Mandiant

SET UP YOUR DIRECTOR TO USE AN EXTERNAL DATABASE

If necessary, you can configure an on-prem Security Validation Director to use an external database. This option allows you to scale system performance and meet IT risk requirements.

Requirements

Note the following information about the supported PostgreSQL version in relation to your Security Validation version. If your PostgreSQL version is not listed, it's not supported.

Director version	Supported PostgreSQL version	Password encryption required
4.14.6.0 and later	PostgreSQL 16	scram-sha-256
4.14.4.1 - 4.14.5.0	PostgreSQL 16	md5
4.11.1.0 - 4.14.4.0	PostgreSQL 13	md5
Prior to 4.11.1.0	PostgreSQL 11.x	md5

Limitations

- High-availability configuration is not supported. You cannot have multiple Directors using the same database.
- You cannot rely on database snapshots to replace the Director system backup/restore functionality. Directors store important files on the filesystem. The Director will not function correctly if the expected files on disk do not match the database objects.
- System backups from a Director that is using a newer PostgreSQL version cannot be imported on a Director using an older PostgreSQL version.

Configure an external database

To configure an external database, follow this two-step process:

1. **Prepare the database for migration**
2. **Migrate the Security Validation database**

Unless stated otherwise, these commands must be run as a superuser with either `sudo su-` or `sudo` prepended to commands.

Prepare the PostgreSQL database

In a standard PostgreSQL deployment, the required `template1` database should exist by default. Additionally, the database should use UTF-8 encoding.

1. To verify that the `template1` database exists, sign in to the `psql`, the terminal for PostgreSQL:

   ```
   \l
   ```

   Here's an example of a list of databases after running the `\l` command:

```
postgres=# \l
                                 List of databases
   Name    | Owner   | Encoding |   Collate   |    Ctype    |   Access privileges
-----------+----------+----------+-------------+-------------+-----------------------
 postgres  | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 |
 template0 | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 | =c/postgres          +
           |          |          |             |             | postgres=CTc/postgres
 template1 | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 | =c/postgres          +
           |          |          |             |             | postgres=CTc/postgres
(3 rows)
```

2. If the database does not exist, create it using UTF-8 encoding:

```
CREATE DATABASE "template1"
  WITH OWNER "postgres"
  ENCODING 'UTF8'
  LC_COLLATE ='en_US.UTF-8'
  LC_CTYPE ='en_US.UTF-8';
```

3. Open port 5432 on the PostgreSQL host for remote access:
   a. Update IPTables Rules on the server with PostgreSQL. Replace < *Director IP*> with your Director's IP address:

```
iptables -A INPUT -p tcp -s 0/0 --sport 1024:65535 -d <Director IP> --dport 5432 -m state --state NEW,EST
ABLISHED -j ACCEPT
iptables -A OUTPUT -p tcp -s <Director IP> --sport 5432 -d 0/0 --dport 1024:65535 -m state --state ESTABL
ISHED -j ACCEPT
```

   For example, if the Director's IP address is 10.13.13.182:

```
iptables -A INPUT -p tcp -s 0/0 --sport 1024:65535 -d 10.13.13.182 --dport 5432 -m state --state NEW,EST
ABLISHED -j ACCEPT
iptables -A OUTPUT -p tcp -s 10.13.13.182 --sport 5432 -d 0/0 --dport 1024:65535 -m state --state ESTABL
ISHED -j ACCEPT
```

   b. Restart IPTables:

```
systemctl restart iptables.service
```

   c. On Debian-based systems, run the following command:

```
~$ sudo ufw allow 5432
Rule added
Rule added (v6)
~$
```

   d. Disable and then re-enable the Uncomplicated Firewall (UFW):

```
ufw reload
ufw disable
ufw enable
```

4. Enable external access to PostgreSQL. Add either the specific address of the host for the Director machine (most secure) or `0.0.0.0` for all addresses.
   Here's an example of the `postgresql.conf` contents:

```
#------------------------------------------------------------------------------
# CONNECTIONS AND AUTHENTICATION
#------------------------------------------------------------------------------

# - Connection Settings -

listen_addresses = '0.0.0.0'          # what IP address(es) to listen on;
                                      # comma-separated list of addresses;
                                      # defaults to 'localhost'; use '*' for all
                                      # (change requires restart)
#port = 5432                          # (change requires restart)
#max_connections = 100                # (change requires restart)
#superuser_reserved_connections = 3   # (change requires restart)
#unix_socket_directories = '/tmp'     # comma-separated list of directories
                                      # (change requires restart)
#unix_socket_group = ''               # (change requires restart)
#unix_socket_permissions = 0777       # begin with 0 to use octal notation
                                      # (change requires restart)
#bonjour = off                        # advertise server via Bonjour
                                      # (change requires restart)
#bonjour_name = ''                    # defaults to the computer name
                                      # (change requires restart)
```

5. Edit the `pg_hba.conf` file. Add the line `host all all 0.0.0.0/0 scram-sha-256` to allow access from all hosts using an scram-sha-256 encrypted password. The most secure configuration to enable access will only specify the /32 address for the Director static IP address.
   Here's an example of the contents in `pg_hba.conf`:

```
local   all             all                                     scram-sha-256
host    all             all             127.0.0.1/32            scram-sha-256
host    all             all             ::1/128                 scram-sha-256
host    all             all             0.0.0.0/0               scram-sha-256
~
~
~
~
~
~
```

6. Restart the PostgreSQL service. The following example command is for Debian-based systems:

```
sudo systemctl restart postgresql
```

Migrate the database

The `vconfigdb` script (located in the `/opt/apps/verodin/planner/bin` directory) is used during the migration. Based on it's location, it should be in the PATH on a normal Director. The following section details an example of the process, including parameters used.

1. In the Director web interface, create a system back up. For more information, see **Backup and Restore Security Validation (https://docs.mandiant.com/home/msv-backing-up-and-restoring-the-platform)**.
2. Run the migration script to switch to a new database and migrate the data over:

```
vconfigdb switch_db
```

3. Follow the interactive prompts to provide the database hostname, name, and credentials. For a list of the supported arguments, run `vconfigdb help switch_db`.

   Using the name `director` is recommended for clarity.

4. Before finalizing the new database configuration, the script makes a test connection and queries the PostgreSQL version.
   - If the connection or credentials fail, or the version isn't sufficient, the operation is cancelled, and no change is made.
   - If the test is successful, then pg_dump is run against the current database to dump the data, and then it's restored into the new database.

5. When prompted, enter the database name. Any value is acceptable, but consider using `director`. The database will be created if the name isn't being used already. If another database with the name does exist, it will be replaced.
6. Confirm the database name. If there are multiple Director databases running on the same Postgres host, you should take care to name them uniquely so that the wrong one does not get dropped. To avoid resource contention, we recommend having only one Director per Postgres server.

The following code block illustrates the migration process, showing the commands and responses. The parameters that were used when running the command include:

- Hostname: `192.168.1.59`
- SSL root cert: `none`
- Database name: `security_validation`
- Username: `<username>`
- Password: `<password>`
- Database port: `5432`

  For a sample of migration output, see the following:

```
[nodeone@director ~]$ sudo vconfigdb switch_db
Stopping all services
Enter the new database hostname: 192.168.1.59
Enter the new SSL root cert location: none
Enter the new database name: security_validation
Enter the new database username: postgres
Enter the new database password:
Enter the new database port (default: 5432):

Testing database configuration
Test succeeded

Dumping the database to /opt/apps/verodin/planner/tmp/dbdump_20220206152620_planner.sql.gz
Success dumping the database

Updating database config
Backing up existing config to /opt/apps/verodin/planner/tmp/dbconfig_backup_20220206152622.yml
Config file generated: /opt/apps/verodin/planner/config/database.yml
Success writing database config

WARNING! Continuing with this will drop the existing database 'security_validation' at 192.168.1.59.
Continue? (Y/N) y

Restoring the database
Starting all services
[nodeone@director ~]$
```

Verify the new Database was created

After the migration has completed, check that the database is created by installing the `psql` tools locally and connecting to the database.

In the following code, the `\l` command shows the database list before `switch_db` is run:

```
% psql -U postgres -p 5432 -h 192.168.1.59
Password for user postgres:
psql (14.0, server 11.14)
Type "help" for help.

postgres=# show databases
postgres-# \d
Did not find any relations.
postgres-# \l
List of databases
Name | Owner | Encoding | Collate | Ctype | Access privileges
-----------+----------+----------+-------------+-------------+-----------------------
postgres | postgres | UTF8 | en_US.UTF-8 | en_US.UTF-8 |
template0 | postgres | UTF8 | en_US.UTF-8 | en_US.UTF-8 | =c/postgres +
| | | | | postgres=CTc/postgres
template1 | postgres | UTF8 | en_US.UTF-8 | en_US.UTF-8 | =c/postgres +
| | | | | postgres=CTc/postgres
(3 rows)


postgres=# \l+
List of databases
Name | Owner | Encoding | Collate | Ctype | Access privileges | Size | Tablespace | Description
--------------------+----------+----------+-------------+-------------+-----------------------+---------+------------+-------------------------------
---------
postgres | postgres | UTF8 | en_US.UTF-8 | en_US.UTF-8 | | 7685 kB | pg_default | default administrative connection da
tabase
security_validation | postgres | UTF8 | en_US.UTF-8 | en_US.UTF-8 | | 31 MB | pg_default |
template0 | postgres | UTF8 | en_US.UTF-8 | en_US.UTF-8 | =c/postgres +| 7545 kB | pg_default | unmodifiable empty
database
| | | | | postgres=CTc/postgres | | |
template1 | postgres | UTF8 | en_US.UTF-8 | en_US.UTF-8 | =c/postgres +| 7685 kB | pg_default | default template for
new databases
| | | | | postgres=CTc/postgres | | |
(4 rows)
% psql -U postgres -p 5432 -h 192.168.1.59
Password for user postgres:
psql (14.0, server 11.14)
Type "help" for help.

postgres=# show databases
postgres-# \d
Did not find any relations.
postgres-# \l
List of databases
Name | Owner | Encoding | Collate | Ctype | Access privileges
-----------+----------+----------+-------------+-------------+-----------------------
postgres | postgres | UTF8 | en_US.UTF-8 | en_US.UTF-8 |
template0 | postgres | UTF8 | en_US.UTF-8 | en_US.UTF-8 | =c/postgres +
| | | | | postgres=CTc/postgres
template1 | postgres | UTF8 | en_US.UTF-8 | en_US.UTF-8 | =c/postgres +
| | | | | postgres=CTc/postgres
(3 rows)
```

In the following code, the `\l` command shows the database list after `switch_db` is run:

```
postgres=# \l+
List of databases
Name | Owner | Encoding | Collate | Ctype | Access privileges | Size | Tablespace | Description
-------------------+----------+----------+-------------+-------------+-----------------------+---------+------------+-------------------------------------
---------
postgres | postgres | UTF8 | en_US.UTF-8 | en_US.UTF-8 | | 7685 kB | pg_default | default administrative connection da
tabase
security_validation | postgres | UTF8 | en_US.UTF-8 | en_US.UTF-8 | | 31 MB | pg_default |
template0 | postgres | UTF8 | en_US.UTF-8 | en_US.UTF-8 | =c/postgres +| 7545 kB | pg_default | unmodifiable empty
database
| | | | | postgres=CTc/postgres | | |
template1 | postgres | UTF8 | en_US.UTF-8 | en_US.UTF-8 | =c/postgres +| 7685 kB | pg_default | default template for
new databases
| | | | | postgres=CTc/postgres | | |
(4 rows)
```

Troubleshoot

- The `vconfigdb` script catches errors when possible. When critical errors happen during the switch to a new database, it attempts to recover and revert to the previous database config.
    - If errors occur during the switching process, the services may have been stopped. Run `vrestart` to ensure the reverted database config is loaded.

- If something goes wrong during the switching process and isn't caught, the default database config can be recovered manually by running `vconfigdb revert_db`. This switches back to the local Postgres running on Director. The user will be prompted for the password for the local database, which they can leave blank to use the default password.

> If any issues arise, pay attention to the console output from the script. The output logs the filepaths where previous configs and data dumps are going, which is useful if manual recovery is needed. The script also logs more detail to `verodin_configdb_log`.

SSL

You can decide whether you want to force SSL encryption for the Postgres connection. If the database is reachable only over a VPC from the Director, you might want to allow non-SSL connections to the database.

By default, `vconfigdb` prompts for the filepath to an SSL certificate that is pinned for the database connection. To opt out of SSL, enter `none` when prompted.

If using SSL encryption:

- Put the certificate file in `/opt/apps/verodin/planner` and provide the full path when asked for that file.
- If using AWS RDS, you can download their combined ca bundle from this link: **https://s3.amazonaws.com/rds-downloads/rds-combined-ca-bundle.pem**.