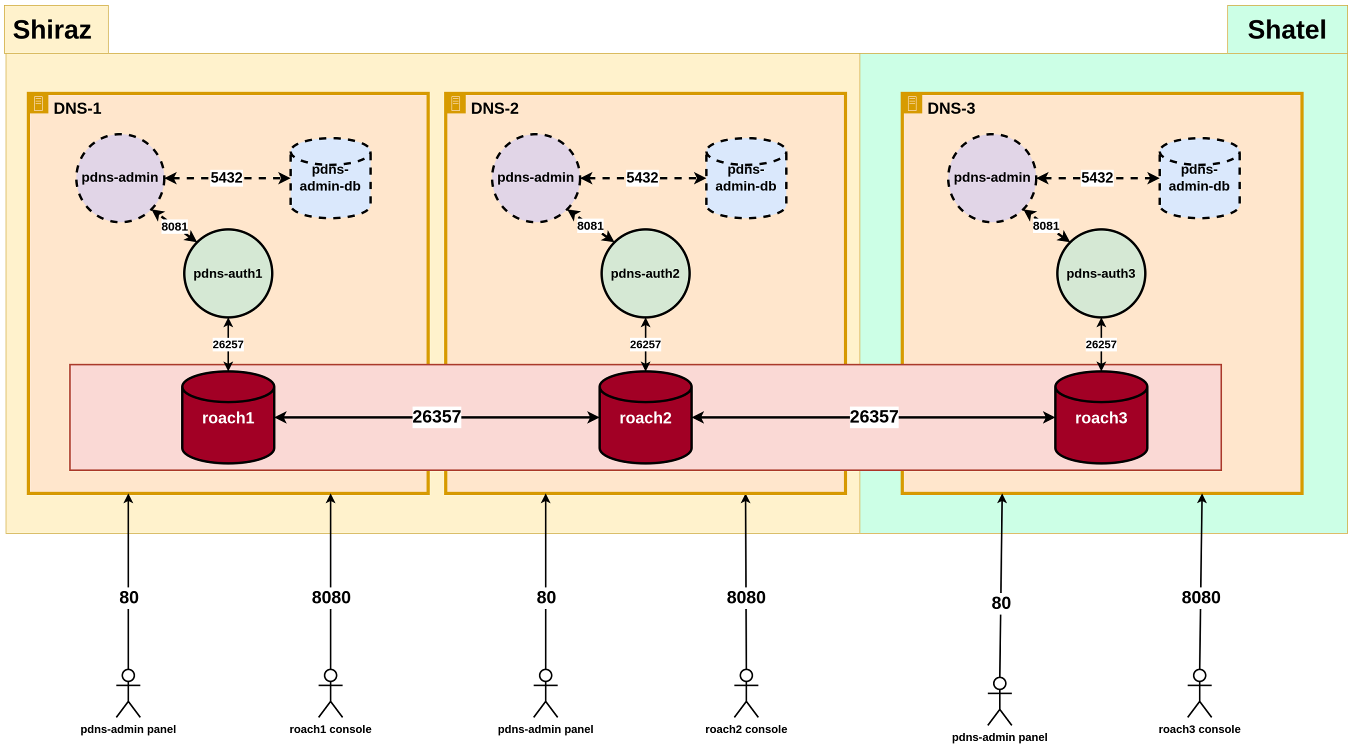
PowerDNS Authoritative Server for XaaS

Architecture



System Requirement

- for cockroach
  - 8CPUs, 32GB RAM, 40 GB Volume
  - measure your IOPS, and MB/s:

```
fio --filename=/tmp/fiotest.tmp --ioengine=libaio --direct=1 --group_reporting --name=Randwrite --bs=4k --
rw=randwrite --numjobs=<number of cpus> --iodepth=32 --size=1g
```

- for powerdns
  - 2CPUs, 2GB RAM, 4GB swap
- for host
  - systems shouldn't have private IP and just have public IPv4

Docker

- Install it!

Create CockroachDB Cluster

- Clock Synchronization and Time Zone

```
# on three servers do these
# insert these two variables in to /etc/systemd/timesyncd.conf
# tip for NTP = in shiraz it NTP works and doesn't works for shatel:
NTP=asia.pool.ntp.org
# in both it NTP works:
NTP=ir.pool.ntp.org
# after choosing one of NTPs, now for both insert these:
FallbackNTP=0.asia.pool.ntp.org 1.asia.pool.ntp.org 2.asia.pool.ntp.org 3.asia.pool.ntp.org
# restart the timesyncd service
systemctl restart systemd-timesyncd
# change the time zone:
timedatectl set-timezone Asia/Tehran
# see the output
# Time zone: Asia/Tehran (+0330, +0330)
# System clock synchronized: yes
# NTP service: active

timedatectl status
```

- Generate Certificates

```
# on three servers do these
# roach source files
mkdir -p /root/cockroachdb/certs /root/cockroachdb/my-safe-directory
# insert these to /etc/hosts because we are using host network for containers
<dns-1 IPv4> roach1
<dns-2 IPv4> roach2
<dns-3 IPv4> roach3
```

Up These Compose Files Just for Generating Certificates for Each Server

```
# on DNS1
cd /root/cockroachdb
docker compose -f roach1.yml up -d
```

```
# roach1.yml
---
services:
  roach1:
    image: docker.arvancloud.ir/cockroachdb/cockroach:v24.1.2
    container_name: roach1
    hostname: roach1
    network_mode: host
    volumes:
      - ./roach1:/cockroach/roach1
      - ./certs:/certs
      - ./my-safe-directory:/my-safe-directory
      - /etc/localtime:/etc/localtime:ro
    stdin_open: true
    tty: true
    restart: always
    logging:
      driver: "json-file"
      options:
        max-size: "10m"
        max-file: "5"
    command: >
      bash -c "
      cockroach cert create-ca  --certs-dir=/certs --ca-key=/my-safe-directory/ca.key --allow-ca-key-
reuse
      && cockroach cert create-node localhost roach1 <dns-1 IPv4> --certs-dir=/certs --ca-key=/my-safe-
directory/ca.key --overwrite
      && cockroach cert create-client root --certs-dir=/certs --ca-key=/my-safe-directory/ca.key --
overwrite
      && cockroach start --certs-dir=/certs --store=roach1 --advertise-addr=roach1:26357 --http-
addr=roach1:8080 --listen-addr=roach1:26357 --sql-addr=roach1:26257 --join=roach1:26357,roach2:26357,
roach3:26357
      "

# --http-addr=roach1:8080        ----->  admin can access to DB console by port 8080 and IP address you
insert for roach1.
# --advertise-addr=roach1:26357  ----->  (it is optional) by advertising port 26357, inter-node cluster
traffic between nodes
#                                        is just on this port on each node. it tells other nodes to use
this IP and port for
#                                                                             speacking with this
container. This
#                                        port is not published, so inter-node traffic does not leave
this network.(this is
#                                        helpful when we use a specific network for containers not host
network mode! but we did it!).
# --listen-addr=roach1:26357     ----->  it will listen on this port and IP.
# --sql-addr=roach1:26257        ----->  it is the IP and port for client SQL traffic.
```

```
# send certificates
scp my-safe-directory/ca.key root@<dns-{2 and 3}-IPv4>:/root/cockroachdb/my-safe-directory/
scp certs/ca.crt root@<DNS{2 and 3}-IP>:/root/cockroachdb/certs/
```

```
# on DNS2 and DNS3 do it
# change permition for ca.key
chmod 0600 /root/cockroachdb/my-safe-directory/ca.key
```

```
# on DNS2
cd /root/cockroachdb
docker compose -f roach2.yml up -d
```

```
# roach2.yml
---
services:
  roach2:
    image: docker.arvancloud.ir/cockroachdb/cockroach:v24.1.2
    container_name: roach2
    hostname: roach2
    network_mode: host
    volumes:
      - ./roach2:/cockroach/roach2
      - ./certs:/certs
      - ./my-safe-directory:/my-safe-directory
      - /etc/localtime:/etc/localtime:ro
    stdin_open: true
    tty: true
    restart: always
    logging:
      driver: "json-file"
      options:
        max-size: "10m"
        max-file: "5"
    command: >
      bash -c "
      cockroach cert create-node localhost roach2 <dns-2 IPv4> --certs-dir=/certs --ca-key=/my-safe-
directory/ca.key --overwrite
      && cockroach cert create-client root --certs-dir=/certs --ca-key=/my-safe-directory/ca.key --
overwrite
      && cockroach start --certs-dir=/certs --store=roach2 --advertise-addr=roach2:26357 --http-
addr=roach2:8080 --listen-addr=roach2:26357 --sql-addr=roach2:26257 --join=roach1:26357,roach2:26357,
roach3:26357
      "
```

```
# on DNS3
cd /root/cockroachdb
docker compose -f roach3.yml up -d
```

```
# roach3.yml
---
services:
  roach3:
    image: docker.arvancloud.ir/cockroachdb/cockroach:v24.1.2
    container_name: roach3
    hostname: roach3
    network_mode: host
    volumes:
      - ./roach3:/cockroach/roach3
      - ./certs:/certs
      - ./my-safe-directory:/my-safe-directory
      - /etc/localtime:/etc/localtime:ro
    stdin_open: true
    tty: true
    restart: always
    logging:
      driver: "json-file"
      options:
        max-size: "10m"
        max-file: "5"
    command: >
      bash -c "
      cockroach cert create-node localhost roach3 <dns-3 IPv4> --certs-dir=/certs --ca-key=/my-safe-
directory/ca.key --overwrite
      && cockroach cert create-client root --certs-dir=/certs --ca-key=/my-safe-directory/ca.key --
overwrite
      && cockroach start --certs-dir=/certs --store=roach3 --advertise-addr=roach3:26357 --http-
addr=roach3:8080 --listen-addr=roach3:26357 --sql-addr=roach3:26257 --join=roach1:26357,roach2:26357,
roach3:26357
      "
```

remove everything and just keep the certificates

```
# on DNS1
cd /root/cockroachdb
docker compose -f roach1.yml down
rm -r /root/cockroachdb/roach1
```

```
# on DNS2
cd /root/cockroachdb
docker compose -f roach2.yml down
rm -r /root/cockroachdb/roach2
```

```
# on DNS3
cd /root/cockroachdb
docker compose -f roach3.yml down
rm -r /root/cockroachdb/roach3
```

- Start Creating Cluster (mix of 1, 2, and 3 sources)

```
# on DNS1
cd /root/cockroachdb
# first edit the yml file and then up it
docker compose -f roach1.yml up -d
```

```
# roach1.yml
---
services:
  roach1:
    image: docker.arvancloud.ir/cockroachdb/cockroach:v24.1.2
    container_name: roach1
    hostname: roach1
    network_mode: host
    volumes:
      - ./roach1:/cockroach/roach1
      - ./certs:/certs
      - ./my-safe-directory:/my-safe-directory
      - /etc/localtime:/etc/localtime:ro
    stdin_open: true
    tty: true
    restart: always
    logging:
      driver: "json-file"
      options:
        max-size: "10m"
        max-file: "5"
    command: >
      bash -c "
          cockroach start --certs-dir=/certs --store=roach1 --advertise-addr=roach1:26357 --http-
addr=roach1:8080 --listen-addr=roach1:26357 --sql-addr=roach1:26257 --join=roach1:26357,roach2:26357,
roach3:26357
      "
```

```
# on DNS2
cd /root/cockroachdb
# first edit the yml file and then up it
docker compose -f roach2.yml up -d
```

```
# roach2.yml
---
services:
  roach2:
    image: docker.arvancloud.ir/cockroachdb/cockroach:v24.1.2
    container_name: roach2
    hostname: roach2
    network_mode: host
    volumes:
      - ./roach2:/cockroach/roach2
      - ./certs:/certs
      - ./my-safe-directory:/my-safe-directory
      - /etc/localtime:/etc/localtime:ro
    stdin_open: true
    tty: true
    restart: always
    logging:
      driver: "json-file"
      options:
        max-size: "10m"
        max-file: "5"
    command: >
      bash -c "
          cockroach start --certs-dir=/certs --store=roach2 --advertise-addr=roach2:26357 --http-
addr=roach2:8080 --listen-addr=roach2:26357 --sql-addr=roach2:26257 --join=roach1:26357,roach2:26357,
roach3:26357
      "
```

```
# on DNS3
cd /root/cockroachdb
# first edit the yml file and then up it
docker compose -f roach3.yml up -d
```

```
# roach2.yml
---
services:
  roach3:
    image: docker.arvancloud.ir/cockroachdb/cockroach:v24.1.2
    container_name: roach3
    hostname: roach3
    network_mode: host
    volumes:
      - ./roach3:/cockroach/roach3
      - ./certs:/certs
      - ./my-safe-directory:/my-safe-directory
      - /etc/localtime:/etc/localtime:ro
    stdin_open: true
    tty: true
    restart: always
    logging:
      driver: "json-file"
      options:
        max-size: "10m"
        max-file: "5"
    command: >
      bash -c "
          cockroach start --certs-dir=/certs --store=roach3 --advertise-addr=roach3:26357 --http-
addr=roach3:8080 --listen-addr=roach3:26357 --sql-addr=roach3:26257 --join=roach1:26357,roach2:26357,
roach3:26357
      "
```

```
# on DNS1
# initial the cluster
docker exec -it roach1 ./cockroach --host=roach1:26357 --certs-dir=/certs init
# see cluster info
docker exec -it roach1 grep 'node starting' roach1/logs/cockroach.log -A 13
```

- ## Test the Cluster

```
# on DNS1
docker exec -it roach1 ./cockroach --certs-dir=/certs --host=roach1:26257 sql
```

```
--- run these queries
CREATE DATABASE bank;
CREATE TABLE bank.accounts (id INT PRIMARY KEY, balance DECIMAL);
INSERT INTO bank.accounts VALUES (1, 1000.50);
SELECT * FROM bank.accounts;
```

```
# on DNS2
docker exec -it roach2 ./cockroach --certs-dir=/certs --host=roach2:26257 sql
```

```
SELECT * FROM bank.accounts;
--- you shold see this:
---     id | balance
---   +----+---------+
---      1 | 1000.50
```

```
# on DNS3
docker exec -it roach3 ./cockroach --certs-dir=/certs --host=roach3:26257 sql
```

```
SELECT * FROM bank.accounts;
--- you shold see this:
---     id | balance
---   +----+---------+
---      1 | 1000.50
```

```
--- if you've seen correct and same output, delete the test parts by these queries(it is not important
in which server)
USE bank;
DROP TABLE accounts;
USE defaultdb;
DROP DATABASE bank;
```

- Setup Database Schema for PowerDNS Authoritative Server

```
# on DNS1
docker exec -it roach1 ./cockroach --certs-dir=/certs --host=roach1:26257 sql
```

```
--- prerequisite

CREATE DATABASE pdns_db;
USE pdns_db;
CREATE USER pdns_usr WITH LOGIN PASSWORD 'xaas@32n53e';
GRANT ALL ON DATABASE pdns_db TO pdns_usr;
ALTER DATABASE pdns_db OWNER TO pdns_usr;

--- schema.pgsql.sql

CREATE SEQUENCE domain_id MAXVALUE 2147483648;
CREATE SEQUENCE record_id MAXVALUE 2147483648;
CREATE SEQUENCE comment_id MAXVALUE 2147483648;
CREATE SEQUENCE meta_id MAXVALUE 2147483648;
CREATE SEQUENCE key_id MAXVALUE 2147483648;
CREATE SEQUENCE tsig_id MAXVALUE 2147483648;

CREATE TABLE domains (
  id                      INT DEFAULT nextval('domain_id') PRIMARY KEY,
  name                    VARCHAR(255) NOT NULL,
  master                  VARCHAR(128) DEFAULT NULL,
  last_check              INT DEFAULT NULL,
  type                    TEXT NOT NULL,
  notified_serial         BIGINT DEFAULT NULL,
  account                 VARCHAR(40) DEFAULT NULL,
  options                 TEXT DEFAULT NULL,
  catalog                 TEXT DEFAULT NULL,
  CONSTRAINT c_lowercase_name CHECK (((name)::TEXT = LOWER((name)::TEXT)))
);

CREATE UNIQUE INDEX name_index ON domains(name);
CREATE INDEX catalog_idx ON domains(catalog);
CREATE TABLE records (
  id                    INT DEFAULT nextval('record_id') PRIMARY KEY,
  domain_id             INT DEFAULT NULL,
  name                  VARCHAR(255) DEFAULT NULL,
  type                  VARCHAR(10) DEFAULT NULL,
  content               VARCHAR(65535) DEFAULT NULL,
  ttl                   INT DEFAULT NULL,
  prio                  INT DEFAULT NULL,
  disabled              BOOL DEFAULT 'f',
  ordername             VARCHAR(255),
  auth                  BOOL DEFAULT 't',
  CONSTRAINT domain_exists
  FOREIGN KEY(domain_id) REFERENCES domains(id)
  ON DELETE CASCADE,
  CONSTRAINT c_lowercase_name CHECK (((name)::TEXT = LOWER((name)::TEXT)))
);

CREATE INDEX rec_name_index ON records(name);
CREATE INDEX nametype_index ON records(name,type);
CREATE INDEX domain_id ON records(domain_id);


CREATE TABLE supermasters (
  ip                    INET NOT NULL,
  nameserver            VARCHAR(255) NOT NULL,
  account               VARCHAR(40) NOT NULL,
  PRIMARY KEY(ip, nameserver)
);


CREATE TABLE comments (
  id                    INT DEFAULT nextval('comment_id') PRIMARY KEY,
  domain_id             INT NOT NULL,
  name                  VARCHAR(255) NOT NULL,
  type                  VARCHAR(10) NOT NULL,
  modified_at           INT NOT NULL,
  account               VARCHAR(40) DEFAULT NULL,
  comment               VARCHAR(65535) NOT NULL,
  CONSTRAINT domain_exists
  FOREIGN KEY(domain_id) REFERENCES domains(id)
  ON DELETE CASCADE,
  CONSTRAINT c_lowercase_name CHECK (((name)::TEXT = LOWER((name)::TEXT)))
);

CREATE INDEX comments_domain_id_idx ON comments (domain_id);
CREATE INDEX comments_name_type_idx ON comments (name, type);
CREATE INDEX comments_order_idx ON comments (domain_id, modified_at);


CREATE TABLE domainmetadata (
  id                    INT DEFAULT nextval('meta_id') PRIMARY KEY,
  domain_id             INT REFERENCES domains(id) ON DELETE CASCADE,
  kind                  VARCHAR(32),
  content               TEXT
);

CREATE INDEX domainidmetaindex ON domainmetadata(domain_id);


CREATE TABLE cryptokeys (
  id                    INT DEFAULT nextval('key_id') PRIMARY KEY,
  domain_id             INT REFERENCES domains(id) ON DELETE CASCADE,
  flags                 INT NOT NULL,
  active                BOOL,
  published             BOOL DEFAULT TRUE,
  content               TEXT
);

CREATE INDEX domainidindex ON cryptokeys(domain_id);
CREATE TABLE tsigkeys (
  id                     INT DEFAULT nextval('tsig_id') PRIMARY KEY,
  name                   VARCHAR(255),
  algorithm              VARCHAR(50),
  secret                 VARCHAR(255),
  CONSTRAINT c_lowercase_name CHECK (((name)::TEXT = LOWER((name)::TEXT)))
);

CREATE UNIQUE INDEX namealgoindex ON tsigkeys(name, algorithm);

ALTER TABLE comments OWNER TO pdns_usr;
ALTER TABLE cryptokeys OWNER TO pdns_usr;
ALTER TABLE domainmetadata OWNER TO pdns_usr;
ALTER TABLE domains OWNER TO pdns_usr;
ALTER TABLE records OWNER TO pdns_usr;
ALTER TABLE supermasters OWNER TO pdns_usr;
ALTER TABLE tsigkeys OWNER TO pdns_usr;

GRANT ALL PRIVILEGES ON SEQUENCE domain_id TO pdns_usr;
GRANT ALL PRIVILEGES ON SEQUENCE record_id TO pdns_usr;
GRANT ALL PRIVILEGES ON SEQUENCE comment_id TO pdns_usr;
GRANT ALL PRIVILEGES ON SEQUENCE meta_id TO pdns_usr;
GRANT ALL PRIVILEGES ON SEQUENCE key_id TO pdns_usr;
GRANT ALL PRIVILEGES ON SEQUENCE tsig_id TO pdns_usr;
```

- CockroachDB Console

```
# on DNS1
docker exec -it roach1 ./cockroach --certs-dir=/certs --host=roach1:26257 sql
```

```
--- should create password admin
ALTER USER root WITH PASSWORD 'xaas-pass';
--- in your browser if you search DNS{1, 2, or 3}-IP:8080 you can se console
```

- CockroachDB Logging

```
# we are using default logging, and logs are in root/cockroachdb/roach{1, 2, or 3}/logs
```

PowerDNS Authoritative Server

- Disable DNS

```
# on three servers
# pay attention: if you apply these, you'll can't pull images from docker, so you should pull each image
you want and then apply these

mkdir /root/pdns-auth-server
cd /root/pdns-auth-server
systemctl disable systemd-resolved
systemctl stop systemd-resolved
systemctl mask systemd-resolved
cp /etc/resolv.conf . # just as a backup
echo "nameserver 8.8.8.8" > /etc/resolv.conf
```

- Config the PowerDNS Authoritative Server

-
  ```
  # on DNS1
  # vim /root/pdns-auth-server/pdns.conf
  # then insert this config to pdns.conf

  #-----------------------------------------Authoritative Server Settings
  local-address=0.0.0.0
  launch=gpgsql

  api=yes
  api-
  key=FA4432518B97E736238BBB7BB7736E7737875AE8F616BA56E1F93BDAC17876E4781F3A59E6CAAE1ED9873362E8E0994D7AA66
  A3A6C2D1D1293889DBDA4E77A03

  # uncomment each you need
  #log-dns-queries=yes
  #log-dns-details=yes
  #log-timestamp=yes
  #loglevel=7
  #query-logging=yes
  #loglevel-show=yes

  #-----------------------------------------enables pdns to accept pdns-admin-panel requests
  webserver=yes
  webserver-address=127.0.0.1
  webserver-port=8081
  webserver-password=eea3c6efc341750abc9146dc70f2e9498917b599

  #----------------------------------------- TCP Connection settings
  max-tcp-connections-per-client=3
  max-tcp-connections=100

  #-----------------------------------------Backend setting
  gpgsql-port=26257
  gpgsql-host=roach1
  gpgsql-dbname=pdns_db
  gpgsql-user=pdns_usr
  gpgsql-password=xaas@32n53e

  #-----------------------------------------when you want use cockroachdb instead of postgresql should
  apply these queries
  gpgsql-get-order-first-query=select ordername from records where domain_id = $1 and disabled = false and
  ordername is not null order by 1 asc limit 1
  gpgsql-get-order-before-query=select ordername, name from records where ordername <= $1 and domain_id =
  $2 and disabled = false and ordername is not null order by 1 desc limit 1
  gpgsql-get-order-after-query=select ordername from records where ordername > $1 and domain_id = $2 and
  disabled = false and ordername is not null order by 1 asc limit 1
  gpgsql-get-order-last-query=select ordername, name from records where ordername != '' and domain_id = $1
  and disabled = false and ordername is not null order by 1 desc limit 1
  ```

```
# on DNS2
# vim /root/pdns-auth-server/pdns.conf
# then insert this config to pdns.conf

#-----------------------------------------Authoritative Server Settings
local-address=0.0.0.0
launch=gpgsql

api=yes
api-
key=FA4432518B97E736238BBB7BB7736E7737875AE8F616BA56E1F93BDAC17876E4781F3A59E6CAAE1ED9873362E8E0994D7AA66
A3A6C2D1D1293889DBDA4E77A03

# uncomment each you need
#log-dns-queries=yes
#log-dns-details=yes
#log-timestamp=yes
#loglevel=7
#query-logging=yes
#loglevel-show=yes

#-----------------------------------------enables pdns to accept pdns-admin-panel requests
webserver=yes
webserver-address=127.0.0.1
webserver-port=8081
webserver-password=eea3c6efc341750abc9146dc70f2e9498917b599

#----------------------------------------- TCP Connection settings
max-tcp-connections-per-client=3
max-tcp-connections=100

#-----------------------------------------Backend setting
gpgsql-port=26257
gpgsql-host=roach2
gpgsql-dbname=pdns_db
gpgsql-user=pdns_usr
gpgsql-password=xaas@32n53e

#-----------------------------------------when you want use cockroachdb instead of postgresql should
apply these queries
gpgsql-get-order-first-query=select ordername from records where domain_id = $1 and disabled = false and
ordername is not null order by 1 asc limit 1
gpgsql-get-order-before-query=select ordername, name from records where ordername <= $1 and domain_id =
$2 and disabled = false and ordername is not null order by 1 desc limit 1
gpgsql-get-order-after-query=select ordername from records where ordername > $1 and domain_id = $2 and
disabled = false and ordername is not null order by 1 asc limit 1
gpgsql-get-order-last-query=select ordername, name from records where ordername != '' and domain_id = $1
and disabled = false and ordername is not null order by 1 desc limit 1
```

```
# on DNS3
# vim /root/pdns-auth-server/pdns.conf
# then insert this config to pdns.conf

#-----------------------------------------Authoritative Server Settings
local-address=0.0.0.0
launch=gpgsql

api=yes
api-
key=FA4432518B97E736238BBB7BB7736E7737875AE8F616BA56E1F93BDAC17876E4781F3A59E6CAAE1ED9873362E8E0994D7AA66
A3A6C2D1D1293889DBDA4E77A03

# uncomment each you need

#log-dns-queries=yes
#log-dns-details=yes
#log-timestamp=yes
#loglevel=7
#query-logging=yes
#loglevel-show=yes

#-----------------------------------------enables pdns to accept pdns-admin-panel requests
webserver=yes
webserver-address=127.0.0.1
webserver-port=8081
webserver-password=eea3c6efc341750abc9146dc70f2e9498917b599

#----------------------------------------- TCP Connection settings
max-tcp-connections-per-client=3
max-tcp-connections=100

#-----------------------------------------Backend setting
gpgsql-port=26257
gpgsql-host=roach3
gpgsql-dbname=pdns_db
gpgsql-user=pdns_usr
gpgsql-password=xaas@32n53e

#-----------------------------------------when you want use cockroachdb instead of postgresql should
apply these queries
gpgsql-get-order-first-query=select ordername from records where domain_id = $1 and disabled = false and
ordername is not null order by 1 asc limit 1
gpgsql-get-order-before-query=select ordername, name from records where ordername <= $1 and domain_id =
$2 and disabled = false and ordername is not null order by 1 desc limit 1
gpgsql-get-order-after-query=select ordername from records where ordername > $1 and domain_id = $2 and
disabled = false and ordername is not null order by 1 asc limit 1
gpgsql-get-order-last-query=select ordername, name from records where ordername != '' and domain_id = $1
and disabled = false and ordername is not null order by 1 desc limit 1
```

- PowerDNS Authoritative Server Compose files

```
# on DNS1

cd /root/pdns-auth-server
docker compose -f pdns-auth1.yml up -d
```

```
# pdns-auth1.yml

---
services:
  pdns-auth1:
    container_name: pdns-auth1
    hostname: pdns-auth1
    image: docker.arvancloud.ir/powerdns/pdns-auth-49:4.9.0
    user: root
    network_mode: host
    restart: always
    volumes:
      - ./pdns.conf:/etc/powerdns/pdns.conf
      - /etc/localtime:/etc/localtime:ro
    logging:
      driver: "json-file"
      options:
        max-size: "10m"
        max-file: "5"
```

```
# on DNS2

cd /root/pdns-auth-server
docker compose -f pdns-auth2.yml up -d
```

```
# pdns-auth2.yml

---
services:
  pdns-auth2:
    container_name: pdns-auth2
    hostname: pdns-auth2
    image: docker.arvancloud.ir/powerdns/pdns-auth-49:4.9.0
    user: root
    network_mode: host
    restart: always
    volumes:
      - ./pdns.conf:/etc/powerdns/pdns.conf
      - /etc/localtime:/etc/localtime:ro
    logging:
      driver: "json-file"
      options:
        max-size: "10m"
        max-file: "5"
```

```
# on DNS3

cd /root/pdns-auth-server
docker compose -f pdns-auth3.yml up -d
```

```
# pdns-auth3.yml

---
services:
  pdns-auth3:
    container_name: pdns-auth3
    hostname: pdns-auth3
    image: docker.arvancloud.ir/powerdns/pdns-auth-49:4.9.0
    user: root
    network_mode: host
    restart: always
    volumes:
      - ./pdns.conf:/etc/powerdns/pdns.conf
      - /etc/localtime:/etc/localtime:ro
    logging:
      driver: "json-file"
      options:
        max-size: "10m"
        max-file: "5"
```

PowerDNS Admin Panel

- Database

```
# on three servers
mkdir -p /root/pdns-admin/database
cd /root/pdns-admin/database
# secure password
vim .env
# insert it in to .env file
POSTGRES_PASSWORD=xaas@32n53e
# up the container
docker compose -f pdns-admin-db.yml up -d
```

```
# pdns-admin-db.yml

---
services:
  pdns-admin-db:
    image: docker.arvancloud.ir/postgres:16.3
    container_name: pdns-admin-db
    hostname: pdns-admin-db
    restart: always
    network_mode: host
    environment:
      - POSTGRES_PASSWORD=${POSTGRES_PASSWORD}
      - POSTGRES_USER=postgres
      - TZ=Asia/Tehran
    volumes:
      - ./postgres-data:/var/lib/postgresql/data
      - /etc/localtime:/etc/localtime:ro
    logging:
      driver: "json-file"
      options:
        max-size: "10m"
        max-file: "5"
```

- setup access to the database

```
vim ./postgres-data/pg_hba.conf
# Add the following lines
host    all             all             0.0.0.0/0                       md5
host    all             all             ::/0                            md5
```

- initialize the databese

```
# down and up the container to apply changes
docker compose -f pdns-admin-db.yml down
docker compose -f pdns-admin-db.yml up -d
docker exec -it pdns-admin-db bash
su - postgres
createuser pdns_admin_usr
createdb -E UTF8 -l en_US.UTF-8 -O pdns_admin_usr -T template0 pdns_admin_db 'The database for PowerDNS-Admin'
psql
ALTER ROLE pdns_admin_usr WITH PASSWORD 'pdns_admin_pas';
```

- Panel

  ```
  # on three servers
  mkdir /root/pdns-admin/panel
  cd /root/pdns-admin/panel
  docker compose -f pdns-admin.yml up -d
  ```

  ```
  # pdns-admin.yml

  ---
  services:
    pdns-admin:
      image: docker.arvancloud.ir/powerdnsadmin/pda-legacy:v0.4.2
      container_name: pdns-admin
      hostname: pdns-admin
      network_mode: host
      restart: always
      environment:
        - PDNS_HOST=127.0.0.1 #use pdns-authoritative server container name when network is nost host
        - PDNS_PORT=8081
        - PDNS_VERSION=4.9.0
        -
  PDNS_API_KEY=FA4432518B97E736238BBB7BB7736E7737875AE8F616BA56E1F93BDAC17876E4781F3A59E6CAAE1ED9873362E8E0
  994D7AA66A3A6C2D1D1293889DBDA4E77A03
        - OFFLINE_MODE=False
        - SECRET_KEY=I0o3qqd2eQE859gPpVCGm4EYW4fJTVnGnX2dJZ9YIejq20SJ77tW
        - SQLALCHEMY_DATABASE_URI=postgresql://pdns_admin_usr:pdns_admin_pas@127.0.0.1/pdns_admin_db #when
  you are using bridge network, use powerdns-admin-db container name
        - PORT=80
      logging:
        driver: "json-file"
        options:
          max-size: "10m"
          max-file: "5"
      volumes:
        - /etc/localtime:/etc/localtime:ro
  ```

Test the Authoritative Server

create zones

```
1. on three servers
        - access to pdns-admin panel
    - create an account, this user will be admin user
        - set PowerDNS API URL to http://127.0.0.1:8081/

-----------------------------------------------------------------

2. in DNS1 pdns-admin panel
        - create esfahani.com zone
3. in DNS2 pdns-admin panel
        - create arash.com zone
4. in DNS3 pdns-admin panel
        - create mojtaba.com zone

-----------------------------------------------------------------

5. in DNS1 pdns-admin panel create a subdomain for arash.com (example: xaas.arash.com)
6. in DNS2 pdns-admin panel create a subdomain for mojtaba.com (example: xaas.mojtaba.com)
7. in DNS3 pdns-admin panel create a subdomain for esfahani.com (example: xaas.esfahani.com)
```

test the zones

```
# now run this script and see the answer is correct or not

#!/bin/bash
figlet -w 100 -c xaas.esfahani.com
dig -t A xaas.esfahani.com @<dns-1 IPv4>
dig -t A xaas.esfahani.com @<dns-2 IPv4>
dig -t A xaas.esfahani.com @<dns-3 IPv4>
figlet -w 100 -c xaas.arash.com
dig -t A xaas.arash.com @<dns-1 IPv4>
dig -t A xaas.arash.com @<dns-2 IPv4>
dig -t A xaas.arash.com @<dns-3 IPv4>
figlet -w 100 -c xaas.mojtaba.com
dig -t A xaas.mojtaba.com @<dns-1 IPv4>
dig -t A xaas.mojtaba.com @<dns-2 IPv4>
dig -t A xaas.mojtaba.com @<dns-3 IPv4>
```

Import Zones and Records From Bind Format

for txt records the data section should be '"*******"', do it by vim record.

docker cp txt-records.sql roach1:/txt-records.sql

docker exec -it roach1 ./cockroach --certs-dir=/certs --host=roach1:26257 sql --database=pdns_db --file=/mx-records.sql

Previous Report

PowerDNS+insta...+dashboard.pdf