ZoneControl

Administrator Guide

Apr 23, 2024

Release 1.6.1

CONTENTS

1 Reference Architecture **3**
1.1 Ports and communications . . . 3

2 Installation **5**
2.1 Post-installation . . . 5

3 Configuration **7**
3.1 Database configuration . . . 7
3.2 Secret configuration . . . 8
3.3 HTTP Service configuration . . . 8
3.4 Scheduled Changes Deployment . . . 9
3.5 Zones List Caching . . . 9
3.6 Zones Overview DNSSEC Retrieval . . . 10
3.7 Restriction of Zone Kinds . . . 10
3.8 Advanced DNSSEC . . . 10
3.9 Default NSEC3PARAM . . . 11
3.10 Automatic PTR Creation . . . 11
3.11 Zone Actions . . . 11
3.12 Prometheus Metrics . . . 11
3.13 Catalog Zone Support . . . 11
3.14 Audit Logging to Standard Out or File . . . 12

4 Fronting with nginx **13**
4.1 ZoneControl . . . 13
4.2 PowerDNS Authoritative Server . . . 14

5 Managing ZoneControl **15**
5.1 Controlling the ZoneControl service . . . 15
5.2 Managing ZoneControl . . . 15

6 The ZoneControl API **17**
6.1 ZoneControl Endpoints . . . 17
6.2 ZoneControl Endpoints in the PowerDNS API . . . 18
6.3 API Tokens . . . 20

7 Administrative Web Interface **21**
7.1 Users and Groups . . . 21
7.2 LDAP Authentication . . . 23
7.3 Administrative Permissions for Staff users . . . 23
7.4 Scheduled Changes . . . 25
7.5 Roles . . . 27
7.6 Servers . . . 29

8 Changelog **31**
8.1 1.6.1 (April 23, 2024) . . . 31

8.2 1.6.0 (March 28, 2024) . . . 31
8.3 1.5.1 (November 1, 2023) . . . 32
8.4 1.5.0 (October 2, 2023) . . . 32
8.5 1.4.2 (May 19, 2023) . . . 33
8.6 1.4.1 (September 2, 2022) . . . 34
8.7 1.4.0 (March 10, 2022) . . . 34
8.8 1.3.4 (October 14, 2021) . . . 34
8.9 1.3.3 (July 09, 2021) . . . 35
8.10 1.3.2 (March 25, 2021) . . . 35
8.11 1.3.1 (January 21, 2021) . . . 35
8.12 1.3.0 (November 26, 2020) . . . 36
8.13 1.2.0 (June 15, 2020) . . . 36
8.14 1.1.1 (July 10, 2019) . . . 37
8.15 1.1.0 (June 24, 2019) . . . 37
8.16 1.0.1 (July 17, 2018) . . . 38
8.17 1.0.0 (September 20, 2017) . . . 39

HTTP Routing Table **41**

This guide assumes general knowledge of Linux systems, experience with the use of the command-line, editors and knowledge of the specifics of the target environment (i.e. whether or not `sudo` is used, if local system users or LDAP is used, etc.).

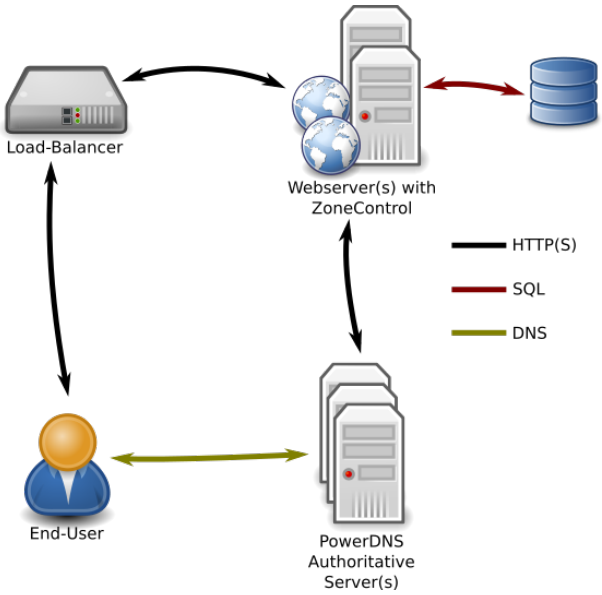
This guide does not discuss setting up the PowerDNS Authoritative Server with a database backend.

CHAPTER

ONE

REFERENCE ARCHITECTURE

The reference architecture for ZoneControl consists of one or more webservers running ZoneControl, fronted by a Load-Balancer. These ZoneControl-instances use one shared database for their configuration and communicate with the API of one or more PowerDNS Authoritative Servers.



1.1 Ports and communications

ZoneControl runs through its built-in gunicorn webserver and it should be put behind another webserver like nginx. See *Fronting with nginx*

CHAPTER

TWO

INSTALLATION

ZoneControl is a Python Django application and is typically installed through our official Ansible packages.

PowerDNS will provide an Ansible role to install ZoneControl and set up an initial user. This role comes with a `README` file that describes all variables for the role.

2.1 Post-installation

After the initial installation, ZoneControl needs to be configured. See the *Configuration* section on how to configure.

After configuration, the database needs to be populated, run:

```
zonecontrol-manage migrate
```

Now create a superuser (if required):

```
zonecontrol-manage createsuperuser
```

CHAPTER

THREE

CONFIGURATION

This chapter describes how to configure the ZoneControl service.

After changing settings, do not forget to restart the service.

3.1 Database configuration

ZoneControl, by default, uses an SQLite database located at `/var/lib/zonecontrol/zonecontrol.db`. Should another database be needed, it needs to be configured in `/etc/zonecontrol/settings.py`.

Django offers connectors for many types of database servers, described in its online documentation.

To enable support for databases other than SQLite, the package with the database driver must be installed. These packages are named "zonecontrol-<database>". The following databases are supported:

Database	Package-suffix	Engine in `DATABASES`
MySQL	`mysql`	django.db.backends.mysql

Now configure the database in `/etc/zonecontrol/settings.py`, set the ENGINE to the Engine from the table. The rest of the settings are self-explanatory:

```
DATABASES = {
    'default': {
        'ENGINE': '$ENGINE',
        'NAME': 'mydatabase',
        'USER': 'mydatabaseuser',
        'PASSWORD': 'mypassword',
        'HOST': '127.0.0.1',
        'PORT': '',
    }
}
```

After setting up a new database, it need to be populated with database tables. Run the following command as the `zonecontrol` user:

```
zonecontrol-manage migrate
```

3.2 Secret configuration

The Django web framework relies on a "secret key" for encrypting cookies, password reset tokens and other uses (see the online documentation for more).

This key is generated upon package installation in `/var/lib/zonecontrol/secret_key`.

Should this key not be set or if it is too short, ZoneControl will not start. Generate a new key and add it to the configuration file:

```
/usr/share/zonecontrol/bin/python3 -c 'import random; import string; print("".
↪join([random.SystemRandom().choice("{}{}{}".format(string.ascii_letters, string.
↪digits, string.punctuation)) for i in range(50)]))' > /var/lib/zonecontrol/secret_
↪key
```

3.3 HTTP Service configuration

The Gunicorn HTTP server is configured in the `/etc/zonecontrol/gunicorn_config.py` file. By default, it will listen for HTTP connections on `127.0.0.1:8083`.

A sane configuration is below:

```
# Config file for the gunicorn HTTP server
# See: http://docs.gunicorn.org/en/stable/settings.html
import multiprocessing
import os

# This only has an effect if setproctitle is installed.
proc_name = 'zonecontrol'

raw_env = ['DJANGO_SETTINGS_MODULE=project.settings', 'GUNICORN=1']
bind = ['127.0.0.1:8053']
workers = multiprocessing.cpu_count() * 2
threads = 8 # per worker
timeout = 30 # sec

chdir = '/usr/share/zonecontrol'

accesslog = '/var/log/zonecontrol/access.log'
access_log_format = '%(h)s (%({X-Forwarded-For}i)s) %(l)s %(u)s %(t)s "%(r)s" %(s)s
↪%(b)s "%(f)s" "%(a)s"'
errorlog = '/var/log/zonecontrol/error.log'
loglevel = 'info'
# syslog = True
```

For more settings, please consult the online documentation.

3.4 Scheduled Changes Deployment

In order to deploy scheduled changes, ZoneControl comes with a small service called `zonecontrol-scheduler.service`. This service reads the *Scheduled Changes* from the database and applies them when needed.

3.5 Zones List Caching

New in version 1.2.0.

ZoneControl can cache the zones-list in redis to speed up calls from the frontend.

Note: Only the list of zones is cached, not the zone contents.

When changes are made, ZoneControl also stores these mutations in the cache after they have been accepted by PowerDNS. A seperate service called `zonecontrol-cache` loads the list of zones into redis and prunes the mutations at regular intervals. The zones-list is stored wih an expiry time so the cache is either empty or recent.

ZoneControl will always attempt to load the zones-list from the cache, if it cannot find the list *or* caching is explicitly disabled, it will fall back to loading the list from PowerDNS.

3.5.1 Enabling Caching

On installations on CentOS, install the `zonecontrol-cache` package, this will also install `redis`. For Debian installations, install the `redis` package.

Then start and enable both redis and the zonecontrol-cache services:

```
systemctl enable redis.service
systemctl start redis.service
systemctl enable zonecontrol-cache.service
systemctl start zonecontrol-cache.service
```

3.5.2 Cache configuration

The caching behaviour is controlled by several options in `/etc/zonecontrol/settings.py`.

`ZONECONTROL_CACHE_ZONES`
: Whether or not to cache zones lists (for `zonecontrol-cache`) and to attempt to retrieve them (for ZoneControl), default `True`.

`ZONECONTROL_CACHE_ZONES_LOCATION`
: Connection URL for Redis. By default database 1 on localhost without authentication and encryption. is used. The redis-py documentation describes this URI in detail. Default is `"redis://127.0.0.1:6379/1"`.

`ZONECONTROL_CACHE_ZONES_PASSWORD`
: Password for Redis. Default is `None` (ie: no password). If Redis ACLs are used, the username can be supplied as part of the URI described above.

`ZONECONTROL_CACHE_ZONES_INTERVAL`
: After `zonecontrol-cache` has added a new list to the cache, this setting controls how long it will wait before starting a new retrieval. Default is `60`.

`ZONECONTROL_CACHE_ZONES_EXPIRE_TTL`
: How long a zones-list is cached in redis before automatically removed, in seconds. Default is `300`.

After changing the settings, restart `zonecontrol-cache.service`.

3.6 Zones Overview DNSSEC Retrieval

New in version 1.2.0.

By default, the PowerDNS API includes the `dnssec` field when the list of zones is retrieved. Generating this field takes 5 queries to the database per zone in the list. When the database is slow, has high latency or the list of zones is huge, this can severely impact the performance of the zones overview in the ZoneControl frontend.

PowerDNS Authoritative Server 4.2.1 introduced the possiblity to request the zones-list *without* dnssec information. ZoneControl takea advantage of this API feature by default. To have ZoneControl request DNSSEC information from PowerDNS, set `ZONECONTROL_ZONE_OVERVIEW_DNSSEC` to `True` in `/etc/zonecontrol/settings.py`.

The downside of this feature is the loss of the "DNSSEC" column on the overview page of ZoneControl.

3.7 Restriction of Zone Kinds

New in version 1.1.0.

Operators might want to limit the kind of zones that can be created through ZoneControl. This can be done using the `ZONECONTROL_ALLOW_ZONE_KINDS` list in `/etc/zonecontrol/settings.py`.

By default, all kinds are allowed. To only allow Native zones:

```
from project.zonecontrol import ZONE_KIND_NATIVE
ZONECONTROL_ALLOW_ZONE_ACTIONS = [ZONE_KIND_NATIVE]
```

Note: while in the UI for the user the Master & Slave kinds have been renamed to Primary & Secondary, this is not the case for the config settings. One should still use `ZONE_KIND_MASTER` and `ZONE_KIND_SLAVE` for restricting zone kinds.

3.8 Advanced DNSSEC

New in version 1.1.0.

When a user has the *"Advanced DNSSEC"* permissions, they can manipulate single DNSSEC keys for zones. The ZoneControl configuration can limit the DNSSEC algorithms that are allowed to be created. There are two settings related to this.

`ZONECONTROL_ADVANCED_DNSSEC_ALGORITHMS`
: This is the list of algorithms that is shown in the selection box and that are allowed when keys are created through the ZoneControl API. By default, these are `[DNSSEC_ALGO_ECDSA256, DNSSEC_ALGO_ECDSA384, DNSSEC_ALGO_ED25519]`.

`ZONECONTROL_ADVANCED_DNSSEC_DEFAULT_ALGORITHM`
: This is the algorithm that is set in the Advanced DNSSEC views' selection box by default. This algorithm **must** be in the `ZONECONTROL_ADVANCED_DNSSEC_ALGORITHMS` list and is `DNSSEC_ALGO_ECDSA256` by default.

3.9 Default NSEC3PARAM

New in version 1.4.2.

The default `NSEC3PARAM` is set to `1 0 0 -` in `/etc/zonecontrol/settings.py` and can be changed there. These parameters are recommended by RFC9276.

3.10 Automatic PTR Creation

New in version 1.1.0.

ZoneControl can create PTR records for A and AAAA records that are created or modified. This can be disabled in the configuration by setting `ZONECONTROL_ALLOW_AUTOMATIC_PTR` to `False` in `/etc/zonecontrol/settings.py`.

3.11 Zone Actions

New in version 1.1.0.

Users with permissions to access zones can force actions like queuing NOTIFY messages to secondaries or (re-)retrieving a secondary zone. The allowed actions are specified in `ZONECONTROL_ALLOW_ZONE_ACTIONS`. By default this is set to `[ZONE_ACTION_NOTIFY, ZONE_ACTION_AXFR_RETRIEVE]`.

3.12 Prometheus Metrics

New in version 1.5.1.

Prometheus metrics will be exported on the `/metrics` URL path. This is the default path for Prometheus so just pointing your Prometheus configuration at the `host:port` target should be enough.

The metrics are disabled by default and can be enabled by setting `ZONECONTROL_METRICS_ENABLED` to `True` in `/etc/zonecontrol/settings.py`.

The metric for the number of zones (`zonecontrol_zones`) is only available when the caching of zones is enabled. See *Zones List Caching*.

3.13 Catalog Zone Support

New in version 1.6.0.

Note: Catalog zone support in ZoneControl will only work for PowerDNS Authoritative Server v4.9 and higher versions.

Support for Catalog Zones is disabled by default (see the note above) and can be enabled by setting `ZONECONTROL_CATALOG_ZONE_SUPPORT` to `True` in `/etc/zonecontrol/settings.py`.

When viewing a catalog zone there will be a new section where zones can be added and removed from the catalog.

3.14 Audit Logging to Standard Out or File

New in version 1.6.0.

Audit logging can be echoed to standard out or a file by setting `ZONECONTROL_AUDITLOG_STDOUT` to `True` in `/etc/zonecontrol/settings.py`. It is disabled by default.

These audit log lines are in JSON format but it is also possible to have plain text output; this is done by setting `ZONECONTROL_JSON_LOGGING` to `False` in the same settings file.

(The `ZONECONTROL_JSON_LOGGING` setting only applies to the audit logging at the moment, but it is the intention to do all ZoneControl logging in JSON format in some future ZoneControl major release at which point this setting will apply to all logging.)

CHAPTER

FOUR

FRONTING WITH NGINX

It is recommended to proxy traffic to both ZoneControl and the PowerDNS Authoritative server through the nginx webserver. This will catch possibly broken HTTP requests and allows for SSL termination.

4.1 ZoneControl

To proxy to a ZoneControl instance running on `127.0.0.1:8083`, the following configuration is a good starting point. Don't forget to set the `ssl_certificate` and `ssl_certificate_key` directives to your certificates and keys.

```
server {
  # redirect to https
  listen 80;
  listen [::]:80;
  server_name _;
  access_log /var/log/nginx/access.log;
  error_log /var/log/nginx/error.log error;
  location / {
    return 301 https://$http_host$request_uri;
  }
}

server {
  listen 443 ssl;
  listen [::]:443 ssl;
  server_name _;
  access_log /var/log/nginx/access.log;
  error_log /var/log/nginx/error.log error;
  ssl_certificate /etc/ssl/public/chain.pem;
  ssl_certificate_key /etc/ssl/private/privkey.pem;

  location / {
    proxy_pass http://127.0.0.1:8053/;

    proxy_set_header Host $host;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_set_header X-Forwarded-Proto $scheme;

    proxy_redirect off;
    proxy_buffering off;
  }
}
```

4.2 PowerDNS Authoritative Server

Fronting the PowerDNS Authoritative Server with nginx is very similar to fronting ZoneControl. The biggest difference is that HTTP is disabled and no headers are passed to the server. Don't forget to set the `ssl_certificate` and `ssl_certificate_key` directives to your certificates and keys.

```
server {
  listen 443 ssl;
  listen [::]:443 ssl;
  server_name _;
  access_log /var/log/nginx/access.log;
  error_log /var/log/nginx/error.log error;
  ssl_certificate /etc/ssl/public/chain.pem;
  ssl_certificate_key /etc/ssl/private/privkey.pem;

  location / {
    proxy_pass http://127.0.0.1:8081/;
    proxy_redirect off;
    proxy_buffering off;
  }
}
```

CHAPTER

FIVE

MANAGING ZONECONTROL

5.1 Controlling the ZoneControl service

ZoneControl comes with scripts and files relevant to the service management system for the Operating System it is installed on.

Note that ZoneControl does **not** support run-time reloading of its configuration.

5.1.1 systemd (RHEL/CentOS 7+, Debian 8+, Ubuntu 15.04+)

ZoneControl's service is called (unsurprisingly) `zonecontrol.service` and can be controlled in the normal fashion.

Starting:

```
systemctl start zonecontrol.service
```

Stopping:

```
systemctl stop zonecontrol.service
```

Restarting:

```
systemctl restart zonecontrol.service
```

5.2 Managing ZoneControl

ZoneControl comes with a tool called `zonecontrol-manage`, which is a wrapper around the Django manage.py script.

This script supports all the default commands from Django and some ZoneControl related ones.

5.2.1 `zonecontrol-manage addserver`

This command can be used to add PowerDNS Authoritative Servers to the configuration without logging in as an admin to the web-interface. It can be invoked as follows:

```
zonecontrol-manage addserver Server1 https://192.0.2.3:8083 MySecretAPIKey
```

This will add a new server called "Server1". This server is connected to on `https://192.0.2.3:8083` and "MySecretAPIKey" is used as the shared secret to connect.

This command also supports some other options, please refer to the output of

```
zonecontrol-manage addserver --help
```

5.2.2 zonecontrol-manage runscheduled

Run all *scheduled tasks*.

CHAPTER

SIX

THE ZONECONTROL API

The ZoneControl frontend communicates with the ZoneControl server application using the ZoneControl REST API. This API is the PowerDNS API with some additional endpoints for ZoneControl object like comments, and history.

Apart from that, ZoneControl

6.1 ZoneControl Endpoints

`GET /api/zonecontrol/info`

Returns information for the frontend regarding enabled features of the server and some permissions for the user. This is used by the frontend to show or hide certain UI elements.

Response JSON Object

- `username` (*str*) – Username of the logged in user.
- `is_admin` (*bool*) – User is a staff user.
- `can_add_zones` (*bool*) – User can create zones on at least one server.
- `can_add_zone_kinds` (*list(str)*) – The kinds of zones this user can create.
- `show_advanced_dnssec` (*bool*) – Whether this user had Advanced DNSSEC permissions.
- `can_add_ptr` (*bool*) – Whether automatic PTR creation is enabled.
- `can_queue_axfr_retrieve` (*bool*) – Whether or not the user can queue AXFR retrievals.
- `can_queue_notify` (*bool*) – Whether or not the user can queue NOTIFY messages.
- `dnssec_in_overview` (*bool*) – Whether or not DNSSEC is shown in the zones overview.
- `dnssec_advanced_default_algorithm` (*bool*) – The default algorithm in the Advanced DNSSEC view.
- `dnssec_advanced_algorithms` (*bool*) – The algorithms shown in the Advanced DNSSEC view.
- `nsec3param_default` (*bool*) – The default value for NSEC3.

`POST /api/zonecontrol/login`

This endpoint allows a login.

Form Parameters

- `username` – Username to log in.
- `password` – Password for the user.

Status Codes

- 200 OK – Login successful.
- 401 Unauthorized – Login failed.

On success, a JSON object is returned.

Response JSON Object

- `success` (*bool*) – Always true on a login.

6.2 ZoneControl Endpoints in the PowerDNS API

`GET /api/v1/servers/{server_id}/zones/{zone_id}/_names/{name}/_comments/{rrtype}`

Retrieve the comments for a certain record.

Parameters

- `server_id` (*str*) – Name of the server
- `zone_id` (*str*) – id (name) of the zone
- `name` (*str*) – Name of the RRSet
- `rrtype` (*str*) – RRType in textual format (e.g. A, AAAA, CNAME)

Returns a JSON object with one element `data` with contains an array of objects.

Response JSON Array of Objects

- `server` (*str*) – Name of the server this comment was made for.
- `zone` (*str*) – Name of the zone this comment was made for.
- `name` (*str*) – RRSet name.
- `type` (*str*) – RRType.
- `timestamp` (*str*) – Timestamp the comment was created.
- `user` (*str*) – Username of the comment's author.
- `comment` (*str*) – The actual comment.
- `from-server` (*bool*) – Whether or not the server was sent from the server, always true on a GET request.

`POST /api/v1/servers/{server_id}/zones/{zone_id}/_names/{name}/_comments/{rrtype}`

Create a comment for a certain record. The request body must be a single JSON string with the comment. All database fields (user, timestamp etc.) will be filled by the server.

Parameters

- `server_id` (*str*) – Name of the server
- `zone_id` (*str*) – id (name) of the zone
- `name` (*str*) – Name of the RRSet
- `rrtype` (*str*) – RRType in textual format (e.g. A, AAAA, CNAME)

Returns all comments as if *the GET for the same URL* was called.

`DELETE /api/v1/servers/{server_id}/zones/{zone_id}/_names/{name}/_comments/{rrtype}/_comment/{comment_id}`

Delete the comment with id `comment_id`

Parameters

- `server_id` – Name of the server

- `zone_id` – id (name) of the zone
- `name` – Name of the RRSet
- `comment_id` – id of the comment

`GET /api/v1/servers/{server_id}/zones/{zone_id}/_history`

Get all versions for the zone.

Parameters

- `server_id` – Name of the server
- `zone_id` – id (name) of the zone

Returns a JSON object with one element `data` with contains an array of zone versions.

Response JSON Array of Objects

- `id` (*int*) – id of the ZoneVersion
- `serial` (*int*) – SOA Serial of the zone
- `action` (*string*) – The action of this version of the zone, one of

  create
  Zone was created at this version

  delete
  Zone was deleted at this version

  settings
  Zone settings (e.g. kind) was changed

  metadata
  Zone Metadata was changed

  dnssec_enable
  DNSSEC was enabled

- `action_display` (*string*) – The action as a user-readable string, can be null.
- `can_be_restored` (*bool*) – Whether or not the zone can be rolled back to this version.
- `comment` (*string*) – Comment made when the version was created.
- `timestamp` (*str*) – Timestamp the version was created.
- `username` (*str*) – Name of the user who created the version, can be null.
- `external` (*bool*) – True if the version was not created through a change in ZoneControl.

`GET /api/v1/servers/{server_id}/zones/{zone_id}/_history/diff/{version_2}`

`GET /api/v1/servers/{server_id}/zones/{zone_id}/_history/diff/{version_1}/{version_2}`

Get a difference between zone versions.

Parameters

- `server_id` – Name of the server
- `zone_id` – id (name) of the zone
- `version_1` – Version to compare to
- `version_2` – Diff to retrieve

`POST /api/v1/servers/{server_id}/zones/{zone_id}/_history/restore/{version}`

Restore a Zone to the provided version.

Parameters

- `server_id` – Name of the server
- `zone_id` – id (name) of the zone
- `version` – Version to restore to

When restoration succeeds, a HTTP 200 response is returned with the following JSON:

Response JSON Object

- `success` (*bool*) – Whether or not the restoration succeeded.
- `new_version_recorded` (*bool*) – Whether or a new version was recorded.
- `new_version` (*int*) – id of the new version that was recorded.
- `new_serial` (*int*) – SOA serial of the zone after the restoration.
- `warning` (*str*) – A warning if a failure occurred.
- `warning_details` (*str*) – More warning information.

6.3 API Tokens

Users can be assigned API tokens from the administrative interface. This token can be used in HTTP requests to the ZoneControl server in the `X-Api-Key` header. This allows users to write their own applications that interact with the PowerDNS API while still keeping the Role permissions in place.

CHAPTER

SEVEN

ADMINISTRATIVE WEB INTERFACE

After logging in to ZoneControl, the administrative web interface can be reached by clicking on the username in the upper-right corner and clicking on "Admin Panel".

Here, *users*, *groups* and *roles* can be managed. This is also the portal where the audit-logs can be reviewed.

7.1 Users and Groups

In ZoneControl, three "levels" of users exists:

All users can log in to the zone-management interface (unless the user is marked disabled). Staff-users can open the administrative web interface and, depending on their permissions, perform administrative tasks there. Finally, the superusers can perform all administrative tasks without being granted specific permissions.

7.1.1 Users

A User is an entity that can log in to ZoneControl. When creating a User several attributes are requested:

- Username: The name with which the user will log in
- Password: The password for this User. Will be saved to the database in a hashed format

A User can be made part of zero or more *Groups*.

A User can additionally have zero or more *Roles*.

7.1.2 User Information and Configuration

Attached to users can be some extra information, several flags, and permissions.

The numbers below correspond with the numbers in Fig. 7.1.

Personal Info (1)

These fields are used to set some information on the user.

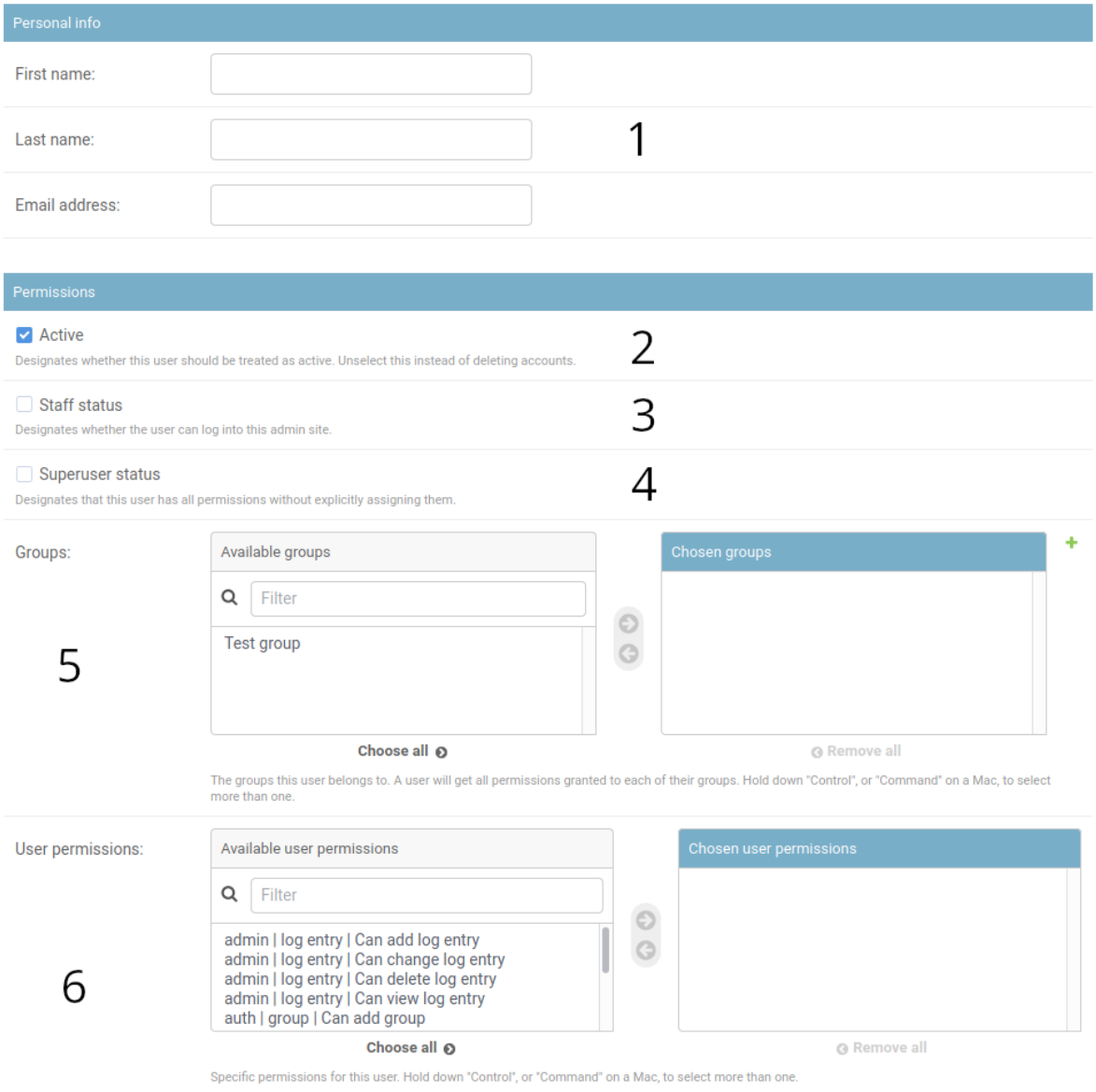


Fig. 7.1: Administrative webinterface user form

Active (2)

Users that are not active can not log in to ZoneControl. It is recommended to deactivate users when they should no longer be allowed to log in. This ensures that zone comments and zone versions are still attributed to this user. Removing the user will also remove this attribution.

Staff status (3)

A user with the staff-status is allowed to access the administrative web interface. They can perform administrative actions based on their permissions.

Super user (4)

A super user has all permissions implicitly and can access the administrative web interface.

Groups (5)

Users can be part of one or more groups. A super user can add users to any groups, a staff user can only add users to groups they belong to.

Permissions (6)

These permissions are related to what a staff user can do in the administrative web interface. See *Administrative Permissions for Staff users* for information on administrative permissions.

7.1.3 Groups

A group is nothing more than a collection of *Users*. Groups can be used to apply a single *Role* to multiple Users at once.

7.2 LDAP Authentication

For organisational integration, the administrative interface supports the authentication and authorisation of users against LDAP and Active Directory. The `zonecontrol-ldap` package provides the required components and the example configuration file contains basic information on the configuration of LDAP.

Configuration of the LDAP server, domain components, common names for groups etcetera should be made in the `/etc/zonecontrol/settings.py` file. The exact configuration depends on the organisation's LDAP tree and attributes, the *OX PowerDNS Team* can help with the integration. More information can be found on the django-ldap-auth website.

7.3 Administrative Permissions for Staff users

Permissions consist of 3 parts, delimited by a pipe (|), these are in order:

- Category of the permissions
- Object of the permission in the category
- Permission on this object

There are 4 kinds of permissions.

view
The staff user can only see this object, but not change it.

change
Viewing and altering the object is permitted

add
The user can add an object of this type.

delete
The user is able to delete objects of this type.

Note: It is **highly** recommended to not give add, delete, and change permissions for objects in the zonecontrol category to users. These objects can usually be manipulated via the zone-editor. In the administrative interface, Role restrictions are enforced for many of these objects regardless.

7.3.1 admin category

These are permissions related to the administrative web interface.

log entry
Access permissions for Log Entries of all actions done by users in the administrative interface.

7.3.2 auth category

This category relates to all authentication and authorization.

user
Access permissions for *Users*.

group
Access permissions for *Groups*.

Permission details

Users with the `auth | group | Can change group` and `auth | user | Can change user` permissions and staff status can add users to the groups they themselves already belong to. Groups the staff-user is not a member of, are not shown to them unless they have the superuser permissions.

Only the super-user can:

- See or give staff status
- See or give superuser status
- Edit user permissions
- Edit Group permissions

Recommended auth permissions for staff-user are:

```
auth | user | Can change user
auth | group | Can change group
```

7.3.3 authtoken category

This category is about users' API tokens.

Token
Access permissions for Tokens.

7.3.4 zonecontrol category

This category has all the objects that are stored inside zone control.

auditlog
These permissions control the user's access to the audit logs. Note that audit logs are read-only, whether or not the user has "change" or "delete" permissions.

role
Permissions for *Roles*.

scheduled change
Access to the administrative page for scheduled changes.

server
Permissions on the *Servers* objects.

zone comment
Permissions relating to Zone Comments visible in the administrative interface.

zone version
Permissions relating to Zone Comments visible in the administrative interface.

7.4 Scheduled Changes

ZoneControl stores changes that need to happen at a certain time as a Scheduled Change.

A scheduled change as shown in the interface can be seen in Fig. 7.2.

When a staff-user is allowed by any of its roles to access the Scheduled Change and the change has not occured yet, these fields can be edited in the administrative interface:

- Zone
- RR name
- Type
- TTL
- RR records
- API server
- Scheduled at

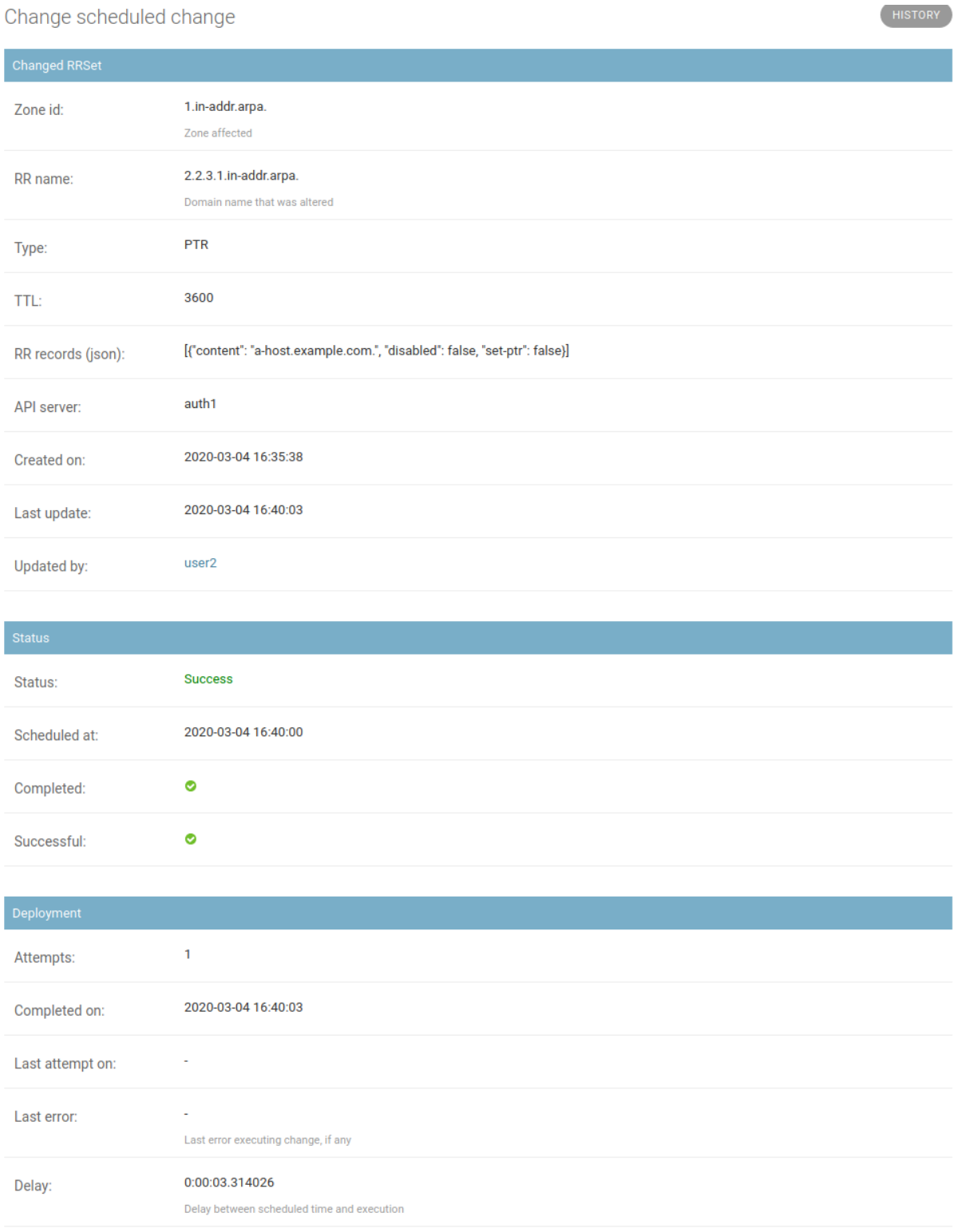


Fig. 7.2: Finished scheduled change in the admin interface

7.5 Roles

A Role represents the permissions *Users* or *Groups* have to view, manipulate, update and delete domains from the user-facing web interface.

A Role consists of a (list of) *Server(s)* and Zones that Users and Groups with this Role have access to, as well as a list of Permissions.

7.5.1 Configuring access to zones

Note: Domain names must be specified with a trailing dot!

Zones can be specified as either complete domainnames, or wildcards:

```
example.com.
```

This will give the user with this role access to `example.com`, but not to subdomains.

```
*.example.org.
```

This will give the user with this role access to all domains ending with `.example.org`, excluding `example.org` itself.

```
*
```

This wildcard gives the user access to all domains on all the Servers it can access.

A Zone that does not match in any of the Roles applied to a User is not shown in the web interface for the user.

Having access does not mean the User is allowed to manipulate the domains, these permissions can be set in the role.

Allowing access to certain records (RRSets)

New in version 1.1.0.

When a *Role* has access to a zone, it might be that it only requires access to a subset of records. The "website" role, for instance, could only require access to the A and AAAA records for the domain itself and the 'www'-variant. In a similar vain, the "mail-admin" roles could only require access to the MX records of a domain.

The syntax to limit the RRSets a Role can access within the Zones it has access to, is similar to the one used for domains. This syntax is `domainname/rrtype(s)`, one per line. For example:

```
example.com./A,AAAA
www.example.com./A,AAAA
```

Allows this role access to the A and AAAA records for example.com and www.example.com, but nothing more. When a Role requires access to all MX records of all Zones it can access, the following can be used:

```
*/MX
```

RRSets that do not match are not shown in the web interface and can not be created by the role. When checking the RRSet permissions, when one of the permissions allows the access, it is allowed.

By default, a Role has access to all RRSets in the Zones it can access:

```
*
```

RRSet permission interactions with other permissions

When a Role also has the *"Restore version"* permission, the RRSet permission is **not** checked when restoring.

When the Role has *"Readonly"* permissions, only the allowed RRSets are shown. Editing, deleting or adding is still disallowed.

7.5.2 Permissions

Apart from what Zones and RRSets a Role can see, ZoneControl allows configuration of **what** can be done with these zones and records.

The following permissions exists for each Role:

Readonly

This permission allows users to only view the Zone contents but not edit them in any way.

Add zones

This permission allows users to add Zones to the *Servers* they have access to. When a User attempts to add a Zone, the name of the Zone has to match the domain name(s) in this role.

Delete zones

Deleting Zones from and Authoritiative Server can lead to unreachability of a domain, depending on the exact DNS infrastructure setup. This is why this is a special permission.

Change settings

This permission allows users to edit settings for a Zone. e.g. to change a Zone from Primary to Secondary, setting AXFR ACLs, Notification settings etcetera.

Change DNSSEC

This permission allows the user to enable DNSSEC on a Zone.

Restore version

When a user has this permission, it can revert the whole zone to a previous version, discarding all changes to the Zone since that version.

All permissions

Gives all the above (apart from Readonly) permissions to users with this Role.

Advanced DNSSEC

This permission allows the user to access the "Advanced" tab for zones. Here, single keys can be created, uploaded, removed, or set (in)active.

7.5.3 Users and Groups with multiple Roles

Users and Groups can have more than one role assigned. When this is the case, the most 'open' permission is used.

For instance, when a user has 2 roles configured like this:

```
role1:
  Access to Zones:
    - example.com.
  RRSets:
    - */A,AAAA
  Permissions: (None specified)

role2:
  Access to Zones:
    - example.com.
  RRSets:
    - *
  Permissions:
    - Read Only
```

The user can see all records in example.com, but can only edit the A and AAAA records in the zone.

7.6 Servers

A Server is a PowerDNS Authoritative Server that exposes its API for use by ZoneControl. The following attributes are required when creating a Server:

- Name: A unique name for this Server. This is displayed in various places in the Web Interface
- Base url: The URL where this server can be contacted, e.g. `https://ns1.example.net:8443`
- API Key: The access token to authenticate to the API. Must be same as the one set in the api-key setting of the Authoritative Server.

When adding a Server, ZoneControl will validate the connectivity before allowing it to be saved.

By default ZoneControl uses a trust store of the Python installation bundled with the ZoneControl packages. Should administrators prefer to use a different trust store, for example the OS one, its location can be configured by the environmental variable `REQUESTS_CA_BUNDLE`. For the web interface the environmental variable needs to added to the systemd unit file. This file is located at `/etc/systemd/system/zonecontrol.service.d/override.conf` and can be created either manually or by using `systemctl edit zonecontrol.service` command. The example is listed below:

```
[Service]
Environment=REQUESTS_CA_BUNDLE=/etc/ssl/certs/ca-certificates.crt
```

Note: The command line tools of ZoneControl, like `zonecontrol-manage` for example, do not read the content of the systemd service and the above environmental variable needs to be set either for each execution or in the general environmental the tool is executed.

CHAPTER

EIGHT

CHANGELOG

8.1 1.6.1 (April 23, 2024)

8.1.1 DEPLOYMENT NOTES

The option to set the password for Redis has been added by the addition of `ZONECONTROL_CACHE_ZONES_PASSWORD` to `settings.py`. By default it is set to `None` which will give the same behaviour as previous ZoneControl versions.

8.1.2 IMPROVEMENTS

- Add option for Redis password in cache config (!309)

8.2 1.6.0 (March 28, 2024)

8.2.1 DEPLOYMENT NOTES

Catalog zones are only supported for PowerDNS Authoritative Server v4.9.0 and higher. Enable catalog zone support by setting `ZONECONTROL_CATALOG_ZONE_SUPPORT` to `True` in `settings.py`.

Note that a customer can have a custom `ZONECONTROL_ALLOW_ZONE_KINDS` set, if `ZONE_KIND_PRODUCER` and `ZONE_KIND_CONSUMER` are not in there the customer will not be able to add these zones through ZoneControl. (But will be able to use catalog functionality for existing Producer zones.)

Audit logging can now also be echoed to standard out or a file. This is disabled by default and can be enabled by setting `ZONECONTROL_AUDITLOG_STDOUT` to `True` in `settings.py`. By default this logging will be in JSON format but it is also possible to disable this and have plain text output; this is done by setting `ZONECONTROL_JSON_LOGGING` to `False`. (The intention is to switch to JSON logging output for ALL ZoneControl logging in some future major version.)

There's now a `showversion` management command that will show the ZoneControl version for an official release. Otherwise it will show the hardcoded `0.0.123` value.

8.2.2 NEW FEATURES

- Support for catalog zones (!301)
- Add `showversion` management command (!304)

8.2.3 IMPROVEMENTS

- Warn when chunk length is over 255 bytes for Lua records (!302)
- Improve warning when loading zones that do not exist (!305)
- Also, optionally, echo audit logging to standard out or a file (!307)

8.2.4 BUG FIXES

- Improve error handling for Prometheus metrics code (!303)
- Return `409 Conflict` (like the Auth) instead of a stacktrace and internal server error when creating a zone that already exists (!306)

8.3 1.5.1 (November 1, 2023)

8.3.1 DEPLOYMENT NOTES

Prometheus metrics are now published on `/metrics` (the default for Prometheus and in our other products). By default this is turned *off* and can be turned on by setting `ZONECONTROL_METRICS_ENABLED` to `True` in `settings.py`. Accessing the metrics requires no authentication. The number of zones metric (`zonecontrol_zones`) is only exported when the caching of zones is enabled (`ZONECONTROL_CACHE_ZONES = True`).

8.3.2 BUG FIXES

- Make sure users with RRSet limits cannot export zones (!296, !297)
- Fix issue with `Native` staying the default in the background when removing it from `ZONECONTROL_ALLOW_ZONE_KINDS` (!295)

8.3.3 NEW FEATURES

- Publish Prometheus metrics on `/metrics` (!298)

8.4 1.5.0 (October 2, 2023)

8.4.1 DEPLOYMENT NOTES

Zone names are now normalized when parsing them from the API or UI. In the past Curl scripts used for migrations at customers used zones without a dot at the end. These will still work but realize the zone will *always* get a dot at the end inside ZoneControl now.

Database tables for zone audit logs and zone history at customers may still have entries without a dot at the end. This release comes with a migration to append a dot to zones without a dot (at the end) in the audit log and zone history tables. **Please test this migration in a non-production environment first!**

The HTTP status response code for a successful `POST` call to `api/v1/servers/«SERVER_ID»/zones` has been changed from `200` to `201`. This is the API call for creating a zone.

This might affect software and scripts that check for an exact 2xx status response instead of checking for a success response code.

It is now possible to override the Gunicorn config with an `/etc/zonecontrol/gunicorn_overrides.py` file. This procedure is similar to how it is done in the Platform Filter and is already implemented in the ZoneControl Ansible role.

8.4.2 BUG FIXES

- Fix silent failure when adding a zone with non-canonical nameservers (no dot at the end) (!294)
- Normalize zone names when receiving them from the API or UI (!286)
- Make background opaque on template dropdown UI element (!285)
- Check for AnonymousUser in more places, preventing stack traces (!284)
- Make ZoneControl API status code for creating a zone equal that of the Authoritative API (!276)

8.4.3 NEW FEATURES

- Support Gunicorn config overrides (!277)

8.4.4 IMPROVEMENTS

- Remove the DNSSEC dialog flicker when opening a zone for the second (or more) time; this also removes a lot of errors on the dev console (!287)
- Fix several build issues making them less flaky and faster (!289, !282, !281)
- Adapt PDNS Auth API tests for ZoneControl, adding a new test suite (!280)
- Add documentation for zone templates (!292)

8.5 1.4.2 (May 19, 2023)

8.5.1 DEPLOYMENT NOTES

`ZONECONTROL_NSEC3PARAM_DEFAULT` can be changed in `settings.py`, but it is advised to keep it at the default of `"1 0 0 -"`.

8.5.2 BUG FIXES

- Support for PEP 440 compliant version numbers so RPM packages built again (!270)

8.5.3 NEW FEATURES

- NSEC3 support in DNSSEC settings for zones (!271)

8.5.4 IMPROVEMENTS

- Added EL9 packages (!275)
- Upload documentation from GitLab CI to docs.powerdns.com (!273)
- Auth API docs are now proxied through ZoneControl (!272)

8.6 1.4.1 (September 2, 2022)

8.6.1 BUG FIXES

- Fix quotes in LUA example record (!261)
- App: hide PTR checkbox when auto PTRs are disabled (!261)
- Fix crash due to double saving of audit logs (!261)
- Fix editing of LUA records (ZC-156, !261)

8.7 1.4.0 (March 10, 2022)

From this release onwards Debian Buster is no longer supported.

8.7.1 NEW FEATURES

- Admin option `ZONECONTROL_SERVERS_READONLY` (ZC-147, !259)
- Allow disabling history. scheduling, comments and auditlog (ZC-147, !257)

8.7.2 IMPROVEMENTS

- Upgrade to a Python 3.9 that we ship (ZC-150, !258)

8.8 1.3.4 (October 14, 2021)

8.8.1 BUG FIXES

- Take the Server into account when displaying RRSet comments (ZC-144, !256)

8.9 1.3.3 (July 09, 2021)

8.9.1 NEW FEATURES

- Implement the search API endpoint with permission checking (ZC-120, !249)

8.9.2 BUG FIXES

- Stop frontend freezing when searching for template zones with many eligable zones (ZC-130, !253)

8.10 1.3.2 (March 25, 2021)

8.10.1 IMPROVEMENTS

- Expose that ZoneControl does APIv1 on `/api`, as some clients use this information (ZC-117, !247)

8.10.2 BUG FIXES

- Adding and editing a root zone (`.`) now works in the ZoneControl frontend (ZC-118, !248)
- Fixed a broken shebang in `/usr/share/zonecontrol/bin/manage.py` in the Debian packages (!250)

8.11 1.3.1 (January 21, 2021)

8.11.1 NEW FEATURES

- PostGreSQL support has been added and can be enabled by installing `zonecontrol-postgresql` (!245)

8.11.2 BUG FIXES

- Zone names starting with an _ are no longer improperly escaped (ZC-98, !244)

8.11.3 IMPROVEMENTS

- `zonecontrol-cache` has been split into a seperate package for Debian, as was the case for EL

8.12 1.3.0 (November 26, 2020)

- Alpha 1: September 22, 2020

8.12.1 NEW FEATURES

- `zonecontrol-ldap` package has been added for integration into LDAP/AD environments (ZC-93, !230)

8.12.2 BUG FIXES

- Requests that trigger the creation of an Auditlog no longer fail when the User Agent string is too long (ZC-88, !228)
- Ensure the API endpoints return `Content-Type: application/json` (ZC-92, !232)

8.12.3 IMPROVEMENTS

- Add OPENPGPKEY help text (ZC-7, !233)
- Update the link to the ALIAS record howto (!233)
- Audit log now uses X-Forwarded-For header to get the remote IP address when ZoneControl is behind a proxy (ZC-91, !229)
- Remove the proprietary HTTP Redirect record (!233)

8.13 1.2.0 (June 15, 2020)

- Alpha 1: February 26 2020
- Beta 1: March 16 2020
- RC 1: April 22, 2020

8.13.1 NEW FEATURES

- Implement automatic PTR record creation (ZC-42, !196)
- The zones list can be cached in Redis, drastically improving performance of the web-interface when PowerDNS is serving many domains (!206)
  - This requires the `zonecontrol-cache` service to be running
  - This caching can be controlled by the the `ZONECONTROL_CACHE_ZONES_*` settings
- The DNSSEC information is not longer requested by default for the zone-overview, improving performance of the frontend (!203)
  - This can be controlled with the `ZONECONTROL_ZONE_OVERVIEW_DNSSEC` configuration option
- The 'edited' serial is now displayed in the zone-overview (ZC-40, !201)
- New distribution support:
  - RHEL/CentOS/OL 8
  - Debian Buster

8.13.2 BUG FIXES

- The slave notifications status is hidden for Native zones (ZC-52, !208)
- Underscored records no longer appear in the wrong place in the interface (ZC-5, !207)
  - This also improves the display of rrset with the same owner name

8.13.3 IMPROVEMENTS

- Permission management has been tightened (ZC-1, !194)
  - Users with these permissions and staff status can add users to their groups:
    * auth | group | Can change group
    * auth | user | Can change user
  - The same user can not see groups they don't belong to
  - Staff users with 'Can change user' permissions can no longer - See or give staff status - See or give superuser status - Edit user permissions
  - No one but superuser can now edit - User permissions - Group permissions
  - Users can only see users from their groups in the user-admin, but they can add them to their groups in the group-admin
  - Several fields in the administrative interface have been hidden as a result
  - Zone and server permissions are now enforced for staff-users in the administrative back-end (ZC-14, !216)
- Several visuals in the frontend have been harmonized and improved (!215)
- Django is updated to the latest LTS version, 2.2 (!204)
- Javascript components are upgraded. Most importantly, Angular is now at version 7.2 (!193)
- Documentation has been updated with all new features and sections on the permissions model

8.14 1.1.1 (July 10, 2019)

8.14.1 BUG FIXES

- Don't have an empty result when the per-RRset limits list is longer than 1 (!199)

8.15 1.1.0 (June 24, 2019)

- Alpha 1: internal only
- Alpha 2: internal only
- Alpha 3: May 20 2019
- Beta 1: May 23 2019

8.15.1 BREAKING CHANGES

- Support for the PowerDNS Authoritative Server 4.0 has been dropped. (#146)
- pkgs: Depend on the Python 3.6 from EPEL, not on the one from SCL (!186)

8.15.2 NEW FEATURES

- Ability to limit the kinds of zones that are allowed to be created. By default, all 3 kinds are allowed. This limit can be set in the settings.py file by the system administrator. (!172)
- The zone overview is paginated, showing 25 zones per page. (PDNS-37, !171)
- The records overview is paginated, showing 200 records per page. (PDNS-37, !171)
- PTR records can be automatically generated by appending a '*' after the ip address. This feature can be disabled in the settings.py by the system administrator. (PDNS-158, PDNS-189, !174)
- The zone settings dialog has been extended with buttons to queue NOTIFY messages (for master zones) and retrieve the zone (for slave zones). The system administrator can disable this feature in settings.py. (!173)
- The DNSSEC dialog shows what kind of keys will be created once the user clicks "Enable DNSSEC". (PDNS-39, !175)
- An 'advanced DNSSEC' role permission has been added. Users with this permission have a tab in the DNSSEC dialog that allows manipulation of all keys. (PDNS-39, !175)
- Per-RRSet role-based access control has been added. This feature allows roles to be restricted to records of certain names and types. By default, all records are allowed when a role has access to a zone. (PDNS-165, PDNS-35, !178)

8.15.3 BUG FIXES

- Fix alert when attempting to add an internally managed record at the apex. (#169)
- Ensure the UI shows 'repeated' zone names. (PDNS-206, #166)
- Mark required textareas in forms. (#167)

8.16 1.0.1 (July 17, 2018)

8.16.1 NEW FEATURES

- Implement rolebased API "pass through" (#128)
- Comments for zone records (#122) - Tool to import comments from PowerDNS Authoritative Server
- roles: Granular permissions for zone settings, DNSSEC and version restores (#104)

8.16.2 IMPROVEMENTS

- pkg: Let yum figure out which mysql lib to pull in (#144)
- pkg: Add MySQL support package for CentOS 7 (#127)
- admin: More user-friendly role admin UI (#104)
- dnssec: When enabling DNSSEC, the configured DNSSEC server defaults are now honored (#105)

8.16.3 BUG FIXES

- Fix API error when adding or removing zone (#139)
- Catch SOA entries that do not exist (#133)
- backend: Fix 500 error when adding non-template zone (cf0393e)
- builder: Fix missing styling in built packages, debian build fixes (#95)

8.17 1.0.0 (September 20, 2017)

First release.

8.17.1 NEW FEATURES

- Zone version history and restore (#56)
- Zone templates (#63)
- builder: New build system, CentOS 7 and Debian Stretch packages (#70)

IMPROVEMENTS: - libs: Upgraded to Angular 4.3.6 and Webpack 3.5.5

BUG FIXES: - dnssec: Correct handling of ksk/zsk keys in UI (#65)

HTTP ROUTING TABLE

/api

GET /api/v1/servers/{server_id}/zones/{zone_id}/_history, 19
GET /api/v1/servers/{server_id}/zones/{zone_id}/_history/diff/{version_1}/{version_2}, 19
GET /api/v1/servers/{server_id}/zones/{zone_id}/_history/diff/{version_2}, 19
GET /api/v1/servers/{server_id}/zones/{zone_id}/_names/{name}/_comments/{rrtype}, 18
GET /api/zonecontrol/info, 17
POST /api/v1/servers/{server_id}/zones/{zone_id}/_history/restore/{version}, 19
POST /api/v1/servers/{server_id}/zones/{zone_id}/_names/{name}/_comments/{rrtype}, 18
POST /api/zonecontrol/login, 17
DELETE /api/v1/servers/{server_id}/zones/{zone_id}/_names/{name}/_comments/{rrtype}/_comment/{commen 18