper.net'. Nameserver timeouts were not being processed correctly, leading PowerDNS to not update the internal clock, which in turn meant that any queries immediately following an error would time out as well. Because of retries, this would usually not be a problem except on very busy servers, for domains with different nameservers at different levels of the DNS-hierarchy, like 'juniper.net'.

  This issue was fixed rapidly because of the help of XS4ALL (Eric Veldhuyzen, Kai Storbeck), Brad Dameron and Kees Monshouwer. Fix in commit 1178.
- The new high-quality random generator was not used for all random numbers, especially in source port selection. This means that 3.1.5 is still a lot more secure than 3.1.4 was, and its algorithms more secure than most other nameservers, but it also means 3.1.5 is not as secure as it could be. A quick upgrade is recommended. Discovered by Thomas Biege of Novell (SUSE), fixed in commit 1179.

17.13.21 Recursor version 3.1.5

Released on the 31st of March 2008.

Much like 3.1.4, this release does not add a lot of major features. Instead, performance has been improved significantly (estimated at around 20%), and many rare and not so rare issues were addressed. Multi-part TXT records now work as expected - the only significant functional bug found in 15 months. One of the oldest feature requests was fulfilled: version 3.1.5 can finally forward queries for designated domains to multiple servers, on differing port numbers if needed. Previously only one forwarder address was supported. This lack held back a number of migrations to PowerDNS.

We would like to thank Amit Klein of Trusteer for bringing a serious vulnerability to our attention which would enable a smart attacker to 'spoof' previous versions of the PowerDNS Recursor into accepting possibly malicious data.

Details can be found on this Trusteer page.

It is recommended that all users of the PowerDNS Recursor upgrade to 3.1.5 as soon as practicable, while we simultaneously note that busy servers are less susceptible to the attack, but not immune.

The PowerDNS Security Advisory can be found in *PowerDNS Security Advisory 2008-01*.

This version can properly benefit from all IPv4 and IPv6 addresses in use at the root-servers as of early February 2008. In order to implement this, changes were made to how the Recursor deals internally with A and AAAA queries for nameservers, see below for more details.

Additionally, newer releases of the G++ compiler required some fixes (see ticket 173).

This release was made possible by the help of Wichert Akkerman, Winfried Angele, Arnoud Bakker (Fox-IT), Niels Bakker (no relation!), Leo Baltus (Nederlandse Publieke Omroep), Marco Davids (SIDN), David Gavarret (Neuf Cegetel), Peter Gervai, Marcus Goller (UPC), Matti Hiljanen (Saunalahti/Elisa), Ruben Kerkhof, Alex Kiernan, Amit Klein (Trusteer), Kenneth Marshall (Rice University), Thomas Rietz, Marcus Rueckert (OpenSUSE),

Augie Schwer (Sonix), Sten Spans (Bit), Stefan Schmidt (Freenet), Kai Storbeck (xs4all), Alex Trull, Andrew Turnbull (No Wires) and Aaron Thompson, and many more who filed bugs anonymously, or who we forgot to mention.

Security related issues

- Amit Klein has informed us that System random generator output can be predicted based on its past behaviour, allowing a smart attacker to 'spoof' our nameserver. Full details in *PowerDNS Security Advisory 2008-01*.
- The Recursor will by default no longer query private-space nameservers. This closes a slight security risk and simultaneously improves performance and stability. For more information, see **dont-query** in pdns_recursor settings. Implemented in commit 923.
- Applied fix for ticket 110 ('PowerDNS should change directory to '/' in chroot), implemented in commit 944.

Performance

- The DNS packet writing and parsing infrastructure performance was improved in several ways, see commits 925, 926, 928, 931, 1021, 1050.
- Remove multithreading overhead from the Recursor (commit 999).

Bug fixes

- Built-in authoritative server now properly derives the TTL from the SOA record if not specified. Implemented in commit 1165. Additionally, even when TTL was specified for the built-in authoritative server, it was ignored. Reported by Stefan Schmidt, closing ticket 147.
- Empty TXT record components can now be served. Implemented in commit 1166, closing ticket 178. Spotted by Matti Hiljanen.
- The Recursor would not properly override old data with new, sometimes serving old and new data concurrently. Fixed in commit 1137.
- SOA records with embedded carriage-return characters are now parsed correctly. Implemented in commit 1167, closing ticket 162.
- Some routing conditions could cause UDP connected sockets to generate an error which PowerDNS did not deal with properly, leading to a leaked file descriptor. As these run out over time, the recursor could crash. This would also happen for IPv6 queries on a host with no IPv6 connectivity. Thanks to Kai of xs4all and Wichert Akkerman for reporting this issue. Fix in commit 1133.
- Empty unknown record types can now be stored without generating a scary error (commit 1129)
- Applied fix for ticket 111, ticket 112 and ticket 153 - large (multipart) TXT records are now retrieved and served properly. Fix in commit 996.
- Solaris compilation instructions in Recursor documentation were wrong, leading to an instant crash on startup. Luckily nobody reads the documentation, except for Marcus Goller who found the error. Fixed in commit 1124.
- On Solaris, finally fix the issue where queries get distributed strangely over CPUs, or not get distributed at all. Much debugging and analysing performed by Alex Kiernan, who also supplied fixes. Implemented in commit 1091, commit 1093.
- Various fixes for modern G++ versions, most spotted by Marcus Rueckert (commits 964, 965, 1028, 1052), and Ruben Kerkhof (commit 1136, closing ticket 175).
- Recursor would not properly clean up pidfile and control socket, closing ticket 120, code in commit 988, commit 1098 (part of fix by Matti Hiljanen, spotted by Leo Baltus)

- Recursor can now serve multi-line records from its limited authoritative server (commit 1014).
- When parsing zones, the 'm' time specification stands for minutes, not months! Closing Debian bug 406462 (commit 1026)
- Authoritative zone parser did not support '@' in the content of records. Spotted by Marco Davids, fixed in commit 1030.
- Authoritative zone parser could be confused by trailing TABs on record lines (commit 1062).
- EINTR error code could block entire server if received at the wrong time. Spotted by Arnoud Bakker, fix in commit 1059.
- Fix crash on NetBSD on Alpha CPUs, might improve startup behaviour on empty caches on other architectures as well (commit 1061).
- Outbound TCP queries were being performed sub-optimally because of an interaction with the 'MPlexer'. Fixes in commit 1115, commit 1116.

New features

- Implemented **rec_control** command **get uptime**, as suggested by Niels Bakker (commit 935). Added to default rrdtool scripts in commit 940.
- The Recursor Authoritative component, meant for having the Recursor serve some zones authoritatively, now supports $INCLUDE and $GENERATE. Implemented in commit 951 and commit 952, commit 967 (discovered by Thomas Rietz),
- Implemented **forward-zones-file** option in order to support larger amounts of zones which should be forwarded to another nameserver (commit 963).
- Both **forward-zones** and **forward-zones-file** can now specify multiple forwarders per domain, implemented in commit 1168, closing ticket 81. Additionally, both these settings can also specify non-standard port numbers, as suggested in ticket ticket 122. Patch authored by Aaron Thompson, with additional work by Augie Schwer.
- Sten Spans contributed **allow-from-file**, implemented in commit 1150. This feature allows the Recursor to read access rules from a (large) file.

General improvements

- Ruben Kerkhof fixed up weird permission bits as well as our SGML documentation code in commit 936 and commit 937.
- Full IPv6 parity. If configured to use IPv6 for outgoing queries (using **query-local-address6=::0** for example), IPv6 and IPv4 addresses are finally treated 100% identically, instead of 'mostly'. This feature is implemented using 'ANY' queries to find A and AAAA addresses in one query, which is a new approach. Treat with caution.
- Now perform EDNS0 root refreshing queries, so as to benefit from all returned addresses. Relevant since early February 2008 when the root-servers started to respond with IPv6 addresses, which made the default non-EDNS0 maximum packet length reply no longer contain all records. Implemented in commit 1130. Thanks to dns-operations AT mail.oarc.isc.org for quick suggestions on how to deal with this change.
- **rec_control** now has a timeout in case the Recursor does not respond. Implemented in commit 945.
- (Error) messages are now logged with saner priorities (commit 955).
- Outbound query IP interface stemmed from 1997 (!) and was in dire need of a cleanup (commit 1117).
- L.ROOT-SERVERS.NET moved (commit 1118).

17.13.22 Recursor version 3.1.4

Released the 13th of November 2006.

This release contains almost no new features, but consists mostly of minor and major bug fixes. It also addresses two major security issues, which makes this release a highly recommended upgrade.

Security issues

- Large TCP questions followed by garbage could cause the recursor to crash. This critical security issue has been assigned CVE-2006-4251, and is fixed in commit 915. More information can be found in *"PowerDNS Security Advisory 2006-01: Malformed TCP queries can lead to a buffer overflow which might be exploitable"*.
- CNAME loops with zero second TTLs could cause crashes in some conditions. These loops could be constructed by malicious parties, making this issue a potential denial of service attack. This security issue has been assigned CVE-2006-4252 and is fixed by commit 919. More information can be found in *"PowerDNS Security Advisory 2006-02: Zero second CNAME TTLs can make PowerDNS exhaust allocated stack space, and crash"*. Many thanks to David Gavarret for helping pin down this problem.

Bugs

- On certain error conditions, PowerDNS would neglect to close a socket, which might therefore eventually run out. Spotted by Stefan Schmidt, fixed in commits 892, 897, 899.
- Some nameservers (including PowerDNS in rare circumstances) emit a SOA record in the authority section. The recursor mistakenly interpreted this as an authoritative "NXRRSET". Spotted by Bryan Seitz, fixed in commit 893.
- In some circumstances, PowerDNS could end up with a useless (not working, or no longer working) set of nameserver records for a domain. This release contains logic to invalidate such broken NSSETs, without overloading authoritative servers. This problem had previously been spotted by Bryan Seitz, 'Cerb' and Darren Gamble. Invalidations of NSSETs can be plotted using the "nsset-invalidations" metric, available through **rec_control get**. Implemented in commit 896 and commit 901.
- PowerDNS could crash while dumping the cache using **rec_control dump-cache**. Reported by Wouter of WideXS and Stefan Schmidt and many others, fixed in commit 900.
- Under rare circumstances (depleted TCP buffers), PowerDNS might send out incomplete questions to remote servers. Additionally, on big-endian systems (non-Intel and non-AMD generally), sending out large TCP answers questions would not work at all, and possibly crash. Brought to our attention by David Gavarret, fixed in commit 903.
- The recursor contained the potential for a dead-lock processing an invalid domain name. It is not known how this might be triggered, but it has been observed by 'Cerb' on #powerdns. Several dead-locks where PowerDNS consumed all CPU, but did not answer questions, have been reported in the past few months. These might be fixed by commit 904.
- IPv6 'allow-from' matching had problems with the least significant bits, sometimes allowing disallowed addresses, but mostly disallowing allowed addresses. Spotted by Wouter from WideXS, fixed in commit 916.

Improvements

- PowerDNS has support to drop answers from so called 'delegation only' zones. A statistic ("dlg-only-drops") is now available to plot how often this happens. Implemented in commit 890.
- Hint-file parameter was mistakenly named "hints-file" in the documentation. Spotted by my Marco Davids, fixed in commit 898.

- **rec_control quit** should be near instantaneous now, as it no longer meticulously cleans up memory before exiting. Problem spotted by Darren Gamble, fixed in commit 914, closing ticket 84.
- init.d script no longer refers to the Recursor as the Authoritative Server. Spotted by Wouter of WideXS, fixed in commit 913.
- A potentially serious warning for users of the GNU C Library version 2.5 was fixed. Spotted by Marcus Rueckert, fixed in commit 920.

17.13.23 Recursor version 3.1.3

Released the 12th of September 2006.

Compared to 3.1.2, this release again consists of a number of mostly minor bug fixes, and some slight improvements.

Many thanks are again due to Darren Gamble who together with his team has discovered many misconfigured domains that do work with some other name servers. DNS has long been tolerant of misconfigurations, PowerDNS intends to uphold that tradition. Almost all of the domains found by Darren now work as well in PowerDNS as in other name server implementations.

Thanks to some recent migrations, this release, or something very close to it, is powering over 40 million internet connections that we know of. We appreciate hearing about successful as well as unsuccessful migrations, please feel free to notify pdns.bd@powerdns.com of your experiences, good or bad.

Bug-fixes

- The MThread default stack size was too small, which led to problems, mostly on 64-bit platforms. This stack size is now configurable using the **stack-size** setting should our estimate be off. Discovered by Darren Gamble, Sten Spans and a number of others. Fixed in commit 868.
- Plug a small memory leak discovered by Kai and Darren Gamble, fixed in commit 870.
- Switch from the excellent nedmalloc to dlmalloc, based on advice by the nedmalloc author. Nedmalloc is optimised for multithreaded operation, whereas the PowerDNS recursor is single threaded. The version of nedmalloc shipped contained a number of possible bugs, which are probably resolved by moving to dlmalloc. Some reported crashes on hitting 2G of allocated memory on 64 bit systems might be solved by this switch, which should also increase performance. See commit 873 for details.

Improvements

- The cache is now explicitly aware of the difference between authoritative and unauthoritative data, allowing it to deal with some domains that have different data in the parent zone than in the authoritative zone. Patch in commit 867.
- No longer try to parse DNS updates as if they were queries. Discovered and fixed by Jan Gyselinck, fix in commit 871.
- Rebalance logging priorities for less log cluttering and add IP address to a remote server error message. Noticed and fixed by Jan Gyselinck (commit 877).
- Add **logging-facility** setting, allowing syslog to send PowerDNS logging to a separate file. Added in commit 871.

17.13.24 Recursor version 3.1.2

Released Monday 26th of June 2006.

Compared to 3.1.1, this release consists almost exclusively of bug-fixes and speedups. A quick update is recommended, as some of the bugs impact operators of authoritative zones on the internet. This version has been tested by some of the largest internet providers on the planet, and is expected to perform well for everybody.

Many thanks are due to Darren Gamble, Stefan Schmidt and Bryan Seitz who all provided excellent feedback based on their large-scale tests of the recursor.

Bug-fixes

- Internal authoritative server did not differentiate between 'NXDOMAIN' and 'NXRRSET', in other words, it would answer 'no such host' when an AAAA query came in for a domain that did exist, but did not have an AAAA record. This only affects users with **auth-zones** configured. Discovered by Bryan Seitz, fixed in commit 848.
- ANY queries for hosts where nothing was present in the cache would not work. This did not cause real problems as ANY queries are not reliable (by design) for anything other than debugging, but did slow down the nameserver and cause unnecessary load on remote nameservers. Fixed in commit 854.
- When exceeding the configured maximum amount of TCP sessions, TCP support would break and the nameserver would waste CPU trying to accept TCP connections on UDP ports. Noted by Bryan Seitz, fixed in commit 849.
- DNS queries come in two flavours: recursion desired and non-recursion desired. The latter is not very useful for a recursor, but is sometimes (erroneously) used by monitoring software or load balancers to detect nameserver availability. A non-rd query would not only not recurse, but also not query authoritative zones, which is confusing. Fixed in commit 847.
- Non-standard DNS TCP queries, that did occur however, could drive the recursor to 100% CPU usage for extended periods of time. This did not disrupt service immediately, but does waste a lot of CPU, possibly exhausting resources. Discovered by Bryan Seitz, fixed in commit 858, which is post-3.1.2-rc1.
- The PowerDNS recursor did not honour the rare but standardised 'ANY' query class (normally 'ANY' refers to the query type, not class), upsetting the Wildfire Jabber server. Discovered and debugged by Daniel Nauck, fixed in commit 859, which is post-3.1.2-rc1.
- Everybody's favorite, when starting up under high load, a bogus line of statistics was sometimes logged. Fixed in commit 851.
- Remove some spurious debugging output on dropping a packet by an unauthorized host. Discovered by Kai. Fixed in commit 854.

Improvements

- Misconfigured domains, with a broken nameserver in the parent zone, should now work better. Changes motivated and suggested by Darren Gamble. This makes PowerDNS more compliant with RFC 2181 by making it prefer authoritative data over non-authoritative data. Implemented in commit 856.
- PowerDNS can now listen on multiple ports, using the **local-address** setting. Added in commit 845.
- A number of speedups which should have a noticeable impact, implemented in commits 850, 852, 853, 855
- The recursor now works around an issue with the Linux kernel 2.6.8, as shipped by Debian. Fixed by Christof Meerwald in commit 860, which is post 3.1.2-rc1.

17.13.25 Recursor version 3.1.1

Released on the 23rd of May 2006.

Warning: 3.1.1 is identical to 3.1 except for a bug in the packet chaining code which would mainly manifest itself for IPv6 enabled Konqueror users with very fast connections to their PowerDNS installation. However, all 3.1 users are urged to upgrade to 3.1.1. Many thanks to Alessandro Bono for his quick aid in solving this problem.

Many thanks are due to the operators of some of the largest internet access providers in the world, each having many millions of customers, who have tested the various 3.1 pre-releases for suitability. They have uncovered and helped fix bugs that could impact us all, but are only (quickly) noticeable with such vast amounts of DNS traffic.

After version 3.0.1 has proved to hold up very well under tremendous loads, 3.1 adds important new features

- Ability to serve authoritative data from 'BIND' style zone files (using **auth-zones** statement).
- Ability to forward domains so configured to external servers (using **forward-zones**).
- Possibility of 'serving' the contents of `/etc/hosts` over DNS, which is very well suited to simple domestic router/DNS setups. Enabled using **export-etc-hosts**.
- As recommended by recent standards documents, the PowerDNS recursor is now authoritative for RFC-1918 private IP space zones by default (suggested by Paul Vixie).
- Full outgoing IPv6 support (off by default) with IPv6 servers getting equal treatment with IPv4, nameserver addresses are chosen based on average response speed, irrespective of protocol.
- Initial Windows support, including running as a service ('NET START "POWERDNS RECURSOR"'). **rec_channel** is still missing, the rest should work. Performance appears to be below that of the UNIX versions, this situation is expected to improve.

Bug fixes

- No longer send out SRV and MX record priorities as zero on big-endian platforms (UltraSPARC). Discovered by Eric Sproul, fixed in commit 773.
- SRV records need additional processing, especially in an Active Directory setting. Reported by Kenneth Marshall, fixed in commit 774.
- The root-records were not being refreshed, which could lead to problems under inconceivable conditions. Fixed in commit 780.
- Fix resolving domain names for nameservers with multiple IP addresses, with one of these addresses being lame. Other nameserver implementations were also unable to resolve these domains, so not a big bug. Fixed in commit 780.
- For a period of 5 minutes after expiring a negative cache entry, the domain would not be re-cached negatively, leading to a lot of duplicate outgoing queries for this short period. This fix has raised the average cache hit rate of the recursor by a few percent. Fixed in commit 783.
- Query throttling was not aggressive enough and not all sorts of queries were throttled. Implemented in commit 786.
- Fix possible crash during startup when parsing empty configuration lines (commit 807).
- Fix possible crash when the first query after wiping a cache entry was for the just deleted entry. Rare in production servers. Fixed in commit 820.
- Recursor would send out differing TTLs when receiving a misconfigured, standards violating, RRSET with different TTLs. Implement fix as mandated by RFC 2181, paragraph 5.2. Reported by Stephen Harker (commit 819).
- The **top-remotes** would list remotes more than once, once per source port. Discovered by Jorn Ekkelenkamp, fixed in commit 827, which is post 3.1-pre1.
- Default **allow-from** allowed queries from fe80::/16, corrected to fe80::/10. Spotted by Niels Bakker, fixed in commit 829, which is post 3.1-pre1.
- While PowerDNS blocks failing queries quickly, multiple packets could briefly be in flight for the same domain and nameserver. This situation is now explicitly detected and queries are chained to identical queries already in flight. Fixed in commit 833 and commit 834, post 3.1-pre1.

Improvements

- ANY queries are now implemented as in other nameserver implementations, leading to a decrease in outgoing queries. The RFCs are not very clear on desired behaviour, what is implemented now saves bandwidth

and CPU and brings us in line with existing practice. Previously ANY queries were not cached by the PowerDNS recursor. Implemented in commit 784.

- **rec_control** was very sparse in its error reporting, and user unfriendly as well. Reported by Erik Bos, fixed in commit 818 and commit 820.
- IPv6 addresses were printed in a non-standard way, fixed in commit 788.
- TTLs of records are now capped at two weeks, commit 820.
- **allow-from** IPv4 netmasks now automatically work for IP4-to-IPv6 mapper IPv4 addresses, which appear when running on the wildcard **::** IPv6 address. Lack of feature noted by Marcus 'darix' Rueckert. Fixed in commit 826, which is post 3.1-pre1.
- Errors before daemonizing are now also sent to syslog. Suggested by Marcus 'darix' Rueckert. Fixed in commit 825, which is post 3.1-pre1.
- When launching without any form of configured network connectivity, all root-servers would be cached as 'down' for some time. Detect this special case and treat it as a resource-constraint, which is not accounted against specific nameservers. Spotted by Seth Arnold, fixed in commit 835, which is post 3.1-pre1.
- The recursor now does not allow authoritative servers to keep supplying its own NS records into perpetuity, which causes problems when a domain is redelegated but the old authoritative servers are not updated to this effect. Noticed and explained at length by Darren Gamble of Shaw Communications, addressed by commit 837, which is post 3.1-pre2.
- Some operators may want to follow RFC 2181 paragraph 5.2 and 5.4. This harms performance and does not solve any real problem, but does make PowerDNS more compliant. If you want this, enable **auth-can-lower-ttl**. Implemented in commit 838, which is post 3.1-pre2.

17.13.26 Recursor version 3.0.1

Released 25th of April 2006, download.

This release consists of nothing but tiny fixes to 3.0, including one with security implications. An upgrade is highly recommended.

- Compilation used both `cc` and `gcc`, leading to the possibility of compiling with different compiler versions (commit 766).
- **rec_control** would leave files named `lsockXXXXXX` around in the configured socket-dir. Operators may wish to remove these files from their socket-dir (often `/var/run`), quite a few might have accumulated already (commit 767).
- Certain malformed packets could crash the recursor. As far as we can determine these packets could only lead to a crash, but as always, there are no guarantees. A quick upgrade is highly recommended (commits 760, 761). Reported by David Gavarret.
- Recursor would not distinguish between NXDOMAIN and NXRRSET (commit 756). Reported and debugged by Jorn Ekkelenkamp.
- Some error messages and trace logging statements were improved (commits 756, 758, 759).
- stderr was closed during daemonizing, but not dupped to /dev/null, leading to slight chance of odd behaviour on reporting errors (commit 757)

Operating system specific fixes

- The stock Debian sarge Linux kernel, 2.6.8, claims to support epoll but fails at runtime. The epoll self-testing code has been improved, and PowerDNS will fall back to a select based multiplexer if needed (commit 758) Reported by Michiel van Es.
- Solaris 8 compilation and runtime issues were addressed. See the README for details (commit 765). Reported by Juergen Georgi and Kenneth Marshall.

- Solaris 10 x86_64 compilation issues were addressed (commit 755). Reported and debugged by Eric Sproul.

17.13.27 Recursor version 3.0

Released 20th of April 2006, download.

This is the first separate release of the PowerDNS Recursor. There are many reasons for this, one of the most important ones is that previously we could only do a release when both the recursor and the authoritative nameserver were fully tested and in good shape. The split allows us to release new versions when each part is ready.

Now for the real news. This version of the PowerDNS recursor powers the network access of over two million internet connections. Two large access providers have been running pre-releases of 3.0 for the past few weeks and results are good. Furthermore, the various pre-releases have been tested nearly non-stop with DNS traffic replayed at 3000 queries/second.

As expected, the 2 million households shook out some very rare bugs. But even a rare bug happens once in a while when there are this many users.

We consider this version of the PowerDNS recursor to be the most advanced resolver publicly available. Given current levels of spam, phishing and other forms of internet crime we think no recursor should offer less than the best in spoofing protection. We urge all operators of resolvers without proper spoofing countermeasures to consider PowerDNS, as it is a Better Internet Nameserver Daemon.

Some more information, based on a previous version of PowerDNS, can be found on the PowerDNS development blog.

Warning: Because of recent DNS based denial of service attacks, running an open recursor has become a security risk. Therefore, unless configured otherwise this version of PowerDNS will only listen on localhost, which means it does not resolve for hosts on your network. To fix, configure the **local-address** setting with all addresses you want to listen on. Additionally, by default service is restricted to RFC 1918 private IP addresses. Use **allow-from** to selectively open up the recursor for your own network. See pdns_recursor settings for details.

Important new features of the PowerDNS recursor 3.0

- Best spoofing protection and detection we know of. Not only is spoofing made harder by using a new network address for each query, PowerDNS detects when an attempt is made to spoof it, and temporarily ignores the data. For details, see Anti-spoofing.
- First nameserver to benefit from epoll/kqueue/Solaris completion ports event reporting framework, for stellar performance.
- Best statistics of any recursing nameserver we know of, see Statistics.
- Last-recently-used based cache cleanup algorithm, keeping the 'best' records in memory
- First class Solaris support, built on a 'try and buy' Sun CoolThreads T 2000.
- Full IPv6 support, implemented natively.
- Access filtering, both for IPv4 and IPv6.
- Experimental SMP support for nearly double performance. See PowerDNS Recursor performance.

Many people helped package and test this release. Jorn Ekkelenkamp of ISP-Services helped find the '8000 SOAs' bug and spotted many other oddities and XS4ALL internet funded a lot of the recent development. Joaquín M López Muñoz of the boost::multi_index_container was again of great help.

CHAPTER

EIGHTEEN

NEWLY OBSERVED DOMAIN TRACKING

A common security technique for detecting domains that may be suspicious or be associated with bad actors such as hosting malware, phishing or botnet command and control, is to investigate domains that haven't been seen before, i.e. are newly observed.

Deciding whether a domain is truly a new domain would involve deterministic methods, such as maintaining a database of all domains ever seen, and comparing all domain lookups against that database. Such a mechanism would not be scalable in a recursor, and so is best suited to offline analysis. However, determining candidate domains for such an offline service is a problem that can be solved in the recursor, given that sending all domain lookups to such an offline service would still be prohibitively costly, and given that the true number of newly observed domains is likely to be relatively small in a given time period.

A simple method to determine a candidate domain would simply be to check if the domain was not in the recursor cache; indeed this is a method used by many security researchers. However, while that does produce a smaller list of candidate domains, cache misses are still relatively common, particularly in deployments where techniques such as EDNS client-subnet are used.

Therefore, a feature has been developed for the recursor which uses probabilistic data structures (specifically a Stable Bloom Filter (SBF): [http://webdocs.cs.ualberta.ca/~drafiei/papers/DupDet06Sigmod.pdf]). This recursor feature is named "Newly Observed Domain" or "NOD" for short.

The use of a probabilistic data structure means that the memory and CPU usage for the NOD feature is minimal, however it does mean that there can be false positives (a domain flagged as new when it is not), and false negatives (a domain that is new is not detected). The size of the SBF data structure can be tuned to reduce the FP/FN rate, although it is created with a default size (67108864 cells) that should provide a reasonably low FP/FN rate. To configure a different size use the `new-domain-db-size` setting to specify a higher or lower cell count. Each cell consumes 1-bit of RAM (per recursor thread) and 1-byte of disk space.

NOD is disabled by default, and must be enabled through the use of the following setting in recursor.conf:

```
new-domain-tracking=yes
```

Once enabled the recursor will keep track of previously seen domains using the SBF data structure, which is periodically persisted to the directory specified in the `new-domain-history-dir`, which defaults to /var/lib/pdns-recursor/nod.

Administrators may wish to prevent certain domains or subdomains from ever triggering the NOD algorithm, in which case those domains must be added to the `new-domain-ignore-list` setting as a comma separated list. No domain (or subdomain of a domain) listed will be considered a newly observed domain. It is also possible to use `new-domain-ignore-list-file` to read a file with ignored domains, one domain per line.

There are several ways to receive the information about newly observed domains:

18.1 Logging

The setting `new-domain-log` is enabled by default once the NOD feature is enabled, and will log the newly observed domain to the recursor logfile.

18.2 DNS Lookup

The setting `new-domain-lookup=<base domain>` will cause the recursor to issue a DNS A record lookup to `<newly observed domain>.<base domain>`. This can be a suitable method to send NOD data to an offsite or remote partner, however care should be taken to ensure that data is not leaked inadvertently. To log NOD information to a dnstap stream, refer to *`dnstapNODFrameStreamServer()`*.

18.3 Protobuf Logging

If both NOD and protobuf logging are enabled, then the `newlyObservedDomain` field of the protobuf message emitted by the recursor will be set to true. Additionally newly observed domains will be tagged in the protobuf stream using the tag `pdns-nod` by default. The setting `new-domain-pb-tag=<tag>` can be used to alter the tag.

CHAPTER

NINETEEN

UNIQUE DOMAIN RESPONSE

A similar feature to NOD is Unique Domain Response (UDR). This feature uses the same probabilistic data structures as NOD to store information about unique responses for a given lookup domain. Determining if a particular response is unique for a given lookup domain is extremely useful for determining potential security issues such as:

- Fast-Flux Domain Names
- Cache-Poisoning Attacks
- Botnet Command and Control Servers etc.

This is because well-behaved domains tend to return fairly stable results to DNS record lookups, and thus domains which don't exhibit this behaviour may be suspicious or may indicate a domain under attack.

UDR is disabled by default - to enable it, set `unique-response-tracking=yes` in recursor.conf.

The data is persisted to /var/lib/pdns-recursor/udr by default, which can be changed with the setting `unique-response-history-dir=<new directory>`.

The SBF (which is maintained separately per recursor thread) cell size defaults to 67108864, which can be changed using the setting `unique-response-db-size`. The same caveats regarding FPs/FNs apply as for NOD.

Similarly to NOD, administrators may wish to prevent certain domains or subdomains from ever triggering the UDR algorithm, in which case those domains must be added to the `udr-ignore-list` setting as a comma separated list. No domain (or subdomain of a domain) listed will be considered a new unique domain response. It is also possible to use `udr-ignore-list-file` to read a file with ignored domains, one domain per line.

Similarly to NOD, unique domain responses can be tracked using several mechanisms:

19.1 Logging

The setting `unique-response-log` is enabled by default once the NOD feature is enabled, and will log the newly observed domain to the recursor logfile. To log UDR information to a dnstap stream, refer to *`dnstapNODFrameStreamServer()`*.

19.2 Protobuf Logging

If both UDR and protobuf logging are enabled, then unique domain responses will be tagged in the protobuf stream using the tag `pdns-udr` by default. The setting `unique-response-pb-tag=<tag>` can be used to alter the tag.

CHAPTER

TWENTY

END OF LIFE STATEMENTS

We aim to have a major release every six months. The latest major release train receives correctness, stability and security updates by the way of minor releases. We support older releases with critical updates for one year after the following major release.

Older releases are marked end of life and receive no updates at all. Pre-releases do not receive immediate security updates.

The currently supported release train of the PowerDNS Recursor is 5.1.

PowerDNS Recursor 5.0 will only receive critical updates and will be End of Life one year after PowerDNS Recursor 5.1 was released.

PowerDNS Recursor 4.9 will only receive critical updates and will be End of Life one year after PowerDNS Recursor 5.0 was released.

PowerDNS Recursor 4.0 through 4.8, 3.x, and 2.x are End of Life.

Note: Users with a commercial agreement with PowerDNS.COM BV or Open-Xchange can receive extended support for releases which are End Of Life. If you are such a user, these EOL statements do not apply to you. Please refer to the support commitment for details. Note that for the Open Source support channels we only support the latest minor release of a release train. That means that we ask you to reproduce potential issues on the latest minor release first.

Table 1: PowerDNS Recursor Release Life Cycle

Version	Release date	Critical-Only updates	End of Life
5.1	July 10 2024	~ February 2025	~ February 2026
5.0	January 10 2024	July 10 2024	July 10 2025
4.9	June 30 2023	January 10 2024	January 10 2025
4.8	December 12 2022	June 30 2023	EOL June 30 2024
4.7	May 30 2022	December 12 2022	EOL January 10 2024
4.6	December 17 2021	May 30 2022	EOL June 30 2023
4.5	May 11 2021	December 17 2021	EOL December 12 2022
4.4	October 19 2020	May 11 2021	EOL May 30 2022
4.3	March 3 2020	October 19 2020	EOL December 17 2021
4.2	July 16 2019	October 19 2020	EOL May 11 2021
4.1	December 4 2017	March 3 2020	EOL October 19 2020
4.0 and older	EOL	EOL	EOL

Note we did not have a very regular release schedule in the past, so the dates for older releases do not follow the pattern described above.

CHAPTER

TWENTYONE

FREQUENTLY ASKED QUESTIONS

This document lists categorized answers and questions with links to the relevant documentation.

21.1 EDNS bufsize in response packets

You may have spotted the 512 in something like the following (after `EDNS ... udp:`):

```
$ dig example.com @127.0.0.1

; <<>> DiG 9.11.5-P4-5.1+deb10u3-Debian <<>> example.com @127.0.0.1
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 20155
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 512
;; QUESTION SECTION:
;example.com.           IN  A

;; ANSWER SECTION:
example.com.        43200   IN  A   93.184.216.34

;; Query time: 86 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Thu Apr 15 13:56:34 CEST 2021
;; MSG SIZE  rcvd: 56
```

and wonder 'why is the Recursor using a bufsize of 512? Did we not decide on a Flag Day, all together, that we would use 1232?'

The EDNS buffer size in a DNS packet, generated by side A, tells the recipient of that packet (side B) the maximum packet size that side A will accept from side B. So, when the Recursor talks to an Authoritative, the Recursor reports the buffer size the Authoritative is allowed to use to it - usually 1232 (*edns-outgoing-bufsize*). But the example above is the Recursor responding to a client, and it is telling the client 'from you, I accept packets of up to 512 bytes'. Or, to say it differently, the Recursor is telling the client that *questions* must fit in 512 bytes. This is fine for the Recursor - unlike an Authoritative, that might need to handle big UPDATE requests, the Recursor really only answers simple questions from clients, and those always comfortably fit in 512 bytes, because the maximum length of a DNS name is 256 bytes.

Similarly, the maximum size of a response from the Recursor to a client is governed by the buffer size sent by the client (in `dig`, you can see that number by doing `dig +qr`), and the *udp-truncation-threshold* setting in the Recursor configuration.

To see the buffer size the Recursor is sending to authoritatives, you can ask the question below, which gets sent to an authoritative server reporting in a TXT answer record what it saw in the query:

```
$ dig txt header.lua.powerdns.org +short @127.0.0.1
"id: 52938, aa: false, rd: false, ad: false, cd: false, do: true, ednsbufsiz: 1232,
↪ tcp: false"
```

Or, in a diagram:

```
    udp-truncation-threshold      edns-outgoing-bufsize
        [default is 1232]           [default is 1232]
              |                           |
+------+      v      +----------+         v             +------------+
| stub | <=========> | recursor | <===================> | responders |
+------+             +----------+                       +------------+
             ^
             |
  client bufsize (stub => recursor)
 bufsize reported to client (recursor => stub [always 512])
```

21.2 Handling of root hints

On startup, the **Recursor** uses root hints to resolve the names and addresses of the root name servers and puts the record sets found into the record cache. This is needed to be able to resolve names, as the recursive algorithm starts at the root (using cached data) and then tries to resolve delegations until it finds the name servers that are authoritative for the domain in question.

If the *hint-file* is not set, **Recursor** wil use a compiled-in table as root hints.

Periodically, based on the *max-cache-ttl*, the **Recursor** will refetch the root data using data in its cache by doing a . *NS* query. If that does not succeed, it will fall back to using the root hints to fill the cache with root data. Prior to version 4.7.0, the period for re-fetching root data was *max-cache-ttl* divided by 12, with a minimum of 10 seconds. Starting with version 4.7.0, the period is adaptive, starting at 80% of *max-cache-ttl*, reducing the interval on failure.

The root hints and resolved root data can differ if the root hints are outdated. As long as at least one root server mentioned in the root hints can be contacted, the periodic refresh will produce the desired record sets corresponding to the current up-to-date root server data.

Starting with version 4.6.2, if *hint-file* is set to `no`, the **Recursor** will not prime the cache with root data obtained from hints, but will still do the periodic refresh. A (recursive) forward configuration is needed to make the periodic refresh work.

Starting with version 4.9, setting *hint-file* to `no-refresh` disables both the initial reading of the hints and the periodic refresh of cached root data. This prevents **Recursor** from resolving names by itself, so it is only useful in cases where all queries are forwarded.

With versions older than 4.8, there is another detail: after refreshing the root records, the **Recursor** will resolve the `NS` records for the top level domain of the root servers. For example, in the default setup the root name servers are called `[a-m].root-servers.net`, so the **Recursor** will resolve the name servers of the `.net` domain. This is needed to correctly determine zone cuts to be able to decide if the `.root-servers.net` domain is DNSSEC protected. Newer versions solve this by querying the needed information top-down.

Starting with version 5.0.0, enabling *allow-no-rd* allows for queries without the recursion desired bit to be answered from cache. Older versions of the `dig` program provided by ISC do not set the RD bit on the initial `+trace` query causing it to sometimes fail to perform a `+trace` when asking a freshly restarted **Recursor** despite the *allow-no-rd* option being set. This is because there is a short while after restarting that the cache has no authoritative data on the root, so it will answer with an NODATA (NOERROR and no answer records) in that period for RD=0 queries asking for the root name servers. For `dig` this has been fixed in BIND 9.15.1 by setting the RD bit.

CHAPTER

TWENTYTWO

COMPILING POWERDNS RECURSOR

As **PowerDNS Recursor** is distributed with a configure script, compiling it is a matter of:

```
tar xf pdns-recursor-$VERSION.tar.bz2
cd pdns-recursor-$VERSION
./configure
make
make install
```

22.1 Getting the sources

Warning: Do not use the tarballs auto-generated by GitHub from the tags, as these are not proper release tarballs.

There are 3 ways of getting the source.

If you want the bleeding edge, you can clone the repository at GitHub and run `autoreconf -vi` in the `pdns/recursordist` directory of the clone.

You can also download snapshot tarballs.

You can also download releases on the website. These releases are PGP-signed with one of these key-ids:

- FBAE 0323 821C 7706 A5CA 151B DCF5 13FA 7EED 19F3
- D630 0CAB CBF4 69BB E392 E503 A208 ED4F 8AF5 8446
- 16E1 2866 B773 8C73 976A 5743 6FFC 3343 9B0D 04DF
- 990C 3D0E AC7C 275D C6B1 8436 EACA B90B 1963 EC2B

There is a PGP keyblock with these keys available on https://doc.powerdns.com/powerdns-keyblock.asc.

Older releases (4.3.x and earlier) can also be signed with one of the following keys:

- 1628 90D0 689D D12D D33E 4696 1C5E E990 D2E7 1575
- B76C D467 1C09 68BA A87D E61C 5E50 715B F2FF E1A7

22.2 Dependencies

To build **PowerDNS Recursor**, a C++ compiler with support for C++ 2017 is required. This means `gcc 5` and newer and `clang 5` and newer. Furthermore, the Makefiles require GNU `make`, not BSD `make`. Starting with version 5, a Rust compiler is needed.

By default, the **Recursor** requires the following libraries and headers:

- Boost 1.35 or newer

- Lua 5.1+ or LuaJit
- OpenSSL
- For **Recursor** version 5 and higher, cargo version 1.64 or newer.

Note: On Debian and Ubuntu, the following will get you the dependencies:

```
apt-get install libboost-dev libboost-filesystem-dev libboost-serialization-dev \
   libboost-system-dev libboost-thread-dev libboost-context-dev \
   libboost-test-dev libssl-dev libboost-test-dev g++ make pkg-config \
   libluajit-5.1-dev cargo
```

22.3 Compiling from a git checkout

Source code is available on GitHub:

```
git clone https://github.com/PowerDNS/pdns.git
```

This repository contains the sources for the PowerDNS Recursor, the PowerDNS Authoritative Server, and dnsdist (a powerful DNS loadbalancer). The sources for the recursor are located in the *pdns/recursordist* subdirectory of the repository.

To compile from a git checkout, install the dependencies above plus ragel, automake, autoconf, libtool, virtualenv and curl. Then run:

```
cd pdns/pdns/recursordist/
autoreconf -vi
./configure
make
```

22.4 macOS Notes

If you want to compile yourself, the dependencies can be installed using Homebrew. You need to tell configure where to find OpenSSL, too:

```
brew install boost lua pkg-config ragel openssl
./configure PKG_CONFIG_PATH=/usr/local/opt/openssl/lib/pkgconfig
make -j4
```

22.4.1 Lua scripting

To benefit from Lua scripting, as described on https://doc.powerdns.com/md/recursor/scripting/ Install Lua and development headers. PowerDNS supports Lua 5.1, 5.2, 5.3 and LuaJIT. On Debian/Ubuntu, install e.g. *liblua5.2-dev* to use Lua 5.2.

The configure script will automatically detect the Lua version. If more than one version of Lua is installed, the *–with-lua* configure flag can be set to the desired version. e.g.:

```
./configure --with-lua=lua51
```

(On older versions of Debian/Ubuntu, you'll need to pass *–with-lua=lua5.1* instead.)

22.5 Optional dependencies

Several options that can be passed to `./configure` can enable and disable different features. These will require additional dependencies

22.5.1 ed25519 support with libsodium

The **Recursor** can link with libsodium to support ed25519 (DNSSEC algorithm 15). To detect libsodium, use the `--with-libsodium` configure option.

Changed in version 4.2.0: This option was previously `--enable-libsodium`

22.5.2 ed25519 and ed448 support with libdecaf

libdecaf is a library that allows **Recursor** to support ed25519 and Ed448 (DNSSEC algorithms 15 and 16). To detect libdecaf, use the `--with-libdecaf` configure option.

Changed in version 4.2.0: This option was previously `--enable-libdecaf`

22.5.3 Protobuf to emit DNS logs

The **Recursor** can log DNS query information over *Protocol Buffers*.

This functionality from 4.5.0 and upwards, without needing any external library. Before 4.5.0, installing the protobuf library and compiler is required to enable this functionality. The configure script will automatically detect this and bump the Boost version dependency to 1.42. To disable building this functionality before 4.5.0, use `--without-protobuf`.

22.5.4 `systemd` notify support

During configure, `configure` will attempt to detect the availability of systemd or systemd-daemon headers. To force the use of `systemd` (and failing configure if the headers do not exist), use `--enable-systemd`. To set the directory where the unit files should be installed, use `--with-systemd=/path/to/unit/dir`.

Note: If you want systemd support, you will need to install the corresponding development package. On Debian and Ubuntu, this means *apt install libsystemd-dev*.

22.6 Documentation

After compiling, run *pdns_recursor –config* to view the configuration options and a short description. The full documentation is online at https://doc.powerdns.com/recursor/

CHAPTER

TWENTYTHREE

CRYPTOGRAPHIC SOFTWARE AND EXPORT CONTROL

In certain legal climates, PowerDNS might potentially require an export control status, particularly since PowerDNS software contains cryptographic primitives.

PowerDNS does not itself implement any cryptographic algorithms but relies on third-party implementations of AES, RSA, ECDSA, GOST, MD5 and various SHA-based hashing algorithms.

Starting with 4.0.0, PowerDNS will link in hash and cryptographic primitives from the open source OpenSSL library.

Optionally, PowerDNS can link in a copy of the open source Botan cryptographic library. Starting with 4.2.0, linking in Botan is no longer possible.

Optionally, PowerDNS can link in a copy of the open source Sodium library.

23.1 Specific United States Export Control Notes

PowerDNS is not "US Origin" software. For re-export, like most open source, publicly available "mass market" projects, PowerDNS is considered to be governed by section 740.13(e) of the US EAR, "Unrestricted encryption source code", under which PowerDNS source code would be considered re-exportable from the US without an export license under License Exception TSU (Technology and Software - Unrestricted).

Like most open source projects containing some encryption, the ECCN that best fits PowerDNS software is 5D002.

The official link to the publicly available source code is *https://downloads.powerdns.com/releases*.

If absolute certainty is required, we recommend consulting an expert in US Export Control, or asking the BIS for confirmation.

CHAPTER

TWENTYFOUR

INTERNALS OF THE POWERDNS RECURSOR

Warning: This section is aimed at programmers wanting to contribute to the recursor, or to help fix bugs. It is not required reading for a PowerDNS operator, although it might prove interesting.

This Recursor depends on the use of some fine infrastructure: MTasker, MOADNSParser, MPlexer and the C++ Standard Library/Boost. This page will explain the conceptual relation between these components, and the route of a packet through the program.

24.1 The PowerDNS Recursor

The Recursor started out as a tiny project, mostly a technology demonstration. These days it is a full blown recursor with many features. This combined with a need for very high performance has made the recursor code less accessible than it was. The page you are reading hopes to rectify this situation.

24.2 Synchronous code using MTasker

The original name of the program was **syncres**, which is still reflected in the file name `syncres.cc`, and the class SyncRes. This means that PowerDNS is written naively, with one thread of execution per query, synchronously waiting for packets, Normally this would lead to very bad performance (unless running on a computer with very fast threading, like possibly the Sun CoolThreads family), so PowerDNS employs MTasker for very fast userspace threading.

MTasker, which was developed separately from PowerDNS, does not provide a full multithreading system but restricts itself to those features a nameserver needs. It offers cooperative multitasking, which means there is no forced preemption of threads. This in turn means that no two **MThreads** ever really run at the same time.

This is both good and bad, but mostly good. It means the recursor does not have to think about locking in many cases.

It also means that the recursor could block if any operation takes too long.

The core interaction with MTasker are the waitEvent() and sendEvent() functions. These pass around PacketID objects. Everything PowerDNS needs to wait for is described by a PacketID event, so the name is a bit misleading. Waiting for a TCP socket to have data available is also passed via a PacketID, for example.

The version of MTasker in PowerDNS is newer than that described at the MTasker site, with a vital difference being that the waitEvent() structure passes along a copy of the exact PacketID sendEvent() transmitted. Furthermore, threads can trawl through the list of events being waited for and modify the respective PacketIDs. This is used for example with **near miss** packets: packets that appear to answer questions we asked, but differ in the DNS id. On seeing such a packet, the recursor trawls through all PacketIDs and if it finds any nearmisses, it updates the PacketID::nearMisses counter. The actual PacketID thus lives inside MTasker while any thread is waiting for it.

24.3 MPlexer

The Recursor uses a separate socket per outgoing query. This has the important benefit of making spoofing 64000 times harder, and additionally means that ICMP errors are reported back to the program. In measurements this appears to happen to one in ten queries, which would otherwise take a two-second timeout before PowerDNS moves on to another nameserver.

However, this means that the program routinely needs to wait on hundreds or even thousands of sockets. Different operating systems offer various ways to monitor the state of sockets or more generally, file descriptors. To abstract out the differing strategies (`select`, `epoll`, `kqueue`, `completion ports`), PowerDNS contains **MPlexer** classes, all of which descend from the FDMultiplexer class.

This class is very simple and offers only five important methods: addReadFD(), addWriteFD(), removeReadFD(), removeWriteFD() and run.

The arguments to the **add** functions consist of an fd, a callback, and a boost::any variable that is passed as a reference to the callback.

This might remind you of the MTasker above, and it is indeed the same trick: state is stored within the MPlexer. As long as a file descriptor remains within either the Read or Write active list, its state will remain stored.

On arrival of a packet (or more generally, when an FD becomes readable or writable, which for example might mean a new TCP connection), the callback is called with the aforementioned reference to its parameter.

The callback is free to call removeReadFD() or removeWriteFD() to remove itself from the active list.

PowerDNS defines such callbacks as newUDPQuestion(), newTCPConnection(), handleRunningTCPConnection().

Finally, the run() method needs to be called whenever the program is ready for new data. This happens in the main loop in pdns_recursor.cc. This loop is what MTasker refers to as **the kernel**. In this loop, any packets or other MPlexer events get translated either into new MThreads within MTasker, or into calls to sendEvent(), which in turn wakes up other MThreads.

24.4 MOADNSParser

Yes, this does stand for **the Mother of All DNS Parsers**. And even that name does not do it justice! The MOADNSParser is the third attempt I've made at writing DNS packet parser and after two miserable failures, I think I've finally gotten it right.

Writing and parsing DNS packets, and the DNS records it contains, consists of four things:

1. Parsing a DNS record (from packet) into memory
2. Generating a DNS record from memory (to packet)
3. Writing out memory to user-readable zone format
4. Reading said zone format into memory

This gets tedious very quickly, as one needs to implement all four operations for each new record type, and there are dozens of them.

While writing the MOADNSParser, it was discovered there is a remarkable symmetry between these four transitions. DNS Records are nearly always laid out in the same order in memory as in their zone format representation. And reading is nothing but inverse writing.

So, the MOADNSParser is built around the notion of a **Conversion**, and we write all Conversion types once. So we have a Conversion from IP address in memory to an IP address in a DNS packet, and vice versa. And we have a Conversion from an IP address in zone format to memory, and vice versa.

This in turn means that the entire implementation of the ARecordContent is as follows (wait for it!)

```
conv.xfrIP(d_ip);
```

Through the use of the magic called `c++ Templates`, this one line does everything needed to perform the four operations mentioned above.

At one point, I got really obsessed with PowerDNS memory use. So, how do we store DNS data in the PowerDNS recursor? I mentioned **memory** above a lot - this means we could just store the DNSRecordContent objects. However, this would be wasteful.

For example, storing the following:

```
www.example.org  3600 IN   CNAME  outpost.example.org.
```

Would duplicate a lot of data. So, what is actually stored is a partial DNS packet. To store the CNAMEDNSRecordContent that corresponds to the above, we generate a DNS packet that has **www.example.org IN CNAME** as its question. Then we add **3600 IN CNAME outpost.example.org**. as its answer. Then we chop off the question part, and store the rest in the **www.example.org IN CNAME** key in our cache.

When we need to retrieve **www.example.org IN CNAME**, the inverse happens. We find the proper partial packet, prefix it with a question for **www.example.org IN CNAME**, and expand the resulting packet into the answer **3600 IN CNAME outpost.example.org.**.

Why do we go through all these motions? Because of DNS compression, which allows us to omit the whole **.example.org.** part, saving us 9 bytes. This is amplified when storing multiple MX records which all look more or less alike. This optimization is not performed yet though.

Even without compression, it makes sense as all records are automatically stored very compactly.

The PowerDNS recursor only parses a number of **well known record types** and passes all other information across verbatim - it doesn't have to know about the content it is serving.

24.5 The C++ Standard Library / Boost

C++ is a powerful language. Perhaps a bit too powerful at times, you can turn a program into a real freakshow if you so desire.

PowerDNS generally tries not to go overboard in this respect, but we do build upon a very advanced part of the Boost C++ library: boost::multi index container.

This container provides the equivalent of SQL indexes on multiple keys. It also implements compound keys, which PowerDNS uses as well.

The main DNS cache is implemented as a multi index container object, with a compound key on the name and type of a record. Furthermore, the cache is sequenced, each time a record is accessed it is moved to the end of the list. When cleanup is performed, we start at the beginning. New records also get inserted at the end. For DNS correctness, the sort order of the cache is case insensitive.

The multi index container appears in other parts of PowerDNS, and MTasker as well.

24.6 Actual DNS Algorithm

The DNS RFCs do define the DNS algorithm, but you can't actually implement it exactly that way, it was written in 1987.

Also, like what happened to HTML, it is expected that even non-standards conforming domains work, and a sizable fraction of them is misconfigured these days.

Everything begins with SyncRes::beginResolve(), which knows nothing about sockets, and needs to be passed a domain name, dns type and dns class which we are interested in. It returns a vector of DNSResourceRecord objects, ready for writing either into an answer packet, or for internal use.

After checking if the query is for any of the hardcoded domains (localhost, version.bind, id.server), the query is passed to SyncRes::doResolve, together with two vital parameters: the `depth` and `beenthere` set. As the word **recursor** implies, we will need to recurse for answers. The **depth** parameter documents how deep we've recursed already.

The `beenthere` set prevents loops. At each step, when a nameserver is queried, it is added to the `beenthere` set. No nameserver in the set will ever be queried again for the same question in the recursion process - we know for a fact it won't help us further. This prevents the process from getting stuck in loops.

SyncRes::doResolve first checks if there is a CNAME in cache, using SyncRes::doCNAMECacheCheck, for the domain name and type queried and if so, changes the query (which is passed by reference) to the domain the CNAME points to. This is the cause of many DNS problems, a CNAME record really means **start over with this query**.

This is followed by a call do SyncRes::doCacheCheck, which consults the cache for a straight answer to the question (as possibly rerouted by a CNAME). This function also consults the so called negative cache, but we won't go into that just yet.

If this function finds the correct answer, and the answer hasn't expired yet, it gets returned and we are (almost) done. This happens in 80 to 90% of all queries. Which is good, as what follows is a lot of work.

To recap:

1. beginResolve() - entry point, does checks for hardcoded domains
2. doResolve() - start of recursion process, gets passed `depth` of 0 and empty `beenthere` set
3. doCNAMECacheCheck() - check if there is a CNAME in cache which would reroute the query
4. doCacheCheck() - see if cache contains straight answer to possibly rerouted query.

If the data we were queried for was in the cache, we are almost done. One final step, which might as well be optional as nobody benefits from it, is SyncRes::addCruft. This function does additional processing, which means that if the query was for the MX record of a domain, we also add the IP address of the mail exchanger.

24.6.1 The non-cached case

This is where things get interesting, because we start out with a nearly empty cache and have to go out to the net to get answers to fill it.

The way DNS works, if you don't know the answer to a question, you find somebody who does. Initially you have no other place to go than the root servers. This is embodied in the SyncRes::getBestNSNamesFromCache method, which gets passed the domain we are interested in, as well as the `depth` and `beenthere` parameters mentioned earlier.

From now on, assume our query will be for **''www.powerdns.com.''**. SyncRes::getBestNSNamesFromCache will first check if there are NS records in cache for `www.powerdns.com.`, but there won't be. It then checks `powerdns.com. NS`, and while these records do exist on the internet, the recursor doesn't know about them yet. So, we go on to check the cache for `com. NS`, for which the same holds. Finally we end up checking for `. NS`, and these we do know about: they are the root servers and were loaded into PowerDNS on startup.

So, SyncRes::getBestNSNamesFromCache fills out a set with the **names** of nameservers it knows about for the **''.''** zone.

This set, together with the original query **''www.powerdns.com''** gets passed to SyncRes::doResolveAt. This function can't yet go to work immediately though, it only knows the names of nameservers it can try. This is like asking for directions and instead of hearing **take the third right** you are told **go to 123 Fifth Avenue, and take a right** - the answer doesn't help you further unless you know where 123 Fifth Avenue is.

SyncRes::doResolveAt first shuffles the nameservers both randomly and on performance order. If it knows a nameserver was fast in the past, it will get queried first. More about this later.

Ok, here is the part where things get a bit scary. How does SyncRes::doResolveAt find the IP address of a nameserver? Well, by calling SyncRes::getAs (**get A records**), which in turn calls.. SyncRes::doResolve. Hang on! That's where we came from! Massive potential for loops here. Well, it turns out that for any domain which

can be resolved, this loop terminates. We do pass the `beenthere` set again, which makes sure we don't keep on asking the same questions to the same nameservers.

Ok, SyncRes::getAs will give us the IP addresses of the chosen root-server, because these IP addresses were loaded on startup. We then ask these IP addresses (nameservers can have several) for its best answer for **''www.powerdns.com.''**. This is done using the LWRes class and specifically LWRes::asyncresolve, which gets passed domain name, type and IP address. This function interacts with MTasker and MPlexer above in ways which needn't concern us now. When it returns, the LWRes object contains the best answers the queried server had for our domain, which in this case means it tells us about the nameservers of `com.`, and their IP addresses.

All the relevant answers it gives are stored in the cache (or actually, merged), after which SyncRes::doResolveAt (which we are still in) evaluates what to do now.

There are 6 options:

1. The final answer is in, we are done, return to SyncRes::doResolve and SyncRes::beginResolve
2. The nameserver we queried tells us the domain we asked for authoritatively does not exist. In case of the root-servers, this happens when we query for *''www.powerdns.kom.''* for example, there is no *''kom.''*. Return to SyncRes::beginResolve, we are done.
3. A lesser form - it tells us it is authoritative for the query we asked about, but there is no record matching our type. This happens when querying for the IPv6 address of a host which only has an IPv4 address. Return to SyncRes::beginResolve, we are done.
4. The nameserver passed us a CNAME to another domain, and we need to reroute. Go to SyncRes::doResolve for the new domain.
5. The nameserver did not know about the domain, but does know who does, a *referral*. Stay within doResolveAt and loop to these new nameservers.
6. The nameserver replied saying *no idea*. This is called a *lame delegation*. Stay within SyncRes::doResolveAt and try the other nameservers we have for this domain.

When not redirected using a CNAME, this function will loop until it has exhausted all nameservers and all their IP addresses. DNS is surprisingly resilient that there is often only a single non-broken nameserver left to answer queries, and we need to be prepared for that.

This is the whole DNS algorithm in PowerDNS. It contains a lot of tricky bits though, related to the caches and things like RPZ handling and DNSSEC validation.

24.7 QName Minimization

Since the 4.3 release, the recursor implements a relaxed form of QName Minimization. This is a method to enhance privacy and described in the (draft) RFC 7816. By asking the authoritative server not the full QName, but one more label than we already know it is authoritative for we do not leak which exact names are queried to servers higher up in the hierarchy.

The implementation uses a relaxed form of QName Minimization, following the recommendations found in the paper "A First Look at QNAME Minimization in the Domain Name System" by De Vries et all.

We originally started with using NS probes as the example algorithm in the RFC draft recommends.

We then quickly discovered that using NS probes were somewhat troublesome and after reading the mentioned paper we changed to QType A for probes, which worked better. We did not implemented the extra label prepend, not understanding why that would be needed (a more recent draft of the RFC came to the same conclusion).

Following the recommendations in the paper we also implemented larger steps when many labels are present. We use steps 1-1-1-3-3-...; we already have a limit on the number of outgoing queries induced by a client query. We do a final full QName query if we get an unexpected error. This happens when we encounter authoritative servers that are not fully compliant, there are still many servers like that. The recursor records with respect to this fallback scenario in the `qname-min-fallback-success` metric.

For forwarded queries, we do not use QName Minimization.

24.8 Some of the things we glossed over

Whenever a packet is sent to a remote nameserver, the response time is stored in the SyncRes::s_nsSpeeds map, using an exponentially weighted moving average. This EWMA averages out different response times, and also makes them decrease over time. This means that a nameserver that hasn't been queried recently gradually becomes **faster** in the eyes of PowerDNS, giving it a chance again.

A timeout is accounted as a 1s response time, which should take that server out of the running for a while.

Furthermore, queries are throttled. This means that each query to a nameserver that has failed is accounted in the `s_throttle` object. Before performing a new query, the query and the nameserver are looked up via shouldThrottle. If so, the query is assumed to have failed without even being performed. This saves a lot of network traffic and makes PowerDNS quick to respond to lame servers.

It also offers a modicum of protection against birthday attack powered spoofing attempts, as PowerDNS will not inundate a broken server with queries.

The negative query cache we mentioned earlier caches the cases 2 and 3 in the enumeration above. This data needs to be stored separately, as it represents **non-data**. Each negcache query entry is the name of the SOA record that was presented with the evidence of non-existence. This SOA record is then retrieved from the regular cache, but with the TTL that originally came with the NXDOMAIN (case 2) or NXRRSET (case 3).

24.9 The Recursor Cache

As mentioned before, the cache stores partial packets. It also stores not the **Time To Live** of records, but in fact the **Time To Die**. If the cache contains data, but it is expired, that data should not be deemed present. This bit of PowerDNS has proven tricky, leading to deadlocks in the past.

There are some other very tricky things to deal with. For example, through a process called **more details**, a domain might have more nameservers than listed in its parent zone. So, there might only be two nameservers for `powerdns.com.` in the **''com.''** zone, but the **''powerdns.com''** zone might list more.

This means that the cache should not, when talking to the **''com.''** servers later on, overwrite these four nameservers with only the two copies the **''com.''** servers pass us.

However, in other cases (like for example for SOA and CNAME records), new data should overwrite old data.

Note that PowerDNS deviates from RFC 2181 (section 5.4.1) in this respect.

Starting with version 4.7.0, there is a mechanism to save the parent NS set if it contains *more* names than the child NS set. This allows falling back to the saved parent NS set on resolution errors using the child specified NS set. As experience shows, this configuration error is encountered in the wild often enough to warrant this workaround. See *save-parent-ns-set*.

24.10 Serve Stale

Starting with version 4.8.0, the Recursor implements `Serve Stale` (**RFC 8767**). This is a mechanism that allows records in the record cache that are expired but that cannot be refreshed (due to network or authoritative server issues) to be served anyway.

The *serve-stale-extensions* determines how many times the records lifetime can be extended. Each extension of the lifetime of a record lasts 30s. A value of 1440 means the maximum extra life time is 30 * 1440 seconds which is 12 hours. If the original TTL of a record was less than 30s, the original TTLs will be used as extension period.

On each extension an asynchronous task to resolve the name will be created. If that task succeeds, the record will not be served stale anymore, as an up-to-date value is now available.

If *serve-stale-extensions* is not zero expired records will be kept in the record cache until the number of records becomes too large. Cache eviction will then be done on a least-recently-used basis.

When dumping the cache using `rec_control dump-cache` the `ss` value shows the serve stale extension count. A value of 0 means the record is not being served stale, while a positive value shows the number of times the serve stale period has been extended.

24.11 Some small things

The server-side part of PowerDNS (`pdns_recursor.cc`), which listens to queries by end-users, is fully IPv6 capable using the ComboAddress class. This class is in fact a union of a `struct sockaddr_in` and a `struct sockaddr_in6`. As long as the `sin_family` (or `sin6_family`) and `sin_port` members are in the same place, this works just fine, allowing us to pass a ComboAddress*, cast to a `sockaddr*` to the socket functions. For convenience, the ComboAddress also offers a length() method which can be used to indicate the length - either sizeof(sockaddr_in) or sizeof(sockaddr_in6).

Access to the recursor is governed through the NetmaskGroup class, which internally contains Netmask, which in turn contain a ComboAddress.

CHAPTER

TWENTYFIVE

STRUCTURED LOGGING DICTIONARY

This page describes the common entries of the Structured Logging component. Currently *structured-logging-backend* can have these values:

- The `default` text based backend
- The `systemd-journal` backend
- The `json` backend (added in version 5.1.0).

25.1 The `default` backend

The default backend uses a text representation of the key-value pairs. A line is constructed by appending all key-value pairs as `key="value"`, separated by spaces. The output is written by passing the resulting text line to the standard error stream and also to `syslog` if *disable-syslog* is false. Depending on the value of *log-timestamp* a timestamp is prepended to the log line.

An example line (including prepended timestamp) looks like this:

```
Oct 18 08:45:21 msg="Raised soft limit on number of filedescriptors to match max-
↪mthreads and threads settings" subsystem="config" level="0" prio="Warning" tid="0
↪" ts="1697611521.119" limit="6469"
```

- Key names are not quoted.
- Values are quoted with double quotes.
- If a value contains a double quote, it is escaped with a backslash.
- Backslashes in the value are escaped by prepending a backslash.

The following keys are always present:

Key	Type	Example	Remarks
`msg`	string	"Launching distributor threads"	Value is the same for all instances of this log entry, together with `subsystem` it uniquely identifies the log message.
`subsystem`	string	"incoming"	Uniquely identifies the log entry together with the value of `msg`.
`level`	number	"10"	The detail level of the log entry, do not confuse with *loglevel*. Not actively used currently.
`prio`	enum	"Notice"	One of `Alert=1`, `Critical=2`, `Error=3`, `Warning=4`, `Notice=5`, `Info=6`, `Debug=7`. A log entry will only produced if its `prio` is equal or lower than *loglevel*.
`tid`	number	"12"	The Posix worker thread id that produced the log entry. If not produced by a worker thread, the value is zero.
`ts`	number	"1697614303.039"	Number of seconds since the Unix epoch, including fractional part.

A log entry can also have zero or more additional key-value pairs. Common keys are:

Key	Type	Example	Remarks
error	string	"No such file or directory"	An error cause.
address	ip address:port	"[::]:5301"	An IP: port combination.
addresses	list of subnets	"127.0.0.0/8 ::ffff:0:0/96"	A list of subnets, space separated.
path	filesystem path	"tmp/api-dir/apizones"	
proto	string	"udp"	
qname	DNS name	"example.com"	
qtype	DNS Query Type	"AAAA"	Text representation of DNS query type.
rcode	DNS Response Code	"3"	Numeric DNS response code
mtid	Number	"234"	The id of the MThread that produced the log entry.

25.2 The systemd-journal backend

The systemd-journal structured logging backend uses mostly the same keys and values as the default backend, with the exceptions:

- keys are capitalized as required for systemd-journal.
- msg is translated to MESSAGE.
- prio is translated to PRIORITY.
- ts is translated to TIMESTAMP.
- If the original key is in a list of keys special to systemd-journal, it is capitalized and prepended by PDNS_. The list of special keys is: message, message_id, priority, code_file, code_line, code_func, errno, invocation_id, user_invocation_id, syslog_facility, syslog_identifier, syslog_pid, syslog_timestamp, syslog_raw, documentation, tid, unit, user_unit, object_pid.

To use this logging backend, add the --structured-logging-backend=systemd-journal to the command line in the systemd unit file. Note that adding it to the recursor configuration file does not work as expected, as this file is processed after the logging has been set up.

To query the log, use a command similar to:

```
# journalctl -r -n 1 -o json-pretty -u pdns-recursor.service
```

25.3 The json backend

The json structured logging backend has been added in version 5.1.0 and uses the same keys and values as the default backend. An example of a a log object:

```
{"level": "0", "limit": "10765", "msg": "Raised soft limit on number of␣
↪filedescriptors to match max-mthreads and threads settings", "priority": "4",
↪"subsystem": "config", "tid": "0", "ts": "1709285994.851"}
```

All values are represented as strings.

The JSON log objects are written to the standard error stream.

CHAPTER

TWENTYSIX

CONVERSION OF OLD-STYLE SETTINGS TO YAML FORMAT

Running the command

```
rec_control show-yaml
```

will show the conversion of existing old-style settings into the new YAML format. The existing settings will be read from the default old-style settings file `recursor.conf` in the configuration directory. It is also possible to show the conversion of a specific old-style settings file by running

```
rec_control show-yaml path/to/recursor.conf
```

`rec_control show-yaml` will also show the conversions of any included `.conf` file (if *include-dir* is set) and other associated settings file, like *forward-zones-file*.

26.1 Example

Consider the old style configuration file `recursor.conf`:

```
dnssec=validate
include-dir=recursor.d
forward-zones = example=127.0.0.1:1024
forward-zones-file=fwzones.txt
local-address=128.66.23.4:5353, [::1]:53
```

With the contents of `recursor.d/01.conf`:

```
forward-zones += example.com=128.66.9.99
forward-zones += example.net=128.66.9.100;128.66.9.101
local-address = 0.0.0.0
```

And `fwzones.txt`:

```
^+example.org=127.0.0.1:1024
```

To show the conversion result, run:

```
cd example
rec_control show-yaml recursor.conf
```

Produces the following conversion report:

```
# Start of converted recursor.yml based on recursor.conf
dnssec:
  validation: validate
incoming:
  listen:
  - 128.66.23.4:5353
```

(continues on next page)

(continued from previous page)

```
  - '[::1]:53'
recursor:
  forward_zones:
  - zone: example
    recurse: false
    forwarders:
    - 127.0.0.1:1024
  forward_zones_file: fwzones.txt
  include_dir: recursor.d
# Validation result: OK
# End of converted recursor.conf
#
# Found 1 .conf file in recursor.d
# Converted include-dir recursor.d/01.conf to YAML format:
incoming:
  listen: !override
  - 0.0.0.0
recursor:
  forward_zones:
  - zone: example.com
    recurse: false
    forwarders:
    - 128.66.9.99
  - zone: example.net
    recurse: false
    forwarders:
    - 128.66.9.100
    - 128.66.9.101
# Validation result: OK
# End of converted recursor.d/01.conf
#
# Converted fwzones.txt to YAML format for recursor.forward_zones_file:
- zone: example.org
  forwarders:
  - 127.0.0.1:1024
  recurse: true
  notify_allowed: true
# Validation result: OK
# End of converted fwzones.txt
#
```

Note the `!override` tag for `incoming.listen` (corresponding to the `=` in `recursor.d/01.conf`. The `recursor.forward_zones` settings is extending the setting in the main `recursord.yml` file, as `recursor.d/01.conf` uses a `+=` for the `forward-zones` settings. Consult *PowerDNS Recursor New Style (YAML) Settings* for details on how settings spread over multiple files are merged.

The contents of the report can be used to produce YAML settings equivalent to the old-style settings. This is a manual step and consists of copy-pasting the sections of the conversion report to individual files.

Any converted settings filename in the **include** directory should end with `.yml`. The names of associated files (like a `recursor.forward_zones_file`) should also end in `.yml`, but should *not* be put into the **include** directory, as they do not contain full configuration YAML clauses but YAML sequences of a specific type. The associated files *can* be put in the **config** directory, the directory that is searched for a `recursor.conf` or `recursor.yml` file.

26.2 API Managed Files

The format of API managed files was also changed to use YAML format. Specifically, the list of API managed zones is now a single file containing a sequence of `auth_zones` and a sequence of `forward_zones` instead

of a settings file per zone. The list of ACLs is a YAML sequence of subnets or IP addresses.

When using YAML settings *recursor.include_dir* and *webservice.api_dir* must have a different value. When YAML settings are active the **`Recursor`** will read old-style API managed files from the include directory on startup, convert them to the new format and write them into the API config directory. After conversion, it will inactivate the old-style API managed config files in the include directory by renaming them.

CHAPTER

TWENTYSEVEN

POWERDNS/DNSDIST LICENSE

We remind PowerDNS and dnsdist users that under the terms of the GNU General Public License, PowerDNS and dnsdist come with ABSOLUTELY NO WARRANTY.

```
                    GNU GENERAL PUBLIC LICENSE
                       Version 2, June 1991

 Copyright (C) 1989, 1991 Free Software Foundation, Inc.,
 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
 Everyone is permitted to copy and distribute verbatim copies
 of this license document, but changing it is not allowed.

                            Preamble

  The licenses for most software are designed to take away your
freedom to share and change it.  By contrast, the GNU General Public
License is intended to guarantee your freedom to share and change free
software--to make sure the software is free for all its users.  This
General Public License applies to most of the Free Software
Foundation's software and to any other program whose authors commit to
using it.  (Some other Free Software Foundation software is covered by
the GNU Lesser General Public License instead.)  You can apply it to
your programs, too.

  When we speak of free software, we are referring to freedom, not
price.  Our General Public Licenses are designed to make sure that you
have the freedom to distribute copies of free software (and charge for
this service if you wish), that you receive source code or can get it
if you want it, that you can change the software or use pieces of it
in new free programs; and that you know you can do these things.

  To protect your rights, we need to make restrictions that forbid
anyone to deny you these rights or to ask you to surrender the rights.
These restrictions translate to certain responsibilities for you if you
distribute copies of the software, or if you modify it.

  For example, if you distribute copies of such a program, whether
gratis or for a fee, you must give the recipients all the rights that
you have.  You must make sure that they, too, receive or can get the
source code.  And you must show them these terms so they know their
rights.

  We protect your rights with two steps: (1) copyright the software, and
(2) offer you this license which gives you legal permission to copy,
distribute and/or modify the software.

  Also, for each author's protection and ours, we want to make certain
that everyone understands that there is no warranty for this free
software.  If the software is modified by someone else and passed on, we
```

(continues on next page)

(continued from previous page)

```
want its recipients to know that what they have is not the original, so
that any problems introduced by others will not reflect on the original
authors' reputations.

  Finally, any free program is threatened constantly by software
patents.  We wish to avoid the danger that redistributors of a free
program will individually obtain patent licenses, in effect making the
program proprietary.  To prevent this, we have made it clear that any
patent must be licensed for everyone's free use or not licensed at all.

  The precise terms and conditions for copying, distribution and
modification follow.

                    GNU GENERAL PUBLIC LICENSE
   TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

  0. This License applies to any program or other work which contains
a notice placed by the copyright holder saying it may be distributed
under the terms of this General Public License.  The "Program", below,
refers to any such program or work, and a "work based on the Program"
means either the Program or any derivative work under copyright law:
that is to say, a work containing the Program or a portion of it,
either verbatim or with modifications and/or translated into another
language.  (Hereinafter, translation is included without limitation in
the term "modification".)  Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not
covered by this License; they are outside its scope.  The act of
running the Program is not restricted, and the output from the Program
is covered only if its contents constitute a work based on the
Program (independent of having been made by running the Program).
Whether that is true depends on what the Program does.

  1. You may copy and distribute verbatim copies of the Program's
source code as you receive it, in any medium, provided that you
conspicuously and appropriately publish on each copy an appropriate
copyright notice and disclaimer of warranty; keep intact all the
notices that refer to this License and to the absence of any warranty;
and give any other recipients of the Program a copy of this License
along with the Program.

You may charge a fee for the physical act of transferring a copy, and
you may at your option offer warranty protection in exchange for a fee.

  2. You may modify your copy or copies of the Program or any portion
of it, thus forming a work based on the Program, and copy and
distribute such modifications or work under the terms of Section 1
above, provided that you also meet all of these conditions:

    a) You must cause the modified files to carry prominent notices
    stating that you changed the files and the date of any change.

    b) You must cause any work that you distribute or publish, that in
    whole or in part contains or is derived from the Program or any
    part thereof, to be licensed as a whole at no charge to all third
    parties under the terms of this License.

    c) If the modified program normally reads commands interactively
    when run, you must cause it, when started running for such
    interactive use in the most ordinary way, to print or display an
    announcement including an appropriate copyright notice and a
```

(continues on next page)

(continued from previous page)

```
    notice that there is no warranty (or else, saying that you provide
    a warranty) and that users may redistribute the program under
    these conditions, and telling the user how to view a copy of this
    License.  (Exception: if the Program itself is interactive but
    does not normally print such an announcement, your work based on
    the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole.  If
identifiable sections of that work are not derived from the Program,
and can be reasonably considered independent and separate works in
themselves, then this License, and its terms, do not apply to those
sections when you distribute them as separate works.  But when you
distribute the same sections as part of a whole which is a work based
on the Program, the distribution of the whole must be on the terms of
this License, whose permissions for other licensees extend to the
entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest
your rights to work written entirely by you; rather, the intent is to
exercise the right to control the distribution of derivative or
collective works based on the Program.

In addition, mere aggregation of another work not based on the Program
with the Program (or with a work based on the Program) on a volume of
a storage or distribution medium does not bring the other work under
the scope of this License.

  3. You may copy and distribute the Program (or a work based on it,
under Section 2) in object code or executable form under the terms of
Sections 1 and 2 above provided that you also do one of the following:

    a) Accompany it with the complete corresponding machine-readable
    source code, which must be distributed under the terms of Sections
    1 and 2 above on a medium customarily used for software interchange; or,

    b) Accompany it with a written offer, valid for at least three
    years, to give any third party, for a charge no more than your
    cost of physically performing source distribution, a complete
    machine-readable copy of the corresponding source code, to be
    distributed under the terms of Sections 1 and 2 above on a medium
    customarily used for software interchange; or,

    c) Accompany it with the information you received as to the offer
    to distribute corresponding source code.  (This alternative is
    allowed only for noncommercial distribution and only if you
    received the program in object code or executable form with such
    an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for
making modifications to it.  For an executable work, complete source
code means all the source code for all modules it contains, plus any
associated interface definition files, plus the scripts used to
control compilation and installation of the executable.  However, as a
special exception, the source code distributed need not include
anything that is normally distributed (in either source or binary
form) with the major components (compiler, kernel, and so on) of the
operating system on which the executable runs, unless that component
itself accompanies the executable.

If distribution of executable or object code is made by offering
access to copy from a designated place, then offering equivalent
```

(continues on next page)

(continued from previous page)

```
access to copy the source code from the same place counts as
distribution of the source code, even though third parties are not
compelled to copy the source along with the object code.

  4. You may not copy, modify, sublicense, or distribute the Program
except as expressly provided under this License.  Any attempt
otherwise to copy, modify, sublicense or distribute the Program is
void, and will automatically terminate your rights under this License.
However, parties who have received copies, or rights, from you under
this License will not have their licenses terminated so long as such
parties remain in full compliance.

  5. You are not required to accept this License, since you have not
signed it.  However, nothing else grants you permission to modify or
distribute the Program or its derivative works.  These actions are
prohibited by law if you do not accept this License.  Therefore, by
modifying or distributing the Program (or any work based on the
Program), you indicate your acceptance of this License to do so, and
all its terms and conditions for copying, distributing or modifying
the Program or works based on it.

  6. Each time you redistribute the Program (or any work based on the
Program), the recipient automatically receives a license from the
original licensor to copy, distribute or modify the Program subject to
these terms and conditions.  You may not impose any further
restrictions on the recipients' exercise of the rights granted herein.
You are not responsible for enforcing compliance by third parties to
this License.

  7. If, as a consequence of a court judgment or allegation of patent
infringement or for any other reason (not limited to patent issues),
conditions are imposed on you (whether by court order, agreement or
otherwise) that contradict the conditions of this License, they do not
excuse you from the conditions of this License.  If you cannot
distribute so as to satisfy simultaneously your obligations under this
License and any other pertinent obligations, then as a consequence you
may not distribute the Program at all.  For example, if a patent
license would not permit royalty-free redistribution of the Program by
all those who receive copies directly or indirectly through you, then
the only way you could satisfy both it and this License would be to
refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under
any particular circumstance, the balance of the section is intended to
apply and the section as a whole is intended to apply in other
circumstances.

It is not the purpose of this section to induce you to infringe any
patents or other property right claims or to contest validity of any
such claims; this section has the sole purpose of protecting the
integrity of the free software distribution system, which is
implemented by public license practices.  Many people have made
generous contributions to the wide range of software distributed
through that system in reliance on consistent application of that
system; it is up to the author/donor to decide if he or she is willing
to distribute software through any other system and a licensee cannot
impose that choice.

This section is intended to make thoroughly clear what is believed to
be a consequence of the rest of this License.
```

(continues on next page)

(continued from previous page)

```
  8. If the distribution and/or use of the Program is restricted in
certain countries either by patents or by copyrighted interfaces, the
original copyright holder who places the Program under this License
may add an explicit geographical distribution limitation excluding
those countries, so that distribution is permitted only in or among
countries not thus excluded.  In such case, this License incorporates
the limitation as if written in the body of this License.

  9. The Free Software Foundation may publish revised and/or new versions
of the General Public License from time to time.  Such new versions will
be similar in spirit to the present version, but may differ in detail to
address new problems or concerns.

Each version is given a distinguishing version number.  If the Program
specifies a version number of this License which applies to it and "any
later version", you have the option of following the terms and conditions
either of that version or of any later version published by the Free
Software Foundation.  If the Program does not specify a version number of
this License, you may choose any version ever published by the Free Software
Foundation.

  10. If you wish to incorporate parts of the Program into other free
programs whose distribution conditions are different, write to the author
to ask for permission.  For software which is copyrighted by the Free
Software Foundation, write to the Free Software Foundation; we sometimes
make exceptions for this.  Our decision will be guided by the two goals
of preserving the free status of all derivatives of our free software and
of promoting the sharing and reuse of software generally.

                            NO WARRANTY

  11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY
FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW.  EXCEPT WHEN
OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES
PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED
OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF
MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.  THE ENTIRE RISK AS
TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU.  SHOULD THE
PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING,
REPAIR OR CORRECTION.

  12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING
WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR
REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES,
INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING
OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED
TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY
YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER
PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE
POSSIBILITY OF SUCH DAMAGES.

                      END OF TERMS AND CONDITIONS

            How to Apply These Terms to Your New Programs

  If you develop a new program, and you want it to be of the greatest
possible use to the public, the best way to achieve this is to make it
free software which everyone can redistribute and change under these terms.

  To do so, attach the following notices to the program.  It is safest
to attach them to the start of each source file to most effectively
```

(continues on next page)

(continued from previous page)

```
convey the exclusion of warranty; and each file should have at least
the "copyright" line and a pointer to where the full notice is found.

    <one line to give the program's name and a brief idea of what it does.>
    Copyright (C) <year>  <name of author>

    This program is free software; you can redistribute it and/or modify
    it under the terms of the GNU General Public License as published by
    the Free Software Foundation; either version 2 of the License, or
    (at your option) any later version.

    This program is distributed in the hope that it will be useful,
    but WITHOUT ANY WARRANTY; without even the implied warranty of
    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
    GNU General Public License for more details.

    You should have received a copy of the GNU General Public License along
    with this program; if not, write to the Free Software Foundation, Inc.,
    51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA.

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this
when it starts in an interactive mode:

    Gnomovision version 69, Copyright (C) year name of author
    Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
    This is free software, and you are welcome to redistribute it
    under certain conditions; type `show c' for details.

The hypothetical commands `show w' and `show c' should show the appropriate
parts of the General Public License.  Of course, the commands you use may
be called something other than `show w' and `show c'; they could even be
mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your
school, if any, to sign a "copyright disclaimer" for the program, if
necessary.  Here is a sample; alter the names:

  Yoyodyne, Inc., hereby disclaims all copyright interest in the program
  `Gnomovision' (which makes passes at compilers) written by James Hacker.

  <signature of Ty Coon>, 1 April 1989
  Ty Coon, President of Vice

This General Public License does not permit incorporating your program into
proprietary programs.  If your program is a subroutine library, you may
consider it more useful to permit linking proprietary applications with the
library.  If this is what you want to do, use the GNU Lesser General
Public License instead of this License.
```

HTTP ROUTING TABLE

/api

/jsonstat

/metrics

INDEX

G

I

L

M

N

O

P

Q

R

S

T

U

V

W

Z