Component: Data Storage

Table of Contents

Overview

This page provides a detailed design for the Data Storage Component of the COPS Platform. It is based on the Decision from this Decision Log: Which MLS Database should COPS use?

Responsibilities

- Provide a secure (encrypted) location for storing data at rest
- Provide a storage location for service data
- Structure data in a well-documented way
- Allow authorized users to access only their data
- Prevent users from accessing data without proper authorization (and log access attempts)
- Adhere to Multi-Level Security requirements and guidelines
- Authenticate users with the IdAM service
- Preferably use SELinux labeling to enforce access control

Overall Strategy

The only application in this component is SE-PostgreSQL. In the future, we may migrate to the Crunchy Data MLS version of PostgreSQL, but the overall configuration should be similar.

Component Interfaces

Incoming connections on TCP Port 5432 (SQL Commands and Responses).

Outgoing connections to TCP Port 514 (Logs sent to rsyslog)

Configuration

SELinux

SELinux must be enabled and in ‘enforcing’ mode on the host.

SE-PostgreSQL

PostgreSQL 12.x and the sepgsql module (from the postgresql-contrib package) must be installed on the same CentOS 7 host that will be running the Container Runtime Component (due to the dependency on SELinux). The other components could potentially be running elsewhere, but the Container Runtime and Data Storage Components need to be on the same host.

Installation Tips

- This Ansible role may be a good reference for the PostgreSQL 12 installation: https://gitlab.skyward.cloud/cops/dev-platform/blob/develop/ansible/roles/postgresql/tasks/main.yml
- These commands (based on the above Ansible role) will download all the RPMs for offline installation. Note that it only downloads dependencies that are not already available on the system where the command is run. So try to run this on a freshly-installed system.

```
yum install -y wget
wget
https://download.postgresql.org/pub/repos/yum/reporpms/EL-7-x86_64/pgdg-
redhat-repo-latest.noarch.rpm
wget https://download.postgresql.org/pub/repos/yum/RPM-GPG-KEY-PGDG
rpm --import RPM-GPG-KEY-PGDG
yum install -y pgdg-redhat-repo-latest.noarch.rpm
mkdir pgsql
yumdownloader -y --resolve --destdir=pgsql postgresql12
postgresql12-server postgresql12-contrib python-psycopg2 postgresql-libs
libselinux-python
```

- Transfer the resulting 'pgsql' directory to an offline system, and then run something like

```
cd pgsql
rpm -i *.rpm
```

- Unfortunately, some of the dependencies may conflict with one another (usually multiple similar versions of the same libraries). So it may be necessary to delete the older versions manually before the 'rpm -i' command succeeds.

User Accounts

User Authentication in PostgreSQL does not work with any of the protocols supported by KeyCloak, so accounts must be created directly in PostgreSQL. But the reality is that only service accounts need to be created, since no users (aside from database administrators) need to interact directly with the data. Each service running inside the Container Runtime that needs access to data should be given a different service account.

The individual containers are configured with SELinux labels, so when a service connects to the database from a specific container to access specific data, SELinux should enforce the Mandatory Access Control.

> **ToDo**: Do we need more research into how to configure SELinux for that? It sounds reasonable, but I'm not sure about the low-level practical details. (It sounds a bit hand-wavy to me…but maybe those details will fall out of the low-level design for the Container Runtime, which has not been completed yet)

As a suggestion for implementation, use scram-sha-256 with the Password Authentication configuration. But there are many options available, so choose the one that is most secure and meets the need. https://www.postgresql.org/docs/12/auth-password.html

Logging and Auditing

All logs must be sent to the external Log Aggregator (currently expected to be rsyslog). Additionally, pgaudit must be installed and configured to log 'ALL' classes of statements : https://github.com/pgaudit/pgaudit/blob/master/README.md

Labeling Data

After creating a database, use the SECURITY LABEL SQL Command to label these database objects appropriately:

- database
- table
- column
- schema

The command is described here: https://www.postgresql.org/docs/current/sql-security-label.html

References

- SELinux integration module for PostgreSQL: https://www.postgresql.org/docs/current/sepgsql.html

- How to add labels to data in the database:
    - https://www.postgresql.org/docs/current/sql-security-label.html
- Client Authentication methods for PostgreSQL: https://www.postgresql.org/docs/12/client-authentication.html

Nonfunctional requirements for the software

- The installation procedure must work in an offline/disconnected environment.
- The final configuration must run on CentOS 7 with SELinux enabled (and enforcing)