19. Deploying omnidb-server

Whenever deploying omnidb-server the user must be aware of how OmniDB works in terms of ports so the environment can be properly configured taking the infrastructure into account.

OmniDB uses 2 servers to answer user requests, one is the default webserver serving the application itself and the other is a websocket server used by several parts of OmniDB, such as Query, Console and Debugging Tab, allowing a bi-directional communication between the client and the server which enhances performance and user experience. This means that 2 ports need to be properly configured:

- OmniDB server:
  - **Technology**: CherryPy
  - **Default port**: 8000

- Websocket server:
  - **Technology**: Tornado
  - **Default port**: 25482

Both servers support SSL so OmniDB can run by itself securely without the need of a load balancer or reverse proxy, such as Nginx.

The configuration of ports and certificates can be done via command options or configuration file.

Command options

```
Usage: omnidb-server [options]

Options:
  --version             show program's version number and exit
  -h, --help            show this help message and exit
  -H HOST, --host=HOST  listening address
  -p PORT, --port=PORT  listening port
  -w WSPORT, --wsport=WSPORT
                        websocket port
  -e EWSPORT, --ewsport=EWSPORT
                        external websocket port
  -d HOMEDIR, --homedir=HOMEDIR
                        home directory containing local databases config and
                        log files
  -c CONF, --configfile=CONF
                        configuration file
```

- `-H` specifies in what addresses the servers will listen, the default value is `0.0.0.0` meaning that all addresses bound to the machine will be used (127.0.0.1, 192.168.0.100, 162.154.12.35, for example).

- `-p` specifies in what port OmniDB server will listen, this is the port used in the browser's URL if OmniDB is being accessed directly. The default value is 8000.

- `-w` specifies in what port the websocket server will listen. If OmniDB is being accessed directly the websocket client will connect to this port. The default value is 25482.

- `-e` specifies in what port the websocket client (the page opened in your browser) will connect. This option is used when OmniDB is behind a load balancer and the tornado server isn't being accessed directly, in this case we must tell websocket client what port to use. If not specified the client will use the port specified in `-w`

- `-d` This parameter let's the user choose what folder will store the persistent files, such as omnidb.conf, omnidb.log, db.sqlite3 (sessions database) and omnidb.db (application database). With this option is possible to have several instances of `omnidb-server` running, each one pointing to a specific directory. It also facilitates the deployment with Docker as it enables to point OmniDB to a mounted volume.

- `-c` Points OmniDB to a specific configuration file, can be used along with `-d` to specify a storage folder but choosing a specific config file.

Configuration File

The configuration file, `omnidb.conf` by default, can be used to set all the parameters specified in the previous category and a few additional parameter related to SSL.

This file is created when OmniDB is started for the first time or when a new folder is specified with the option `-d`. If no folder is specified the default location for files is:

- Linux: `~/.omnidb/omnidb-server/`
- Windows: `User Folder/.omnidb/omnidb-server/`

Here is the default configuration file:

```
# OmniDB Server configuration file

[webserver]

# What address the webserver listens to, 0.0.0.0 listens to all addresses bound to the machine
listening_address    = 0.0.0.0

# Webserver port, if port is in use another random port will be selected
listening_port       = 8000

# Websocket port, if port is in use another random port will be selected
websocket_port       = 25482

# External Websocket port, use this parameter if OmniDB isn't directly visible by the client
# external_websocket_port = 25482

# Security parameters
# is_ssl = True requires ssl_certificate_file and ssl_key_file parameters
# This is highly recommended to protect information
is_ssl               = False
ssl_certificate_file = /path/to/cert_file
ssl_key_file         = /path/to/key_file

# Trusted origins, use this parameter if OmniDB is configured with SSL and is being accessed by another domain
csrf_trusted_origins = origin1,origin2,origin3
```

- `is_ssl`: specifies whether to run securely or not.
- `ssl_certificate_file`: path to the certificate file.
- `ssl_key_file`: path to the key file.
- `csrf_trusted_origins`: list of trusted origins. When OmniDB is started with SSL and the browser is accessing it through another domain this parameter must specifies the domain in order to properly establish communication.

Let's take a look on how to deploy `omnidb-server` in different scenarios:

Deploying OmniDB directly

In this case no load balancers or reverse proxies are used, OmniDB is accessed directly and is**extremely** recommended to start it with SSL enabled if it will be visible to the outside world.

For this scenario the user needs to specify the following parameters:

- `-H` or `listening_address`: Specify the address visible to the clients, can be a domain.
- `-p` or `listening_port`: Specify a port that will be used in the browser url: `https://mydomain.com :PORT`
- `-w` or `websocket_port`: Specify a port that will be used by javascript to connect to Tornado server directly.
- `is_ssl`: True
- `ssl_certificate_file`: /path/to/file
- `ssl_key_file`: /path/to/file
- `-e` or `external_websocket_port`: external websocket port isn't needed as `-w` will be used directly.

It is important to mention here that both ports need to visible to every client trying to access OmniDB.

Deploying OmniDB behind a reverse proxy

In this case OmniDB won't be accessed directly but through a properly configured load balancer or reverse proxy.

For this scenario a possible approach is to run `omnidb-server` listening to the local address `127.0.0.1` and without SSL, given that the balancer will handle the security part.

The following parameters are required:

- `-H` or `listening_address`: `127.0.0.1`.
- `-p` or `listening_port`: Specify a port to which the load balancer will redirect all the OmniDB server requests.
- `-w` or `listening_port`: Specify a port to which the load balancer will redirect all the Websocket server requests.
- `-e` or `external_websocket_port`: Specify a port that will be used by javascript to connect to Tornado server. Since OmniDB is behind a load balancer, a port being listened by the load balancer should be specified here and the balancer will redirect all requests to the port specified with `-w`. It is possible to specify the same port used to access OmniDB but then the load balancer needs to proxy requests to the specific server according to the URL pattern.

Consider this example of OmniDB being hosted behind Nginx:

- Starting omnidb-server:

```
omnidb-server -H 127.0.0.1 -p 9000 -w 26500 -e 443
```

In this case OmniDB can only be accessed locally and the browser will try to connect to the websocket server with the default https port (443).

- Nginx configuration file:

```
server {
    listen 443 ssl;
    listen [::]:443 ssl;
    include snippets/ssl-domain.conf;
    include snippets/ssl-params.conf;
    server_name domain.org;
    client_max_body_size 75M;

    location /wss {
        proxy_pass  http://127.0.0.1 :26500;
        proxy_set_header   X-Real-IP $remote_addr;
        proxy_set_header   X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header   X-Forwarded-Ssl https;
        proxy_set_header   X-Forwarded-Proto https;
        proxy_set_header   X-Forwarded-Port 443;
        proxy_set_header   Host $host;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection "upgrade";
    }

    location / {
        proxy_pass  http://127.0.0.1 :9000;
        proxy_set_header   X-Real-IP $remote_addr;
        proxy_set_header   X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header   X-Forwarded-Ssl https;
        proxy_set_header   X-Forwarded-Proto https;
        proxy_set_header   X-Forwarded-Port 443;
        proxy_set_header   Host $host;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection "upgrade";
    }
}
```

As can be seen, Nginx is listenning for requests to `domain.org` in port 443. Since we also specified the external websocket port to 443, websocket requests will be dealt here too.

Websocket requests are always directed to the pattern `/wss` so we use a specific location configuration to redirect all requests to the port specified with `-w`, 26500 in this case.

Other requests that are not to `/wss` should all be redirected to OmniDB server, 9000 in this case.