Appendix B

The following are the core instructions for installing a PostgreSQL cluster with `pgaudit`.

Note: The following instructions use the PGDATA and PGVER environment variables. See supplementary content APPENDIX-F and APPENDIX-H for instructions on configuring PGDATA and PGVER.

Installing pgaudit

First, PostgreSQL must be installed. In this example, we are using Red Hat Enterprise Linux 7 RPMs. Use the appropriate RPM URL from the following webpage: https://yum.postgresql.org/repopackages.php.

```
$ sudo yum update

$ sudo yum install bison flex gcc \
    readline-devel zlib-devel perl-devel \
    perl-ExtUtils-Embed openssl-devel \
    pam-devel libxml2-devel libxslt-devel \
    libuuid-devel openldap-devel tcl-devel \
    python-devel
```

Next, install PostgreSQL:

```

$ sudo yum install https://download.postgresql.org/pub/repos/yum/reporpms/EL-
7-x86_64/pgdg-redhat-repo-latest.noarch.rpm

$ sudo yum install postgresql${PGVER/./} postgresql${PGVER/./}-contrib \
              postgresql${PGVER/./}-devel postgresql${PGVER/./}-docs \
              postgresql${PGVER/./}-libs postgresql${PGVER/./}-server
```

Now that PostgreSQL is installed, set PATH to point to the new binaries:

```
$ export PATH=/usr/pgsql-${PGVER}/bin:$PATH
$ pg_config --configure
```

With PostgreSQL installed, install pgaudit.

In this example, Git is used. The source code can be uploaded to the webserver by different means, depending on your organization's rules.

```
$ cd /usr/pgsql-${PGVER}/share/contrib/

$ sudo git clone https://github.com/pgaudit/pgaudit.git

$ cd ./pgaudit

$ sudo PATH=/usr/pgsql-${PGVER}/bin:$PATH make USE_PGXS=1 install
```

pgaudit is built and ready to be configured. First initialize the database:

```
$ sudo /usr/pgsql-${PGVER}/bin/postgresql${PGVER/./}-setup initdb
$ sudo systemctl enable postgresql-${PGVER}
```

Now as postgres user, add pgaudit to the shared_preload_libraries in postgresql.conf:

```
$ sudo su - postgres
$ vi ${PGDATA?}/postgresql.conf
```

Change shared_preload_libraries to the following:

```
shared_preload_libraries = 'pgaudit'
```

As a sudo user, start the PostgreSQL server:

```
# SERVER USING SYSTEMCTL ONLY
$ sudo systemctl start postgresql-${PGVER}

# SERVER USING INITD ONLY
$ sudo service postgresql-${PGVER} start
```

pgaudit is now installed and ready to be configured.

Configuration

`pgaudit` is configured using either the `postgresql.conf` or an included configuration file (see Optional Configuration Organization).

For a complete list of configuration parameters, see: https://github.com/pgaudit/pgaudit#settings.

Example Settings for `postgresql.conf`

```
# Enable catalog logging - default is 'on'
pgaudit.log_catalog='on'
# Specify the verbosity of log information (INFO, NOTICE, LOG, WARNING, DEBUG)
pgaudit.log_level='log'
# Log the parameters being passed
pgaudit.log_parameter='on'
# Log each relation (TABLE, VIEW, etc) mentioned in a SELECT or DML statement
pgaudit.log_relation='off'
# For every statement and substatement, log the statement and parameters every time
pgaudit.log_statement_once='off'
# Define the master role to use for object logging
# pgaudit.role=''
# Choose the statements to log:
# READ - SELECT, COPY
# WRITE - INSERT, UPDATE, DELETE, TRUNCATE, COPY
# FUNCTION - Function Calls and DO Blocks
# ROLE - GRANT, REVOKE, CREATE/ALTER/DROP ROLE
# DDL - All DDL not included in ROLE
# MISC - DISCARD, FETCH, CHECKPOINT, VACUUM
pgaudit.log='role, read, write, ddl'
```

Setting log_line_prefix

It is advisable to change `log_line_prefix in postgresql.conf` to match your auditing needs.

At a minimum, it is suggested to set the parameter to the following:

```
$ sudo su - postgres
$ vi ${PGDATA?}/postgresql.conf
log_line_prefix = '<%m %a %u %d %r %p>'
```

This will prefix all logged events with “:”

```
< 2020-01-28 19:43:12.126 UTC bob postgres: >
```