CIS Benchmarks

CIS Oracle MySQL Enterprise Edition 8.0

v1.0.0 - 04-14-2021

Terms of Use

Please see the below link for our current terms of use:

https://www.cisecurity.org/cis-securesuite/cis-securesuite-membership-terms-of-use/

Table of Contents

Overview

This document, CIS Oracle MySQL Enterprise Edition 8.0 Benchmark, provides prescriptive guidance for establishing a secure configuration posture for MySQL Enterprise Edition 8.0. This guide was tested against MySQL Enterprise Edition 8.0 running on Ubuntu Linux, but applies to other Linux distributions as well. To obtain the latest version of this guide, please visit http://benchmarks.cisecurity.org. If you have questions, comments, or have identified ways to improve this guide, please write us at feedback@cisecurity.org.

Intended Audience

This document is intended for system and application administrators, security specialists, auditors, help desk, and platform deployment personnel who plan to develop, deploy, assess, or secure solutions that incorporate Oracle MySQL Enterprise Edition 8.0.

Consensus Guidance

This benchmark was created using a consensus review process comprised of subject matter experts. Consensus participants provide perspective from a diverse set of backgrounds including consulting, software development, audit and compliance, security research, operations, government, and legal.

Each CIS benchmark undergoes two phases of consensus review. The first phase occurs during initial benchmark development. During this phase, subject matter experts convene to discuss, create, and test working drafts of the benchmark. This discussion occurs until consensus has been reached on benchmark recommendations. The second phase begins after the benchmark has been published. During this phase, all feedback provided by the Internet community is reviewed by the consensus team for incorporation in the benchmark. If you are interested in participating in the consensus process, please visit https://workbench.cisecurity.org/.

Typographical Conventions

The following typographical conventions are used throughout this guide:

Convention	Meaning
`Stylized Monospace font`	Used for blocks of code, command, and script examples. Text should be interpreted exactly as presented.
`Monospace font`	Used for inline code, commands, or examples. Text should be interpreted exactly as presented.
<*italic font in brackets*>	Italic texts set in angle brackets denote a variable requiring substitution for a real value.
Italic font	Used to denote the title of a book, article, or other publication.
Note	Additional information or caveats

Assessment Status

An assessment status is included for every recommendation. The assessment status indicates whether the given recommendation can be automated or requires manual steps to implement. Both statuses are equally important and are determined and supported as defined below:

Automated

Represents recommendations for which assessment of a technical control can be fully automated and validated to a pass/fail state. Recommendations will include the necessary information to implement automation.

Manual

Represents recommendations for which assessment of a technical control cannot be fully automated and requires all or some manual steps to validate that the configured state is set as expected. The expected state can vary depending on the environment.

Profile Definitions

The following configuration profiles are defined by this Benchmark:

- **Level 1 - MySQL RDBMS on Linux**

  Items in this profile apply to MySQL Enterprise Edition 8.0 running on Linux and intend to:

  - be practical and prudent;
  - provide a clear security benefit; and
  - not inhibit the utility of the technology beyond acceptable means.

- **Level 2 - MySQL RDBMS on Linux**

  This profile extends the "Level 1 - MySQL RDBMS on Linux" profile. Items in this profile apply to MySQL Enterprise Edition 8.0 running on Linux and exhibit one or more of the following characteristics:

  - are intended for environments or use cases where security is paramount
  - acts as defense in depth measure
  - may negatively inhibit the utility or performance of the technology.

- **Level 1 - MySQL RDBMS**

  Items in this profile apply to MySQL Enterprise Edition 8.0 and intend to:

  - be practical and prudent;
  - provide a clear security benefit; and
  - not inhibit the utility of the technology beyond acceptable means.

  Note: the intent of this profile is to include checks that can be assessed by remotely connecting to a MySQL RDBMS. Therefore, file system-related checks are not contained in this profile.

- **Level 2 - MySQL RDBMS**

  This profile extends the "Level 1 - MySQL RDBMS" profile. Items in this profile apply to MySQL Enterprise Edition 8.0 and exhibit one or more of the following characteristics:

  - are intended for environments or use cases where security is paramount
  - acts as defense in depth measure

- may negatively inhibit the utility or performance of the technology.

Note: the intent of this profile is to include checks that can be assessed by remotely connecting to a MySQL RDBMS. Therefore, file system-related checks are not contained in this profile.

Acknowledgements

This benchmark exemplifies the great things a community of users, vendors, and subject matter experts can accomplish through consensus collaboration. The CIS community thanks the entire consensus team with special recognition to the following individuals who contributed greatly to the creation of this guide:

Author
Michael Frank

Editor
Tim Harrison, Center for Internet Security
Robin Drake

Recommendations

1 Operating System Level Configuration

This section contains recommendations related to the Operating System on which the MySQL database server is running.

1.1 Place Databases on Non-System Partitions (Manual)

Profile Applicability:

- Level 1 - MySQL RDBMS on Linux

Description:

It is generally accepted that host operating systems should include different filesystem partitions for different purposes. One set of filesystems is typically called system partitions, and these are generally reserved for host system/application operation. The other set of filesystems is typically called "non-system partitions", and such locations are generally reserved for storing data.

Rationale:

Moving the database off the system partition will reduce the probability of denial of service caused by exhaustion of available disk space to the operating system.

Impact:

Moving database files and directories to a non-system partition may be difficult depending on whether there was only a single partition when the operating system was set up and whether there are additional non-system partitions available.

Audit:

Execute the following steps to assess this recommendation:

- Obtain the location of the `datadir` and other MySQL database files by executing the following SQL statement

```
SELECT VARIABLE_NAME, VARIABLE_VALUE
FROM performance_schema.global_variables
WHERE (VARIABLE_NAME LIKE '%dir' or VARIABLE_NAME LIKE '%file') and
(VARIABLE_NAME NOT LIKE '%core%'
      AND VARIABLE_NAME <> 'local_infile' AND VARIABLE_NAME <>
'relay_log_info_file') order by
      VARIABLE_NAME;
```

- Using the value returned for the `datadir`, and other results from the above query, execute the following in a system terminal

```
df -h <directory>
```

The output returned from the `df` command above should not include root (`/`), `/var`, or `/usr`.

Remediation:

Perform the following steps to remediate this setting for the `datadir`:

1. Backup the database.
2. Choose a non-system partition `new location` for MySQL data.
3. Stop `mysqld` using a command like: `service mysql stop`.
4. Copy the data using a command like: `cp -rp<datadir Value> <new location>`.
5. Set the `datadir` location to the `new location` in the MySQL configuration file.
6. Start mysqld using a command like: service mysql start.
   Note: On some Linux distributions you may need to additionally modify `apparmor` settings. For example, on a Ubuntu 14.04.1 system edit the file `/etc/apparmor.d/usr.sbin.mysqld` so that the `datadir` access is appropriate. The original might look like this:

```
# Allow data dir access
/var/lib/mysql/ r,
/var/lib/mysql/** rwk,
```

Alter those two paths to be the new location you chose above. For example, if that new location were `/media/mysql`, then the `/etc/apparmor.d/usr.sbin.mysqld` file should include something like this:

```
# Allow data dir access
/media/mysql/ r,
/media/mysql/** rwk,
```

Default Value:

Not Applicable.

References:

1. https://dev.mysql.com/doc/mysql-secure-deployment-guide/8.0/en/secure-deployment-permissions.html

CIS Controls:

Version 7

2.10 Physically or Logically Segregate High Risk Applications

Physically or logically segregated systems should be used to isolate and run software that is required for business operations but incur higher risk for the organization.

1.2 Use Dedicated Least Privileged Account for MySQL Daemon/Service (Automated)

Profile Applicability:

- Level 1 - MySQL RDBMS on Linux

Description:

As with any service installed on a host, it can be provided with its own user context. Providing a dedicated user to the service provides the ability to precisely constrain the service within the larger host context.

Rationale:

Utilizing a least privilege account for MySQL to execute as needed may reduce the impact of a MySQL-born vulnerability. A restricted account will be unable to access resources unrelated to MySQL, such as operating system configurations.

Audit:

Execute the following command at a terminal prompt to assess this recommendation:

```
ps -ef | egrep "^mysql.*$"
```

If no lines are returned, then this is a fail.

Note: It is assumed that the MySQL user is `mysql`. Additionally, you may consider running `sudo -l` as the MySQL user or to check the sudoers file.

Remediation:

Create a user which is only used for running MySQL and directly related processes. This user must not have administrative rights to the system. Additionally, its best to avoid providing shell access to such an account.

Shell access can be removed using the following command at a terminal prompt:

```
/usr/sbin/groupadd -g 27 -o -r mysql >/dev/null 2>&1 || :
/usr/sbin/useradd -M -N -g mysql -o -r -d /var/lib/mysql -s /bin/false \
    -c "MySQL Server" -u 27 mysql >/dev/null 2>&1 || :
```

References:

1. https://dev.mysql.com/doc/refman/8.0/en/changing-mysql-user.html

2. https://dev.mysql.com/doc/refman/8.0/en/server-options.html#option_mysqld_user

Additional Information:

The root user may be used to start the MySQL service on Linux/UNIX, but then it must be configured to drop privileges by specifying a service specific user in the my.cnf or my.ini file.

CIS Controls:

Version 7

14 Controlled Access Based on the Need to Know
Controlled Access Based on the Need to Know

1.3 Disable MySQL Command History (Automated)

Profile Applicability:

- Level 2 - MySQL RDBMS on Linux

Description:

On Linux/UNIX, the MySQL client and MySQL Shell log statements executed interactively to a history file. The default MySQL Client file is named `.mysql_history` in the user's home directory. The files are split by language and named `history.sql`, `history.js` and `history.py`. Most interactive commands run in the MySQL client application are saved to a history file. The MySQL command history should be disabled. By default, the MySQL Shell does not save history between sessions.

Rationale:

Disabling the MySQL Client and MySQL Shell command history reduces the probability of exposing sensitive information, such as passwords, encryption keys, or other sensitive data or information.

Audit:

Execute the following commands to assess this recommendation:

```
find /home -name ".mysql_history"
find /root -name ".mysql_history"
```

For MySQL Shell

```
ls -d .??*/* | egrep history | grep mysql
```

For each file returned determine whether that file is symbolically linked to `/dev/null`.

Remediation:

For MySQL Client perform the following steps to remediate this setting:

1. Remove `.mysql_history` if it exists.
2. Use either of the techniques below to prevent it from being created again:
    - Set the `MYSQL_HISTFILE` environment variable to `/dev/null`. This will need to be placed in the shell's startup script.
    - Create `$HOME/.mysql_history` as a symbolic to `/dev/null`.

    ```
    > ln -s /dev/null $HOME/.mysql_history
    ```

Additionally, another way to prevent history from being recorded is to use `--batch` option. For MySQL Shell perform the following steps to remediate this setting:

1. Remove `$HOME/.mysqlsh/history.*` if files exists.
2. Use either of the techniques below to prevent it from being created again:
   - Start shell and list show options using `\option -l`
   - Set to no history using the command `\option --persist history.autoSave=1`

Default Value:

By default, the MySQL command history file is located in `$HOME/.mysql_history`.

References:

1. https://dev.mysql.com/doc/refman/8.0/en/mysql-logging.html
2. https://bugs.mysql.com/bug.php?id=72158
3. https://dev.mysql.com/doc/mysql-shell/8.0/en/mysql-shell-working-with-history.html

CIS Controls:

Version 7

13.2 Remove Sensitive Data or Systems Not Regularly Accessed by Organization
Remove sensitive data or systems not regularly accessed by the organization from the network. These systems shall only be used as stand alone systems (disconnected from the network) by the business unit needing to occasionally use the system or completely virtualized and powered off until needed.

1.4 Verify That the MYSQL_PWD Environment Variables is Not in Use (Automated)

Profile Applicability:

- Level 1 - MySQL RDBMS on Linux

Description:

MySQL can read a default database password from an environment variable called `MYSQL_PWD`. Avoiding use of this environment variable can better safeguard the confidentiality of MySQL credentials.

Rationale:

Using the `MYSQL_PWD` environment variable implies MySQL credentials are stored as clear text.

Audit:

To assess this recommendation, use the `/proc` filesystem to determine if `MYSQL_PWD` is currently set for any process

```
grep MYSQL_PWD /proc/*/environ
```

This may return one entry for the process which is executing the grep command.

Remediation:

Check which users and/or scripts are setting `MYSQL_PWD` and change them to use a more secure method.
For unattended logins you should consider

1. MySQL Configuration Editor
2. Different authentication methods like e.g., X509 certificate verification,
3. Use MySQL Enterprise LDAP plugin with Kerberos or SASL tokens.

Default Value:

Not set.

References:

1. https://dev.mysql.com/doc/refman/8.0/en/environment-variables.html

2. https://dev.mysql.com/doc/refman/8.0/en/mysql-config-editor.html
3. https://dev.mysql.com/doc/refman/8.0/en/pluggable-authentication.html

CIS Controls:

Version 7

16.4 Encrypt or Hash all Authentication Credentials
Encrypt or hash with a salt all authentication credentials when stored.

1.5 Ensure Interactive Login is Disabled (Automated)

Profile Applicability:

- Level 2 - MySQL RDBMS on Linux

Description:

When created, the MySQL user may have interactive access to the operating system, which means that the MySQL user could login to the host as any other user would.

Rationale:

Preventing the MySQL user from logging in interactively may reduce the impact of a compromised MySQL account. There is also more accountability, as accessing the operating system where the MySQL server lies will require the user's own account. Interactive access by the MySQL user is unnecessary and should be disabled.

Impact:

This setting will prevent the MySQL administrator from interactively logging into the operating system using the MySQL user. Instead, the administrator will need to log in using one's own account.

Audit:

Execute the following command to assess this recommendation:

```
getent passwd mysql | egrep "^.*[\/bin\/false|\/sbin\/nologin]$"
```

Lack of output implies a fail.

Remediation:

Execute one of the following commands in a terminal:

```
usermod -s /bin/false mysql
```

Or

```
usermod -s /sbin/nologin mysql
```

CIS Controls:

Version 7

4.3 Ensure the Use of Dedicated Administrative Accounts
Ensure that all users with administrative account access use a dedicated or secondary account for elevated activities. This account should only be used for administrative activities and not internet browsing, email, or similar activities.

1.6 Verify That 'MYSQL_PWD' is Not Set in Users' Profiles (Automated)

Profile Applicability:

- Level 1 - MySQL RDBMS on Linux

Description:

MySQL can read a default database password from an environment variable called `MYSQL_PWD`.

Rationale:

Use of the `MYSQL_PWD` environment variable implies MySQL credentials are stored as clear text. Avoiding use of this environment variable may increase assurance that the confidentiality of MySQL credentials is preserved.

Audit:

To assess this recommendation, check if `MYSQL_PWD` is set in login scripts using the following command:

```
grep MYSQL_PWD /home/*/.{bashrc,profile,bash_profile}
```

Remediation:

Check which users and/or scripts are setting `MYSQL_PWD` and change them to use a more secure method.

Default Value:

Not set.

References:

1. https://dev.mysql.com/doc/refman/8.0/en/environment-variables.html

CIS Controls:

Version 7

16.4 Encrypt or Hash all Authentication Credentials
Encrypt or hash with a salt all authentication credentials when stored.

2 Installation and Planning

This section contains important considerations when deploying MySQL services to your production network and defining the configuration. The recommendations made herein are not scored from a benchmark perspective and generally align with best current practices as conveyed in most control frameworks.

The first consideration is related to the configuration options via the MySQL configuration file (e.g., `my.cnf`) and placing options under the proper section of `[mysqld]`. Options placed in the `my.cnf` configuration file should not prefix with a double dash (--). On Linux systems, `my.cnf` is located in the `/etc/` directory.

The second consideration is for an administrator to connect to a MySQL instance and change or add to the configuration options using the `SET PERSIST` command. This persists system variables in `mysqld-auto.cnf` which is located in the MySQL `datadir` by default. The file permissions on `mysqld-auto.cnf` are by default more restrictive than `my.cnf` (no world permissions).

Finally, configuration options can also be placed on the command line by modifying the MySQL startup script. The startup script is system dependent and based on your operating system.

2.1 Backup and Disaster Recovery

This section contains recommendations related to backup and recovery.

2.1.1 Backup Policy in Place (Manual)

Profile Applicability:

- Level 1 - MySQL RDBMS on Linux

Description:

A backup policy should be in place.

Rationale:

Backing up MySQL databases, including `mysql`, will help ensure the availability of data in the event of an incident.

Impact:

Without backups, it might be hard to recover from an incident.

Audit:

Check with `crontab -l` if there is a backup schedule.

Remediation:

Create a backup policy and backup schedule.

CIS Controls:

Version 7

10.1 Ensure Regular Automated Back Ups
Ensure that all system data is automatically backed up on regular basis.

2.1.2 Verify Backups are Good (Manual)

Profile Applicability:

- Level 1 - MySQL RDBMS on Linux

Description:

Backups should be validated on a regular basis.

Rationale:

Verifying that backups are occurring appropriately will help ensure data availability in the event of an incident.

Impact:

Without a well-tested backup, it might be hard to recover from an incident if the backup procedure contains errors or doesn't include all required data.

Audit:

Check reports of backup validation tests.

Remediation:

Implement regular backup checks and document each check.

CIS Controls:

Version 7

10.3 Test Data on Backup Media
Test data integrity on backup media on a regular basis by performing a data restoration process to ensure that the backup is properly working.

2.1.3 Secure Backup Credentials (Manual)

Profile Applicability:

- Level 1 - MySQL RDBMS on Linux

Description:

The password, certificate, and any other credentials should be protected.

Rationale:

A database user with the least amount of privileges required to perform backup is needed. The credentials for this user should be protected.

Impact:

When the backup credentials are not properly secured, then they might be abused to gain access to the server. The backup user needs an account with many privileges, so an attacker might potentially gain (almost) complete access to the server.

Audit:

Check permissions of files containing passwords and/or SSL keys.

Remediation:

Change file permissions.

CIS Controls:

Version 7

10.4 Ensure Protection of Backups
Ensure that backups are properly protected via physical security or encryption when they are stored, as well as when they are moved across the network. This includes remote backups and cloud services.

2.1.4 The Backups Should be Properly Secured (Manual)

Profile Applicability:

- Level 1 - MySQL RDBMS on Linux

Description:

The backup files will contain all data in the databases. Filesystem permissions and/or encryption should be used to prevent unauthorized users from gaining access to the backups.

Rationale:

Backups should be considered sensitive information.

Impact:

If an unauthorized user can access backups, then they have access to all data in the database. This is true for unencrypted backups and for encrypted backups if the encryption key is stored along with the backup.

Audit:

Check who has access to the backup files.

- Are the files world-readable (e.g., `rw-r--r-`)
  - Are they stored in a world readable directory?
- Is the group MySQL and/or backup specific?
  - If not: the file and directory must not be group readable
- Are the backups stored offsite?
  - Who has access to the backups?
- Are the backups encrypted?
  - Where is the encryption key stored?
  - Does the encryption key consist of a guessable password?

If you are running the MySQL Enterprise Backup verify that the backup uses `--encrypt`. For example:

```
$ mysqlbackup --defaults-file=/home/dbadmin/my.cnf --backup-
image=/home/admin/backups/my.mbi \
  --backup-dir=/home/admin/backup-tmp --encrypt-password backup-to-image
```

If `--encrypt-password` is not included the backup is not encrypted and this is a fail.

`Mysqlbackup` includes encryption, secure backup of keys, and support for secured archival storage.

Remediation:

Implement encryption, properly restrict filesystem permissions, protect and backup encryption keys.

For example, if you run MySQL Enterprise Backup include `--encrypt-password`

```
$ mysqlbackup --defaults-file=/home/dbadmin/my.cnf --backup-
image=/home/admin/backups/my.mbi \
  --backup-dir=/home/admin/backup-tmp --encrypt-password backup-to-image
```

`Mysqlbackup` includes not just the database data, but also provides for secure backup of keys, and support for secured archival storage.

References:

1. https://dev.mysql.com/doc/mysql-enterprise-backup/8.0/en/meb-encrypted-innodb.html

CIS Controls:

Version 7

10.4 Ensure Protection of Backups
Ensure that backups are properly protected via physical security or encryption when they are stored, as well as when they are moved across the network. This includes remote backups and cloud services.

2.1.5 Point-in-Time Recovery (Automated)

Profile Applicability:

- Level 2 - MySQL RDBMS on Linux

Description:

With binlogs it is possible to implement point-in-time recovery. This makes it possible to restore the changes between the last full backup and the point-in-time.

Enabling binlogs is not sufficient. The binlogs need to be backed up to separate media. The restore procedure should be created and tested. The data in the binlog files may contain sensitive information which needs be stored in the proper location with restrictive permissions and may require encryption. Binlogs can grow quite large and take up a large amount of space so auto remove needs to be put into place.

Rationale:

Using binlogs can reduce the amount of information lost when recovering from a backup.

Impact:

Without point-in-time recovery, any data which was stored between the last backup and the time of a disaster might not be recoverable.

Audit:

Check if binlogs are enabled and if there is a restore procedure. Check to see if `--binlog-expire-logs-second` is set.

```
SELECT VARIABLE_NAME, VARIABLE_VALUE, 'BINLOG - Log Expiration' as Note
FROM performance_schema.global_variables where variable_name =
'binlog_expire_logs_seconds';
```

Ensure this value is not set to `0`.
Note: Consider implementing MySQL Enterprise Backup which includes support for at rest encryption of any MySQL Encrypted data files including the binary and relay log files.

Remediation:

Enable binlogs, then create and test a restore procedure.

Default Value:

The default for `binlog-expire-logs-second` is `2592000` seconds, or 30 days.

References:

1. https://dev.mysql.com/doc/refman/8.0/en/replication-options-binary-log.html#sysvar_binlog_expire_logs_seconds
2. https://dev.mysql.com/doc/refman/8.0/en/point-in-time-recovery-binlog.html

CIS Controls:

Version 7

10.2 Perform Complete System Backups
Ensure that each of the organization's key systems are backed up as a complete system, through processes such as imaging, to enable the quick recovery of an entire system.

2.1.6 Disaster Recovery (DR) Plan (Manual)

Profile Applicability:

- Level 1 - MySQL RDBMS on Linux

Description:

A disaster recovery plan should be created.

MySQL Cluster (group replication), MySQL Replica Sets (asynchronous replication) or both may be used.

A replica in a different data center and offsite backups may be used. There should be information regarding the Recovery Time Objective (RTO), i.e., how long recovery will take, and if the recovery site has the same capacity. Additionally, delayed replicas can be a valuable part of a DR plan. Network (default) and at rest encryption should be used to protect data.

Rationale:

A disaster recovery strategy should be planned and formalized.

Impact:

Without a well-tested disaster recovery plan, it might not be possible to recover in time.

Audit:

Check if there is a disaster recovery plan.

Remediation:

Create a disaster recovery plan.

References:

1. https://dev.mysql.com/doc/refman/8.0/en/group-replication-security.html
2. https://dev.mysql.com/doc/refman/8.0/en/replication-security.html

CIS Controls:

Version 7

10 Data Recovery Capabilities
Data Recovery Capabilities

2.1.7 Backup of Configuration and Related Files (Manual)

Profile Applicability:

- Level 1 - MySQL RDBMS on Linux

Description:

It is important to include configuration, log, key, certificates, and customized files in backups.

Rationale:

Including all configuration, log, key, certificates, and customized files in any backup will ensure the backup can fully restore an instance.

Audit:

Check if these files are in use and are saved in the backup.

- Edited Configuration files (`my.cnf` and included files)
- `SET PERSIST` Configuration file (`mysqld-auto.cnf`)
- Files related to Key Management and Keyring (KMIP, other Key Management Services)
- Audit Log Files (if not handled by other methods)
- SSL files (certificates, keys)
- User Defined Functions (UDFs)
- Source code for customizations

Remediation:

Add any omitted files to the backup.

CIS Controls:

Version 7

10.2 Perform Complete System Backups
Ensure that each of the organization's key systems are backed up as a complete system, through processes such as imaging, to enable the quick recovery of an entire system.

2.2 Data Encryption

This section contains recommendations for securing data at rest and in transit for MySQL.

2.2.1 Ensure Binary and Relay Logs are Encrypted (Automated)

Profile Applicability:

- Level 2 - MySQL RDBMS on Linux

Description:

The `binlog_encryption` system variable may be used to configure encryption of the binary and relay logs. This may be configured to `ON` even if binary logging is not enabled in order to encrypt relay log files.

Rationale:

The database, and thus the binary and relay logs, may contain sensitive information. Encrypting the binary and relay logs protects all data stored in these logs from internal and external threats.

Audit:

To audit this setting, run the following command:

```
SELECT VARIABLE_NAME, VARIABLE_VALUE, 'BINLOG - At Rest Encryption' as Note
FROM performance_schema.global_variables where variable_name =
'binlog_encryption';
```

Ensure it is set to `ON`.

Remediation:

To remediate misconfiguration, run this command:

```
SET GLOBAL binlog_encryption=ON;
```

If you receive the error message below, you need to install keyring. For instructions see [Section 6.4.4, "The MySQL Keyring"]() in the MySQL documentation.

```
ERROR 3794 (HY000): Unable to recover binlog encryption master key, please
check if keyring plugin is loaded.
```

Default Value:

The default for `binlog_encryption` is `OFF`.

References:

1. https://dev.mysql.com/doc/refman/8.0/en/replication-binlog-encryption.html
2. https://dev.mysql.com/doc/mysql-enterprise-backup/8.0/en/advanced.encrypted-binlog-relaylog.html
3. https://dev.mysql.com/doc/refman/8.0/en/replication-options-binary-log.html#sysvar_binlog_encryption

Additional Information:

It is necessary to install a keyring plugin prior to configuring encryption.

Consider implementing MySQL Enterprise Backup which includes support for at rest encryption of any MySQL Encrypted data files including the binary and relay log files.

CIS Controls:

Version 7

14.8 Encrypt Sensitive Information at Rest
Encrypt all sensitive information at rest using a tool that requires a secondary authentication mechanism not integrated into the operating system, in order to access the information.

2.3 Dedicate the Machine Running MySQL (Manual)

Profile Applicability:

- Level 1 - MySQL RDBMS on Linux

Description:

It is recommended that MySQL Server software be installed on a dedicated server. This architectural consideration affords flexibility in that the database server can be placed on a separate zone allowing access only from particular hosts and over particular protocols.

Rationale:

The attack surface is reduced on a server with only the underlying operating system, MySQL server software, and any security or operational tooling that may be additionally installed. A smaller attack surface reduces the probability of the data within MySQL being compromised.

Impact:

Care must be taken that applications or services that are required for proper operation of the operating system are not removed.

Custom applications may need to be modified to accommodate database connections over the network rather than on the use (e.g., using TCP/IP connections).

Additional hardware and operating system licenses may be required to make the architectural change.

Audit:

Verify there are no other roles enabled for the underlying operating system and that no additional applications or services unrelated to the proper operation of the MySQL server software are installed.

Remediation:

Remove excess applications or services and/or remove unnecessary roles from the underlying operating system.

CIS Controls:

Version 7

2.10 Physically or Logically Segregate High Risk Applications
Physically or logically segregated systems should be used to isolate and run software that is required for business operations but incur higher risk for the organization.

2.4 Do Not Specify Passwords in the Command Line (Manual)

Profile Applicability:

- Level 1 - MySQL RDBMS on Linux

Description:

When a command is executed on the command line, for example `mysql -u admin -p password` or `mysqlsh -u admin -p password`, the password may be visible in the user's shell/command history or in the process list.

Rationale:

If the password is visible in the process list or user's shell/command history, an attacker will be able to access the MySQL database using the stolen credentials.

Impact:

Depending on the remediation chosen, additional steps may need to be undertaken like:

- Entering a password when prompted;
- Ensuring the file permissions on `.my.cnf` is restricted yet accessible by the user;
- Using `mysql_config_editor` to encrypt the authentication credentials in `.mylogin.cnf`.
- Use a pluggable secure password store, e.g., a keychain.
- In the case of shell don't authenticate until `mysqlsh` is started, then use `\connect`

Additionally, not all scripts/applications may be able to use `.mylogin.cnf`.

Audit:

Check the process or task list if the password is visible.
Check the shell or command history if the password is visible.

Remediation:

MySQL Client:

Use `-p` without password and then enter the password when prompted, use a properly secured `.my.cnf` file, or store authentication information in encrypted format in `.mylogin.cnf`.

MySQL Shell:

Use without password and then enter the password when prompted, store authentication information in encrypted format in `.mylogin.cnf`, enter shell then authenticate using `\connect` command (**Note:** this also ensures the username is not exposed on the command), or use `mysqlsh` pluggable password store, e.g., a keychain.

References:

1. https://dev.mysql.com/doc/refman/8.0/en/mysql-config-editor.html
2. https://dev.mysql.com/doc/refman/8.0/en/password-security-user.html
3. https://dev.mysql.com/doc/mysql-shell/8.0/en/mysql-shell-pluggable-password-store.html

CIS Controls:

Version 7

16.4 Encrypt or Hash all Authentication Credentials
Encrypt or hash with a salt all authentication credentials when stored.

2.5 Do Not Reuse Usernames (Manual)

Profile Applicability:

- Level 1 - MySQL RDBMS on Linux

Description:

Database user accounts should not be reused for multiple applications or users.

Rationale:

Utilizing unique database accounts across applications will reduce the impact of a compromised MySQL account.

Impact:

If a user is reused, then a compromise of this user will compromise multiple parts of the system and/or application.

Audit:

Each user (excluding mysql reserved users) should be linked to one of these

- system accounts
- a person
- an application

To list users (and exclude mysql reserved users)

```
SELECT host, user, plugin,
  IF(plugin = 'mysql_native_password',
 'WEAK SHA1', 'STRONG SHA2') AS HASHTYPE
FROM mysql.user WHERE user NOT IN
  ('mysql.infoschema', 'mysql.session', 'mysql.sys') AND
  plugin NOT LIKE 'auth%' AND plugin <> 'mysql_no_login' AND
  LENGTH(authentication_string) > 0
ORDER BY plugin;
```

Remediation:

Add/Remove users so that each user is only used for one specific purpose.

CIS Controls:

Version 7

4.3 Ensure the Use of Dedicated Administrative Accounts
Ensure that all users with administrative account access use a dedicated or secondary account for elevated activities. This account should only be used for administrative activities and not internet browsing, email, or similar activities.

2.6 Ensure Non-Default, Unique Cryptographic Material is in Use (Manual)

Profile Applicability:

- Level 1 - MySQL RDBMS on Linux

Description:

The cryptographic material used by MySQL, such as digital certificates and encryption keys, should be used only for MySQL and only for one instance. Default cryptographic material should not be used since it is not unique to the instance.

Rationale:

If an attacker gains access to shared cryptographic material, including default material, the attacker can reuse that material to impersonate the MySQL server or otherwise compromise its operations.

Impact:

If a cryptographic material is used on multiple MySQL instances and/or systems, then a compromise of one may lead to the network traffic of all servers being compromised that use the same cryptographic material.

Audit:

Review all cryptographic material. If it is default, used for other MySQL instances and/or for purposes other than MySQL then this is a finding.
Review the server certificate by running

```
cd <data_dir and/or ssl_cert>
sudo openssl x509 -in server-cert.pem -subject -noout | grep
Auto_Generated_Server_Certificate
```

The output for the auto generated pem will look something like:

```
subject= /CN=MySQL_Server_8.0.21_Auto_Generated_Server_Certificate
```

If no rows return, the check is a pass since the certificate is not MySQL auto-generated.

Remediation:

Generate new certificates, keys, and other cryptographic material as needed for each affected MySQL instance.

References:

1. https://dev.mysql.com/doc/refman/8.0/en/using-encrypted-connections.html

2.7 Ensure 'password_lifetime' is Less Than or Equal to '365' (Automated)

Profile Applicability:

- Level 1 - MySQL RDBMS
- Level 1 - MySQL RDBMS on Linux

Description:

Password expiration provides users with a unique time bounded password lifetime.

Rationale:

Allows additional security factors pertinent to a specific user to provide further password security; predetermined by varying security needs and usability requirements in a system or organization.

Audit:

The global password lifetime is set using `default_password_lifetime`. If the value of `default_password_lifetime` is greater than `0`, it indicates the permitted password lifetime. Execute the following command to check the global password lifetime:

```
SELECT VARIABLE_NAME, VARIABLE_VALUE
FROM performance_schema.global_variables where VARIABLE_NAME like
'default_password_lifetime';
```

A value greater than or equal to `365` implies a finding.

When the global password lifetime is less than `365`, or not configured, each user account shall be checked by executing the following command:

```
SELECT user, host, password_lifetime from mysql.user where password_lifetime
= 0 OR password_lifetime >= 365;
```

A lack of results implies compliance.

Note: A value of `0` implies the password never expires.

Remediation:

To configure the global password lifetime to `365` by executing the following command:

```
set persist default_password_lifetime = 365;
```

Alternatively, configure the password lifetime for each user returned by the audit procedure by executing the following command:

```
ALTER USER '<username>'@'<localhost>' PASSWORD EXPIRE INTERVAL 365 DAY;
```

Default Value:

`NULL`

References:

1. https://csrc.nist.gov/csrc/media/publications/sp/800-118/archive/2009-04-21/documents/draft-sp800-118.pdf
2. https://dev.mysql.com/doc/refman/8.0/en/validate-password.html

Additional Information:

When a user's `password_lifetime` is set to `NULL` it takes on the value set in global `default_password_lifetime` variable.

If this recommendation becomes Scored, this will be moved to Section 7 or its equivalent in a future release.

2.8 Ensure Password Complexity is Configured (Automated)

Profile Applicability:

- Level 1 - MySQL RDBMS

- Level 1 - MySQL RDBMS on Linux

Description:

Passwords that are too complex in nature make it harder for users to remember, leading to bad practices. In addition, composition requirements provide no defense against common attack types such as social engineering or insecure password storage. In keeping with the overall goal of having users create a password that is not overly weak, it's best to have at least 14 characters for a password only account.

Rationale:

Malicious actors regularly attempt to compromise databases by attacking or guessing passwords. Stolen credentials may be used to gain access to steal information, engage in financial fraud, and more.

By enforcing practical and secure policies, end user cooperation grows. In general, longer passwords are better (harder to crack), but a forced password length requirement can cause user behavior that is predictable and undesirable. Having a reasonable minimum length with no maximum character limit increases the resulting average password length used and thus increases the security of that password.

Impact:

Enforcing too much complexity or length may be difficult for users to memorize. This may cause users to use predictable patterns or other bad practices, resulting in weaker passwords.

Audit:

Determine if the `component_validate_password` component is installed.

```
SELECT component_urn from mysql.component
WHERE component_urn='file://component_validate_password' group by
component_urn;
```

Results imply compliance.

Inspect the password policy settings.

```
SELECT VARIABLE_NAME, VARIABLE_VALUE
FROM performance_schema.global_variables
WHERE VARIABLE_NAME like 'valid%password%';
```

Compare the results to the organizationally defined policy.

Remediation:

If not already installed, install the password policy component:

```
INSTALL COMPONENT 'file://component_validate_password';
```

Set password policies in accordance with the organizationally defined policy and security best practices:

```
set persist validate_password.check_user_name='ON';
set persist validate_password.dictionary_file='<FILENAME OF DICTIONARY
FILE>';
set persist validate_password.length=14;
```

Use with care. Passwords that are too complex in nature make it harder for users to remember, leading to bad practices.

```
set persist validate_password.mixed_case_count=1;
set persist validate_password.special_char_count=1;
set persist validate_password.number_count=1;
```

Default Value:

The MySQL validate password complexity component is not installed by default.

References:

1. https://dev.mysql.com/doc/refman/8.0/en/validate-password.html

2.9 Ensure Password Resets Require Strong Passwords (Automated)

Profile Applicability:

- Level 1 - MySQL RDBMS
- Level 1 - MySQL RDBMS on Linux

Description:

Disabling password reuse, enforcing password strength, and denying reuse can be implemented to prevent successful usage of stolen or previously guessed passwords by malicious users.

Restricted accounts using passwords on the basis of the number of password changes and length ensure a password cannot be chosen from a specified number of the most recent passwords.

Rationale:

Repeated use of old passwords can increase risk of a compromise. This may lead to access by malicious users who have discovered a user's prior password(s).

Audit:

To assess this recommendation run the following statement:

```
SELECT VARIABLE_NAME, VARIABLE_VALUE
FROM performance_schema.global_variables where VARIABLE_NAME in
('password_history', 'password_reuse_interval');
```

A value of `0` indicates that NO policy is defined for the associated variable.

The `password_history` variable indicates the number of subsequent account password changes that must occur before the password can be reused. The `password_history` should be greater than or equal to `5`, thus attempts to use any of the prior five, or more, passwords for this account will be denied.

The `password_reuse_interval` defines the global policy for controlling reuse of previous passwords based on time elapsed. For an account password used previously, this variable indicates the number of days that must pass before the password can be reused. Password should not be reused over the period of a year. The value of `password_reuse_interval` should be greater than or equal to `365`.

Remediation:

Set a global policy that passwords may not be reused for a minimum of five password changes:

```
SET PERSIST password_history = 5;
```

Set a global policy that passwords have a lifetime to approximately one year (in days)

```
SET PERSIST password_reuse_interval = 365;
```

Default Value:

Both `password_history` and `password_reuse_interval` are `0` (off) by default.

References:

1. https://dev.mysql.com/doc/refman/8.0/en/password-management.html#password-reuse-policy

2.10 Require Current Password for Password Reset (Automated)

Profile Applicability:

- Level 2 - MySQL RDBMS
- Level 2 - MySQL RDBMS on Linux

Description:

Require the current password for password reset.

Rationale:

Requiring a prior password for password reset enables DBAs to prevent users from changing a password without proving that they know the current password. Such changes could otherwise occur, for example, if one user walks away from a terminal session temporarily without logging out, and a malicious user uses the session to change the original user's MySQL password. This can have unfortunate consequences; the most problematic being the malicious user can access MySQL with the user's changed credentials.

Audit:

Run the following statement:

```
SELECT VARIABLE_NAME, VARIABLE_VALUE
FROM performance_schema.global_variables where VARIABLE_NAME in
('password_require_current');
```

The return value should be `ON`.

Remediation:

Set the value to `ON`

```
SET PERSIST password_require_current=ON;
```

Default Value:

The `password_require_current` is `OFF` my default.

References:

1. https://dev.mysql.com/doc/refman/8.0/en/server-system-variables.html#sysvar_password_require_current

2.11 Use Dual Passwords to Enable Higher Frequency Password Rotation (Manual)

Profile Applicability:

- Level 2 - MySQL RDBMS
- Level 2 - MySQL RDBMS on Linux

Description:

Dual passwords act as a tool to encourage password rotations in cases where a synchronized password change is not viable. By having a time delta to have old and new passwords in place, the process of replacing old passwords with new passwords within applications is simplified.

Rationale:

Too often passwords used by applications are not changed regularly because of the difficulty in timing for propagating the new password, keeping the applications connected, and connection failures due to race conditions. If it is difficult to perform a synchronized change you can optionally use dual passwords to simplify the task of password rotation.

Impact:

If the original password isn't removed upon completion of the password rotation process, the potential risk for a compromise is increased.

Audit:

To determine which users currently have dual passwords

```
SELECT user, host FROM mysql.user WHERE length(user_attributes-
>"$.additional_password")>0;
```

If an account has a dual password and the process of password rotation has completed, this is a fail.

Remediation:

To set dual passwords execute the following `ALTER` command:

```
ALTER USER '<user>'@'<hostname>'
  IDENTIFIED BY '<new_password>'
  RETRAIN CURRENT PASSWORD;
```

Once the new password has been distributed `DISCARD` the old password using `ALTER`:

```
ALTER USER '<user>'@'<hostname>'
  DISCARD OLD PASSWORD;
```

References:

1. https://dev.mysql.com/doc/refman/8.0/en/password-management.html#dual-passwords

Additional Information:

You may wish to first assess the state of passwords on your MySQL Server.

Password expiration is based on the policy set by the `default_password_lifetime` or by explicit settings on a per account basis.

To assess when passwords were last changed on accounts and when, or if, they will expire run the following

```
select user, host, password_last_changed,
IF(IFNULL(password_lifetime, CAST(@@default_password_lifetime as
signed))<1,'NEVER',
concat(
cast(
IFNULL(password_lifetime, @@default_password_lifetime) as signed)
+ cast(datediff(password_last_changed, now()) as signed), " days")) as
days_till_expires
from mysql.user;
```

Applications will fail to authenticate when `days_till_expires` reaches `0`.

If `NEVER` returns for an account and the password has not be rotated in a long period of time it is recommended that action be taken to set a new password.

2.12 Lock Out Accounts if Not Currently in Use (Manual)

Profile Applicability:

- Level 2 - MySQL RDBMS

- Level 2 - MySQL RDBMS on Linux

Description:

If users with accounts will not be using their account for some time, to reduce the risk of attacks or inappropriate account usage or if suspicions exist that an account might be under attack, disabling the account will secure it and once it's ready to resume use it can easily be re-enabled.

Rationale:

Only have active accounts that will be used.

Audit:

Review the locked status of accounts:

```
select user, host, account_locked from mysql.user;
```

Accounts not in use and MySQL Reserved accounts should show as locked (`Y`).

Remediation:

To lock accounts - example:

```
ALTER USER 'jeffrey'@'localhost' ACCOUNT LOCK;
```

To unlock accounts - example

```
ALTER USER 'jeffrey'@'localhost' ACCOUNT UNLOCK;
```

Note: Works for `CREATE` as well. It is good practice to `LOCK` an account if created ahead of time.

Default Value:

Accounts are unlocked by default.

References:

1. https://dev.mysql.com/doc/refman/8.0/en/a...

Additional Information:

When a client attempts to connect to a locked account, the attempt fails.

```
Access denied for user 'user_name'@'host_name'.
Account is locked.
```

The server increments the `Locked_connects` status variable that indicates the number of attempts to connect to a locked account. To view the `Locked_conects` execute this query:

```
show global status like 'Locked_connects';
```

The error log will contain the message `ER_ACCOUNT_HAS_BEEN_LOCKED`.

CIS Controls:

Version 7

16.9 Disable Dormant Accounts
Automatically disable dormant accounts after a set period of inactivity.

2.13 Ensure AES Encryption Mode for AES_ENCRYPT/AES_DECRYPT is Configured Correctly (Automated)

Profile Applicability:

- Level 2 - MySQL RDBMS
- Level 2 - MySQL RDBMS on Linux

Description:

A block encryption mode with a Cypher Block Chaining (CBC) mode value and key length of 256 is recommended when using the `AES_ENCRYPT()` and `AES_DECRYPT()` functions for encryption.

Rationale:

The default for backward compatibility on upgraded MySQL databases is `aes-128-ecb`. Using 128-bit keys does not provide sufficient security. Regardless of whether breaking the lowest level is beyond existing technology, larger key sizes are needed to better protect data and satisfy regulations.

Impact:

Configuring a key length of 256 may impact backwards compatibility.

Audit:

Run the following statement:

```
select @@block_encryption_mode;
```

A value other than `aes-256-*` is a fail.
Where `*` is one of the following - `ECB`, `CBC`, `CFB1`, `CFB8`, `CFB128`, `OFB`

Remediation:

Add the following lines to the MySQL server's `/etc/my.cnf`:

For example, if Block Encryption Mode for aes-256 CBC

```
block_encryption_mode=aes-256-cbc
```

Or, run the following command:

```
set persist block_encryption_mode='aes-256-cbc';
```

Restart the server for this change to take effect.

Default Value:

```
aes-128-ecb
```

References:

1. https://dev.mysql.com/doc/mysql-secure-deployment-guide/8.0/en/secure-deployment-block-encryption-mode.html

CIS Controls:

Version 7

18.5 Use Only Standardized and Extensively Reviewed Encryption Algorithms
Use only standardized and extensively reviewed encryption algorithms.

2.14 Ensure Socket Peer-Credential Authentication is Used Appropriately (Manual)

Profile Applicability:

- Level 2 - MySQL RDBMS on Linux

Description:

The server-side `auth_socket` authentication plugin, authenticates clients that connect to the MySQL server from the local host through the Unix socket file. Users authenticated by the `auth_socket` need not specify a password when connecting to the server. However, users authenticated by the `auth_socket` plugin are restricted from connecting remotely; they can only connect from the local host through the Unix socket file. This method is only suitable in situations where the server administrator OS account access is restricted.

Rationale:

This method may be desirable in specific cases, including:

- The Linux system where MySQL is running is dedicated to the MySQL server and only the MySQL DBA and OS Admin have access.
- When control over user authentication is centralized in the operating system.
- It is desirable that audit trails in the database and operating system can use the same user names.
- For certain other narrow installation use cases `auth_socket` may be desirable.
- Only local connections for a user.

Impact:

Things to consider when using the operating system to authenticate users:

- The user must have an operating system account on the computer which must be accessed.
- If a user has logged in using this method and steps away from the terminal, another user could easily log in because this user does not need any passwords or credentials. This could pose a serious security problem.
- When an operating system is used to authenticate database users, managing distributed database environments and database links requires special care. Special care must also be taken not to leave such a terminal unlocked and unattended. Hence, we recommend that you carefully evaluate your requirements before opting for `auth_socket`.
- This will not work where distributed connections are required.

Audit:

To assess this recommendation run the following:

```
SELECT PLUGIN_NAME, PLUGIN_STATUS
       FROM INFORMATION_SCHEMA.PLUGINS
       WHERE PLUGIN_NAME LIKE 'auth%';
```

To determine users who can use auth socket:

```
select user, host, plugin from mysql.user where plugin = 'auth_socket';
```

If this is enabled and the organization does not allow use of this feature, this is a fail.

If `host` is not the localhost or an unauthorized user is listed, this is a fail.

Remediation:

Add these options under the `[mysqld]` option group in the MySQL `/etc/my.cnf`:

```
plugin-load-add=auth_socket.so
auth_socket=FORCE_PLUS_PERMANENT
```

For example:
For an OS user which can login to MySQL using `auth_socket`:

```
CREATE USER '<user>'@'localhost' IDENTIFIED WITH auth_socket;
```

The user can then login using

```
mysql -u <user>
```

References:

1. https://dev.mysql.com/doc/mysql-secure-deployment-guide/8.0/en/secure-deployment-configure-authentication.html

2.15 Ensure MySQL is Bound to an IP Address (Automated)

Profile Applicability:

- Level 2 - MySQL RDBMS
- Level 2 - MySQL RDBMS on Linux

Description:

By default, the MySQL server accepts TCP/IP connections from MySQL user accounts on all server host IPv6 and IPv4 interfaces. You can make this configuration more restrictive by setting the `bind_address` configuration option to a specific IPv4 or IPv6 address so that the server only accepts TCP/IP connections on that address.

Rationale:

Limiting the IP address provides additional controls and restrictions on how client applications can connect to MySQL. If not configured to a specific IP all IPs for this server can be used to connect to MySQL.

Audit:

Run the following statement:

```
SELECT VARIABLE_NAME, VARIABLE_VALUE
FROM performance_schema.global_variables
WHERE VARIABLE_NAME = 'bind_address';
```

No rows returned implies a fail.

Remediation:

For example, to have the MySQL server only accept connections on a specific IPv4 address, add an entry similar to this under the `[mysqld]` option group in the MySQL `/etc/my.cnf`:

```
bind_address=192.0.2.24
```

In this case, clients can connect to the server using `--host=192.0.2.24`. Connections on other server host addresses are not permitted.

Default Value:

Not set.

References:

1. https://dev.mysql.com/doc/mysql-secure-deployment-guide/8.0/en/secure-deployment-secure-connections.html

2.16 Limit Accepted Transport Layer Security (TLS) Versions (Automated)

Profile Applicability:

- Level 2 - MySQL RDBMS
- Level 2 - MySQL RDBMS on Linux

Description:

MySQL supports multiple protocols of TLS. The higher the version the stronger the security and/or better the performance.

Rationale:

Requiring clients attempting to connect to MySQL to use higher versions of TLS to better protect data in transit.

Impact:

Connections attempting to use an unsupported version of TLS or Cipher will fail.

Audit:

To list the versions of TLS the server accepts, run the following statement:

```
select @@tls_version;
```

If the list includes `TLSv1` and/or `TLSv1.1`, this is a fail.

To view current connections and the version of SSL in use run:

```
select * from performance_schema.status_by_thread where VARIABLE_NAME like
'ssl_version';
```

If the list includes, `TLSv1` and/or `TLSv1.1`, this is a fail.

MySQL negotiates to the highest version of TLS, if connections are using older TLS versions, those clients will need to be upgraded to newer MySQL Connectors or community drivers that support newer versions of TLS.

Remediation:

Set the version(s) of TLS you wish to accept in `mysql.conf` specify TLS and Ciphers.
For example to only accept TLS 1.3 set `tls_version` in `my.conf`:

```
tls_version=TLSv1.3
```

If TLS 1.3 is not supported on the Operating System then set to TLS 1.2:

```
tls_version=TLSv1.2
```

Note: with this setting, only clients that support the specified TLS version(s) are able to establish an encrypted connection to the server.

Default Value:

All TLS and cipher versions are enabled by default.

References:

1. https://dev.mysql.com/doc/mysql-secure-deployment-guide/8.0/en/secure-deployment-secure-connections.html
2. https://dev.mysql.com/doc/refman/8.0/en/encrypted-connection-protocols-ciphers.html#encrypted-connection-protocol-configuration

CIS Controls:

Version 7

14.4 Encrypt All Sensitive Information in Transit
Encrypt all sensitive information in transit.

18.5 Use Only Standardized and Extensively Reviewed Encryption Algorithms
Use only standardized and extensively reviewed encryption algorithms.

2.17 Require Client-Side Certificates (X.509) (Automated)

Profile Applicability:

- Level 2 - MySQL RDBMS
- Level 2 - MySQL RDBMS on Linux

Description:

Client-side certificates may be used as proof of identity as well as to encrypt data in transit.

Rationale:

Requiring client-side certificates provides additional validation of a user's identity, thus increasing the level of security, while also providing strong encryption.

Audit:

Run the following statement

```
select user, host, ssl_type from mysql.user;
```

If `ssl_type` returns `X509` or `SPECIFIED`, client-side certificate details must be provided to connect.

Remediation:

Create or Alter users using the `REQUIRE X509`.
For example:

```
CREATE USER 'newuser2'@'%' IDENTIFIED BY <password> require x509;
```

For accounts created with a `REQUIRE X509` clause, clients must specify at least `--ssl-cert` and `--ssl-key`. In addition, `--ssl-ca` (or `--ssl-capath`) is recommended so that the public certificate provided by the server can be verified.

For example:

```
mysql --ssl-ca=ca.pem \
      --ssl-cert=client-cert.pem \
      --ssl-key=client-key.pem
```

References:

1. https://dev.mysql.com/doc/refman/8.0/en/using-encrypted-connections.html

CIS Controls:

Version 7

14.4 Encrypt All Sensitive Information in Transit
Encrypt all sensitive information in transit.

2.18 Ensure Only Approved Ciphers are Used (Automated)

Profile Applicability:

- Level 2 - MySQL RDBMS on Linux
- Level 2 - MySQL RDBMS

Description:

MySQL supports multiple encryption ciphers. Ciphers can vary in strength, speed and overhead.

Rationale:

Requiring clients attempting to connect to MySQL to use strong ciphers protects data in transit.

Impact:

Connections attempting to use an unsupported cipher will fail.

Audit:

Run the following statement:

```
SELECT VARIABLE_NAME, VARIABLE_VALUE
FROM performance_schema.global_variables
WHERE VARIABLE_NAME IN ('ssl_cipher', 'tls_ciphersuites');
```

If `ssl_cipher` is not set to `ECDHE-ECDSA-AES128-GCM-SHA256` or `tls_ciphersuites` is not set to `TLS_AES_256_GCM_SHA384`, this is a fail.

Remediation:

Set `ssl_ciphers` and `tls_ciphersuites` in the `mysql.conf` to an approved cipher suite:

```
tls_ciphersuites='TLS_AES_256_GCM_SHA384'
ssl_ciphers='ECDHE-ECDSA-AES128-GCM-SHA256'
```

OR
Execute the following commands:

```
set persist ssl_cipher='ECDHE-ECDSA-AES128-GCM-SHA256';
set persist tls_ciphersuites='TLS_AES_256_GCM_SHA384';
```

Default Value:

```
ECDHE-ECDSA-AES128-GCM-SHA256
ECDHE-ECDSA-AES256-GCM-SHA384
ECDHE-RSA-AES128-GCM-SHA256
ECDHE-RSA-AES256-GCM-SHA384
ECDHE-ECDSA-AES128-SHA256
ECDHE-RSA-AES128-SHA256
ECDHE-ECDSA-AES256-SHA384
ECDHE-RSA-AES256-SHA384
DHE-RSA-AES128-GCM-SHA256
DHE-DSS-AES128-GCM-SHA256
DHE-RSA-AES128-SHA256
DHE-DSS-AES128-SHA256
DHE-DSS-AES256-GCM-SHA384
DHE-RSA-AES256-SHA256
DHE-DSS-AES256-SHA256
ECDHE-RSA-AES128-SHA
ECDHE-ECDSA-AES128-SHA
ECDHE-RSA-AES256-SHA
ECDHE-ECDSA-AES256-SHA
DHE-DSS-AES128-SHA
DHE-RSA-AES128-SHA
TLS_DHE_DSS_WITH_AES_256_CBC_SHA
DHE-RSA-AES256-SHA
AES128-GCM-SHA256
DH-DSS-AES128-GCM-SHA256
ECDH-ECDSA-AES128-GCM-SHA256
AES256-GCM-SHA384
DH-DSS-AES256-GCM-SHA384
ECDH-ECDSA-AES256-GCM-SHA384
AES128-SHA256
DH-DSS-AES128-SHA256
ECDH-ECDSA-AES128-SHA256
AES256-SHA256
DH-DSS-AES256-SHA256
ECDH-ECDSA-AES256-SHA384
AES128-SHA
DH-DSS-AES128-SHA
ECDH-ECDSA-AES128-SHA
AES256-SHA
DH-DSS-AES256-SHA
ECDH-ECDSA-AES256-SHA
DHE-RSA-AES256-GCM-SHA384
DH-RSA-AES128-GCM-SHA256
ECDH-RSA-AES128-GCM-SHA256
DH-RSA-AES256-GCM-SHA384
ECDH-RSA-AES256-GCM-SHA384
DH-RSA-AES128-SHA256
ECDH-RSA-AES128-SHA256
DH-RSA-AES256-SHA256
ECDH-RSA-AES256-SHA384
ECDHE-RSA-AES128-SHA
ECDHE-ECDSA-AES128-SHA
ECDHE-RSA-AES256-SHA
ECDHE-ECDSA-AES256-SHA
```

```
DHE-DSS-AES128-SHA
DHE-RSA-AES128-SHA
TLS_DHE_DSS_WITH_AES_256_CBC_SHA
DHE-RSA-AES256-SHA
AES128-SHA
DH-DSS-AES128-SHA
ECDH-ECDSA-AES128-SHA
AES256-SHA
DH-DSS-AES256-SHA
ECDH-ECDSA-AES256-SHA
DH-RSA-AES128-SHA
ECDH-RSA-AES128-SHA
DH-RSA-AES256-SHA
ECDH-RSA-AES256-SHA
DES-CBC3-SHA
```

References:

1. https://dev.mysql.com/doc/refman/8.0/en/encrypted-connection-protocols-ciphers.html#encrypted-connection-cipher-configuration

CIS Controls:

Version 7

18.5 Use Only Standardized and Extensively Reviewed Encryption Algorithms
Use only standardized and extensively reviewed encryption algorithms.

2.19 Implement Connection Delays to Limit Failed Login Attempts (Automated)

Profile Applicability:

- Level 1 - MySQL RDBMS
- Level 1 - MySQL RDBMS on Linux

Description:

MySQL Server can enable administrators to introduce an increasing delay in server response to clients after a certain number of consecutive failed connection attempts.

Rationale:

Delaying connection attempts provides a deterrent that slows down brute force attacks that attempt to access MySQL user accounts.

Audit:

Determine if the plugins for delaying connections are installed.

```
SELECT PLUGIN_NAME, PLUGIN_STATUS
       FROM INFORMATION_SCHEMA.PLUGINS
       WHERE PLUGIN_NAME LIKE 'connection%';
```

Two rows should be returned showing `ACTIVE` status.

```
CONNECTION_CONTROL                      | ACTIVE
CONNECTION_CONTROL_FAILED_LOGIN_ATTEMPTS | ACTIVE
```

If both plugins are not active, this is a fail.

Next assess the setting for the connection controls by running

```
SELECT VARIABLE_NAME, VARIABLE_VALUE
       FROM performance_schema.global_variables WHERE VARIABLE_NAME LIKE
'connection_control%';
```

Time doubling throttling (in minutes) between each retry (`0`, `1`, `2`, `4`, `8`, etc.) with a permanent account lockout (IT reset required) after 12 retries.

If `connection_control_failed_connections_threshold` is less than `5` (attempts), this is a fail.

If `connection_control_min_connection_delay` is less than `60000` (ms - 1 minute), this is a fail.

Max delay `connection_control_max_connection_delay` is `0` or less than `1920000` (ms, 32 minutes) a, this is a fail.

Finally, assess the failed login attempts.

```
select host, user, JSON_EXTRACT(user_attributes,
'$.Password_locking.failed_login_attempts') as failed_login_attempts from
mysql.user;
```

If failed login attempts is less than `12` this is a fail.

Remediation:

Add the following lines to `my.cnf`:

```
[mysqld]
plugin-load-add=connection_control.so
connection-control=FORCE_PLUS_PERMANENT
connection-control-failed-login-attempts=FORCE_PLUS_PERMANENT
connection_control_failed_connections_threshold=5
connection_control_min_connection_delay=60000
connection_control_max_connection_delay=1920000
```

Delays are in milliseconds for server response to failed connection attempt.

- `60000` (ms - 1 minute)
- `1920000` (ms, 32 minutes)

For each user set

```
ALTER USER <user> FAILED_LOGIN_ATTEMPTS 12;
```

References:

1. https://dev.mysql.com/doc/refman/8.0/en/connection-control.html

CIS Controls:

Version 7

16 Account Monitoring and Control
Account Monitoring and Control

3 File Permissions

File Permissions are critical for keeping the data and configuration of the MySQL server secure.

3.1 Ensure 'datadir' Has Appropriate Permissions (Automated)

Profile Applicability:

- Level 1 - MySQL RDBMS on Linux

Description:

The data directory is the location of the MySQL databases.

Rationale:

Limiting the accessibility of these objects will protect the confidentiality, integrity, and availability of the MySQL database. If someone other than the MySQL user is allowed to read files from the data directory, it may be possible to read data from the `mysql.user` table which contains passwords. Additionally, the ability to create files can lead to denial of service, or might otherwise allow someone to gain access to specific data by manually creating a file with a view definition.

Audit:

Perform the following steps to assess this recommendation:

- Execute the following SQL statement to determine the `Value` of `datadir`

  ```
  show variables where variable_name = 'datadir';
  ```

  Or

  ```
  SELECT VARIABLE_NAME, VARIABLE_VALUE
  FROM performance_schema.global_variables
  WHERE VARIABLE_NAME LIKE 'datadir';
  ```

- Execute the following command at a terminal prompt

  ```
  sudo ls -ld <datadir> | grep "drwxr-x---.*mysql.*mysql"
  ```

  Lack of output implies a fail.

Remediation:

Execute the following commands at a terminal prompt:

```
chmod 750 <datadir>
chown mysql:mysql <datadir>
```

References:

1. https://dev.mysql.com/doc/mysql-secure-deployment-guide/8.0/en/secure-deployment-permissions.html

CIS Controls:

Version 7

14.6 Protect Information through Access Control Lists
Protect all information stored on systems with file system, network share, claims, application, or database specific access control lists. These controls will enforce the principle that only authorized individuals should have access to the information based on their need to access the information as a part of their responsibilities.

3.2 Ensure 'log_bin_basename' Files Have Appropriate Permissions (Automated)

Profile Applicability:

- Level 1 - MySQL RDBMS on Linux

Description:

MySQL can operate using a variety of log files, each used for different purposes. These are the binary log (which can be encrypted), error log, slow query log, relay log, general log, and in the enterprise edition, the audit log (which can be encrypted). Because these are files on the host operating system, they are subject to the permissions and ownership structure provided by the host and may be accessible by users other than the MySQL user. Additionally, using secure key management and at rest MySQL encryption can further protect data from OS users.

Rationale:

Limiting the accessibility of these objects will protect the confidentiality, integrity, and availability of the MySQL logs.

Impact:

Changing the permissions and ownership of the relay logs and binary log files might have impact on external tools.

If the permissions on the relay logs and binary log files are accidentally changed to exclude the user account which is used to run the MySQL service, then this might break replication.

The binary log file can be used for point-in-time recovery so this can also affect backup, restore, and disaster recovery procedures.

Audit:

Perform the following steps to assess this recommendation:

1. Execute the following SQL statement to determine the Value of `log_bin_basename`:

```
show variables like 'log_bin_basename';
```

2. Execute the following command at a terminal prompt to list all non-compliant `log_bin_basename.*` file permissions:

```
ls -l | egrep '^-(?![r|w]{2}-[r|w]{2}----
.*mysql\s*mysql).*<log_bin_basename>.*$'
```

Lack of output implies compliance.

Remediation:

Execute the following command for each log file location requiring corrected permissions and ownership:

```
chmod 660 <log file>
chown mysql:mysql <log file>
```

References:

1. https://dev.mysql.com/doc/refman/8.0/en/password-logging.html
2. https://dev.mysql.com/doc/mysql-secure-deployment-guide/8.0/en/secure-deployment-permissions.html

CIS Controls:

Version 7

14.6 Protect Information through Access Control Lists
Protect all information stored on systems with file system, network share, claims, application, or database specific access control lists. These controls will enforce the principle that only authorized individuals should have access to the information based on their need to access the information as a part of their responsibilities.

3.3 Ensure 'log_error' Has Appropriate Permissions (Automated)

Profile Applicability:

- Level 1 - MySQL RDBMS on Linux

Description:

MySQL can operate using a variety of log files, each used for different purposes. These are the binary log (which can be encrypted), error log, slow query log, relay log, general log, and in the enterprise edition, the audit log (which can be encrypted). Because these are files on the host operating system, they are subject to the permissions and ownership structure provided by the host and may be accessible by users other than the MySQL user. Additionally, using secure key management and at rest MySQL encryption can further protect data from OS users.

Much of the information about the state of MySQL exists in MySQL, the MySQL `performance_schema` or `informations_schema`. In cases where the information you need is within a running MySQL, use these methods as they are more secure as do not require OS login and access.

Rationale:

Limiting the accessibility of these objects will protect the confidentiality, integrity, and availability of the MySQL logs.

Impact:

Changing the permissions of the error log files might have impact on monitoring tools which use an error log file adapter.

Audit:

Perform the following steps to assess this recommendation:

1. Execute the following SQL statement to determine the Value of `log_error`:

```
show variables like 'log_error';
```

2. Execute the following command at a terminal prompt to list all non-compliant `<log_error>.*` file permissions:

   ```
   ls -l /usr/local/mysql/data/mysqld.local.err | grep '^-rw-------
   .*mysql.*mysql.*$'
   ```

   Lack of output implies a fail.

Remediation:

Execute the following command for each log file location requiring corrected permissions and ownership:

```
chmod 600 <log file>
chown mysql:mysql <log file>
```

References:

1. https://dev.mysql.com/doc/refman/8.0/en/error-log.html
2. https://dev.mysql.com/doc/mysql-secure-deployment-guide/5.7/en/secure-deployment-permissions.html

CIS Controls:

Version 7

14.6 Protect Information through Access Control Lists
Protect all information stored on systems with file system, network share, claims, application, or database specific access control lists. These controls will enforce the principle that only authorized individuals should have access to the information based on their need to access the information as a part of their responsibilities.

3.4 Ensure 'slow_query_log' Has Appropriate Permissions (Automated)

Profile Applicability:

- Level 1 - MySQL RDBMS on Linux

Description:

MySQL can operate using a variety of log files, each used for different purposes. These are the binary log (which can be encrypted), error log, slow query log, relay log, general log, and in the enterprise edition, the audit log (which can be encrypted). Because these are files on the host operating system, they are subject to the permissions and ownership structure provided by the host and may be accessible by users other than the MySQL user. Additionally, using secure key management and at rest MySQL encryption can further protect data from OS users.

Much of the information about the state of MySQL exists in MySQL, the MySQL `performance_schema` or `informations_schema`. If you can get the information you need from within MySQL that is more secure as it does not require OS access. If you are not going to use log files it is best to first disable (don’t enable) and remove any prior logs.

Rationale:

Limiting the accessibility of these objects will protect the confidentiality, integrity, and availability of the MySQL logs.

Impact:

Changing the permissions of the log files may impact monitoring tools which use a log file adapter. Also, the slow query log can be used for performance analysis by application developers.

The information about the performance exists in MySQL `performance_schema` or `sys` schema views. In cases where the information you need is within a running MySQL, disable the slow query log and instead use these methods as they are more secure and do not require OS login and access.

Audit:

Perform the following steps to assess this recommendation:

1. Execute the following SQL statement to determine the Value of `slow_query_log`:

```
show variables like 'slow_query_log';
```

Best for the slow query log to be disabled indicated by OFF.

2. Execute the following SQL statement to determine the location of `slow_query_log_file`:

```
show variables like 'slow_query_log_file';
```

3. Execute the following command at a terminal prompt to list non-compliant *`<slow_query_log_file>.*`* file permissions:

```
ls -l | egrep "^-(?![r|w]{2}-[r|w]{2}----
.*mysql\s*mysql).*<slow_query_log_file>.*$
```

If the slow query log is enabled, lack of output implies compliance.
If the slow query log is disabled, remove any old slow query log files.

Remediation:

Set slow query log to OFF (instead use `SYS` schema views or query `Performance_Schema`)

```
SET PERSIST slow_query_log = OFF;
```

If slow query is enabled, execute the following command to correct permissions and ownership:

```
chmod 660 <log file>
chown mysql:mysql <log file>
```

Default Value:

Slow query log is off by default.

References:

1. https://dev.mysql.com/doc/refman/8.0/en/slow-query-log.html

CIS Controls:

Version 7

14.6 Protect Information through Access Control Lists
Protect all information stored on systems with file system, network share, claims, application, or database specific access control lists. These controls will enforce the principle that only authorized individuals should have access to the information based on their need to access the information as a part of their responsibilities.

3.5 Ensure 'relay_log_basename' Files Have Appropriate Permissions (Automated)

Profile Applicability:

- Level 1 - MySQL RDBMS on Linux

Description:

MySQL can operate using a variety of log files, each used for different purposes. These are the binary log (which can be encrypted), error log, slow query log, relay log (which can be encrypted), general log, and in the enterprise edition, the audit log (which can be encrypted). Because these are files on the host operating system, they are subject to the permissions and ownership structure provided by the host and may be accessible by users other than the MySQL user. Additionally, using secure key management and at rest MySQL encryption can further protect data from OS users.

Rationale:

Limiting the accessibility of these objects will protect the confidentiality, integrity, and availability of the MySQL logs.

Impact:

If the permissions on the relay logs and binary log files are accidentally changed to exclude the user account which is used to run the MySQL service, then this might break replication.

The binary log file can be used for point in time recovery so this can also affect backup, restore and disaster recovery procedures.

Audit:

Perform the following steps to assess this recommendation:

1. Execute the following SQL statement to determine the Value of `relay_log_basename`:

```
show variables like 'relay_log_basename';
```

2. Execute the following command at a terminal prompt to list non-compliant <relay_log_basename>.* file permissions:

   ```
   ls -l | egrep "^-(?![r|w]{2}-[r|w]{2}----
   .*mysql\s*mysql).*<relay_log_basename>.*$
   ```

   Lack of output implies compliance.

Remediation:

Execute the following command for each log file location requiring corrected permissions and ownership:

```
chmod 660 <log file>
chown mysql:mysql <log file>
```

Default Value:

<datadir> + '/' + *<hostname>* + '-relay-bin'

CIS Controls:

Version 7

14.6 Protect Information through Access Control Lists
Protect all information stored on systems with file system, network share, claims, application, or database specific access control lists. These controls will enforce the principle that only authorized individuals should have access to the information based on their need to access the information as a part of their responsibilities.

3.6 Ensure 'general_log_file' Has Appropriate Permissions (Automated)

Profile Applicability:

- Level 1 - MySQL RDBMS on Linux

Description:

MySQL can operate using a variety of log files, each used for different purposes. These are the binary log (which can be encrypted), error log, slow query log, relay log (which can be encrypted), general log, and in the enterprise edition, the audit log (which can be encrypted). Because these are files on the host operating system, they are subject to the permissions and ownership structure provided by the host and may be accessible by users other than the MySQL user. Additionally, using secure key management and at rest MySQL encryption can further protect data from OS users.

Much of the information about the state of MySQL exists in MySQL, the MySQL `performance_schema` or `informations_schema`. If you can get the information you need from within MySQL that is more secure as it does not require OS access. If you are not going to use log files it is best to first disable (don't enable) and remove any prior logs.

Rationale:

Limiting the accessibility, or existence, of these log files will protect the confidentiality, integrity, and availability of the MySQL logs.

Impact:

Changing the permissions of the general log files may impact monitoring tools which use a log file adapter.

Audit:

Perform the following steps to assess this recommendation:

1. Execute the following SQL statement to determine the Values of `general_log` and `general_log_file`:

   ```
   select @@general_log, @@general_log_file;
   ```

   With a `general_log` value of `0`, indicates the log is disabled. If `1` it is enabled.

2. Whether the value is `0` or `1` execute the following command at a terminal prompt to list non-compliant *`<general_log_file>.*`* file permissions:

```
ls -l <general_log_file>
```

If general_log is 0 (disabled) and the log file exists, remove the old general log file.

If general_log is 1 (enabled) review the permissions

```
ls -l <general_log_file> grep '^-rw-------.*mysql.*mysql'
```

Lack of output implies compliance.

Remediation:

If you can, use MySQL SYS, PERFORMANCE_SCHEMA, or MySQL Auditing as these are more secure options.

By default the general_log is disabled (0). Its most secure to disable the general_log.

To disable the general_log_file:

```
SET PERSIST @@GENERAL_LOG=0;
```

If you must use general_log then assure the permissions are correct. Execute the following command for each log file location requiring corrected permissions and ownership:

```
chmod 600 <general_log_file>
chown mysql:mysql <general_log_file>
```

Default Value:

The general log file is off by default.

References:

1. https://dev.mysql.com/doc/refman/8.0/en/query-log.html

CIS Controls:

Version 7

14.6 Protect Information through Access Control Lists
Protect all information stored on systems with file system, network share, claims, application, or database specific access control lists. These controls will enforce the principle that only authorized individuals should have access to the information based on their need to access the information as a part of their responsibilities.

3.7 Ensure SSL Key Files Have Appropriate Permissions (Automated)

Profile Applicability:

- Level 1 - MySQL RDBMS on Linux

Description:

When configured to use SSL/TLS, MySQL relies on Secure Sockets Layer (SSL) key files, which are stored on the host's filesystem. These SSL key files are subject to the host's permissions and ownership structure.

MySQL 8.0 provides ways to create the SSL certificate, SSL key files and RSA key-pair files required to support encrypted connections using SSL and secure password exchange using RSA over unencrypted connections, if those files are missing the server will attempt to autogenerate these files at startup if compiled with OpenSSL.

Rationale:

Limiting the accessibility of these objects will protect the confidentiality, integrity, and availability of the MySQL database and the communication with the client.

If the contents of the SSL key file are known to an attacker, he or she might impersonate the server. This can be used for a man-in-the-middle attack.

Depending on the SSL cipher suite, the key might also be used to decipher previously captured network traffic.

Impact:

If the permissions or ownership for the SSL key file are configured incorrectly, this can cause SSL to be disabled when MySQL is restarted or can cause MySQL not to start at all.

If other applications are using the same key pair, then changing the permissions or ownership of the SSL key file will affect this application. If this were to occur a new key pair must be generated for MySQL.

Audit:

Perform the following steps to assess this recommendation:

1. Locate the SSL keys and certs in use by executing the following SQL statement. To show all ssl variables:

```
SELECT * FROM performance_schema.global_variables
WHERE
REGEXP_LIKE(VARIABLE_NAME,'^.*ssl_(ca|capath|cert|cipher|crl|crlpath|ke
y)$') AND VARIABLE_VALUE <> '';
```

Note: Any `mysqlx_%` values that are null default to the classic protocols equivalent value.

2. Execute the following commands at a terminal prompt to list non-compliant *<ssl_file>* file permissions:

```
ls -l | egrep "^-(?!r-{8}.*mysql\s*mysql).*<ssl_file>.*$
```

Lack of output implies compliance

Remediation:

Execute the following commands at a terminal prompt to remediate these settings using the Value from the audit procedure:

```
chown mysql:mysql <ssl_file>
chmod 400 <ssl_file>
```

References:

1. https://dev.mysql.com/doc/refman/8.0/en/encrypted-connections.html
2. https://dev.mysql.com/doc/refman/8.0/en/creating-ssl-rsa-files-using-mysql.html

Additional Information:

If SSL is not configured this recommendation is not applicable. By default MySQL enables SSL. Using SSL is highly recommended.

CIS Controls:

Version 7

14.6 Protect Information through Access Control Lists
Protect all information stored on systems with file system, network share, claims, application, or database specific access control lists. These controls will enforce the principle that only authorized individuals should have access to the information based on their need to access the information as a part of their responsibilities.

3.8 Ensure Plugin Directory Has Appropriate Permissions (Automated)

Profile Applicability:

- Level 1 - MySQL RDBMS on Linux

Description:

The plugin directory is the location of the MySQL plugins. Plugins are storage engines or user defined functions (UDFs).

Rationale:

Limiting the accessibility of these objects will protect the confidentiality, integrity, and availability of the MySQL database. If someone can modify plugins then these plugins might be loaded when the server starts and the code will get executed.

Impact:

Users other than the MySQL user will no longer be able to update and add/remove plugins unless they're able to switch to the MySQL user.

Audit:

To assess this recommendation, execute the following SQL statement to discover the Value of `plugin_dir`:

```
show variables where variable_name = 'plugin_dir';
```

Then, execute the following command at a terminal prompt (using the discovered `plugin_dir Value`) to determine the permissions and ownership.

```
ls -ld <plugin_dir Value> | grep "dr-xr-x---\|dr-xr-xr--" | grep "plugin"
```

Lack of output implies a fail.
Note: Permissions are intended to be either `550` or `554`.

Remediation:

To remediate these settings, execute the following commands at a terminal prompt using the `plugin_dir Value` from the audit procedure. MySQL server must not be allowed to write to this location.

```
chmod 550 <plugin_dir Value> (or use 554)
chown mysql:mysql <plugin_dir Value>
```

References:

1. https://dev.mysql.com/doc/refman/8.0/en/install-plugin.html

CIS Controls:

Version 7

14.6 Protect Information through Access Control Lists
Protect all information stored on systems with file system, network share, claims, application, or database specific access control lists. These controls will enforce the principle that only authorized individuals should have access to the information based on their need to access the information as a part of their responsibilities.

3.9 Ensure 'audit_log_file' Has Appropriate Permissions (Automated)

Profile Applicability:

- Level 1 - MySQL RDBMS on Linux

Description:

MySQL can operate using a variety of log files, each used for different purposes. These are the binary log, error log, slow query log, relay log, audit log and general log. Because these are files on the host operating system, they are subject to the permissions and ownership structure provided by the host and may be accessible by users other than the MySQL user.

Rationale:

Limiting the accessibility of these objects will protect the confidentiality, integrity, and availability of the MySQL logs.

Impact:

Changing the permissions and ownership of the audit log file may have an impact on who can access and edit the audit log. Such changes can affect monitoring tools which maybe using a log file adapter or scripted alternatives. Also, the audit log may be used for alerting by infrastructure teams which can affect real-time audit capability.

Audit:

To assess this recommendation, execute the following SQL statement to discover the `audit_log_file` value:

```
show global variables where variable_name='audit_log_file';
```

If no value is returned, auditing is not installed, and this is a fail.
Note: If you see the audit file name but no path, the default path will be the path assigned to the `datadir` variable.
Then, execute the following command at a terminal prompt (using the discovered `audit_log_file` value):

```
ls -l <audit_log_file> | egrep "^-([rw-]{2}-){2}---[ \t]*[0-9][ \t]*mysql[
\t]*mysql.*$"
```

No results implies a fail.

Remediation:

Execute the following commands for the `audit_log_file` discovered in the audit procedure:

```
chmod 660 <audit_log_file>
chown mysql:mysql <audit_log_file>
```

CIS Controls:

Version 7

14.6 Protect Information through Access Control Lists
Protect all information stored on systems with file system, network share, claims, application, or database specific access control lists. These controls will enforce the principle that only authorized individuals should have access to the information based on their need to access the information as a part of their responsibilities.

3.10 Secure MySQL Keyring (Automated)

Profile Applicability:

- Level 1 - MySQL RDBMS on Linux

Description:

When configured to use a Keyring plugin, internal MySQL components and plugins may securely store sensitive information for later retrieval. Associated files for the selected keyring type should have proper permissions.

Rationale:

Limiting the accessibility of these objects will protect the confidentiality, integrity, and availability of internal MySQL component and plugin information.

Audit:

Perform the following steps applicable to the plugin in use to assess this recommendation:

Keyring File Plugin (Least Secure - for pre-production testing)

1. Find the `keyring_file_data` value (*`<keyring_file_data_path>`*) by executing the following statement:

```
grep -Po '(?<=^keyring_file_data=).+$' /etc/mysql/my.cnf
```

2. Verify permissions are `750` for `mysql:mysql` for *`<keyring_file_data_path>`*

Keyring Encrypted File Plugin

1. Find the `keyring_encrypted_file_data` value (*`<keyring_encrypted_file_data_path>`*) by executing the following statement:

```
grep -Po '(?<=^keyring_encrypted_file_data=).+$' /etc/mysql/my.cnf
```

2. Verify permissions are `750` for `mysql:mysql` for *`<keyring_encrypted_file_data_path>`*
3. Verify a secure method for provisioning the passphrase for the keyring_encrypted_file_data is in place.

Keyring OKV / KMIP compatible Plugin

1. Find the `keyring_okv` value (*<keyring_okv_path>*) by executing the following statement:

```
grep -Po '(?<=^keyring_okv=).+$' /etc/mysql/my.cnf
```

2. Verify permissions are `750` for `mysql:mysql` for *<keyring_okv_path>*

Keyring Oracle Cloud Infrastructure (OCI) Vault

1. Find the `keyring_oci_key_file` value (*<keyring_oci_key_file>*) by executing the following statement:

```
grep -Po '(?<=^keyring_oci_key_file=).+$' /etc/mysql/my.cnf
```

2. Verify permissions are `750` for `mysql:mysql` for *<keyring_oci_key_file>*

Keyring Hashicorp Vault Plugin

1. Find the `keyring_hashicorp_store_path` value (*<keyring_hashicorp_store_path>*) by executing the following statement:

```
grep -Po '(?<=^keyring_hashicorp_store_path=).+$' /etc/mysql/my.cnf
```

2. Verify permissions are `750` for `mysql:mysql` for *<keyring_hashicorp_store_path>*

AWS Key Management Service

1. Find the `keyring_aws_conf_file` and `keyring_aws_data_file` values by executing the following statement:

```
grep -Po '(?<=^keyring_aws.*=).+$' /etc/mysql/my.cnf
```

2. Verify permissions are `750` for `mysql:mysql` for `keyring_aws_conf_file` and `keyring_aws_data_file`

Additionally, if no keyring plugin or keyring file plugin is configured, this is a fail.

Remediation:

If no keyring plugin or keyring file plugin is configured, instructions for configuring a keyring plugin or keyring file plugin may found at:

- KMIP - https://dev.mysql.com/doc/refman/8.0/en/keyring-okv-plugin.html#keyring-okv-configuration
- OCI Vault - https://dev.mysql.com/doc/refman/8.0/en/keyring-oci-plugin.html
- Hashicorp - https://dev.mysql.com/doc/refman/8.0/en/keyring-hashicorp-plugin.html#keyring-hashicorp-plugin-configuration
- AWS - https://dev.mysql.com/doc/refman/8.0/en/keyring-aws-plugin.html#keyring-aws-plugin-configuration

Execute the following command for each Keyring file location requiring corrected permissions:

```
chmod 750 <keyring file>
chown mysql:mysql <keyring file>
```

References:

1. https://dev.mysql.com/doc/refman/8.0/en/keyring-system-variables.html

Additional Information:

Use of `keyring_file` is intended for development and testing and will not pass most security regulatory requirements.

CIS Controls:

Version 7

14.6 Protect Information through Access Control Lists
Protect all information stored on systems with file system, network share, claims, application, or database specific access control lists. These controls will enforce the principle that only authorized individuals should have access to the information based on their need to access the information as a part of their responsibilities.

4 General

This section contains recommendations related to various parts of the database server.

4.1 Ensure the Latest Security Patches are Applied (Manual)

Profile Applicability:

- Level 1 - MySQL RDBMS on Linux

Description:

Periodically, updates to MySQL server are released to resolve bugs, mitigate vulnerabilities, and provide new features. It is recommended that MySQL installations are up to date with the latest security updates.

Rationale:

Maintaining currency with MySQL patches will help reduce risk associated with known vulnerabilities present in the MySQL server.

Without the latest security patches MySQL might have known vulnerabilities which could be used by an attacker to gain access.

Impact:

To update the MySQL server a restart is required.

Audit:

Execute the following SQL statement to identify the MySQL server version:

```
SHOW VARIABLES WHERE Variable_name LIKE "version";
```

Now compare the version with the security announcements from Oracle and/or the OS if the OS packages are used.

Remediation:

Install the latest patches for your version or upgrade to the latest version.

References:

1. http://www.oracle.com/technetwork/topics/security/alerts-086861.html
2. http://dev.mysql.com/doc/relnotes/mysql/8.0/en/

3. https://nvd.nist.gov/vuln/search/results?form_type=Advanced&results_type=overview&search_type=all&cpe_vendor=cpe%3A%2F%3Aoracle&cpe_product=cpe%3A%2F%3Aoracle%3Amysql&cpe_version=cpe%3A%2F%3Aoracle%3Amysql%3A8.0

CIS Controls:

Version 7

2.2 Ensure Software is Supported by Vendor
Ensure that only software applications or operating systems currently supported by the software's vendor are added to the organization's authorized software inventory. Unsupported software should be tagged as unsupported in the inventory system.

4.2 Ensure Example or Test Databases are Not Installed on Production Servers (Automated)

Profile Applicability:

- Level 1 - MySQL RDBMS

- Level 1 - MySQL RDBMS on Linux

Description:

The default MySQL installation does not contain any example or test databases. However, it is a good idea to review for common example databases and ensure they have been removed from production systems.

Rationale:

Dropping example databases will reduce the attack surface of the MySQL server.

Audit:

Execute the following SQL statement to determine if the test database is present:

```
SELECT * FROM information_schema.SCHEMATA where SCHEMA_NAME not in
('mysql','information_schema', 'sys', 'performance_schema');
```

If this is a production system, and a database name includes an example database this is a finding.
Common example database names are:

- sakila
- world
- world_x
- menagerie

Remediation:

Execute the following SQL statement to drop an example database:

```
DROP DATABASE <database name>;
```

Default Value:

By default, MySQL 8.0 does not contain any example or test databases.

References:

1. http://dev.mysql.com/doc/refman/8.0/en/mysql-secure-installation.html

4.3 Ensure 'allow-suspicious-udfs' is Set to 'OFF' (Automated)

Profile Applicability:

- Level 2 - MySQL RDBMS on Linux

Description:

This option prevents attaching arbitrary shared library functions as user-defined functions by checking for at least one corresponding method named `_init`, `_deinit`, `_reset`, `_clear`, or `_add`.

Rationale:

Preventing shared libraries that do not contain user-defined functions from loading will reduce the attack surface of the server.

Audit:

Perform the following to determine if the recommended state is in place:

- Ensure `--allow-suspicious-udfs` is not specified in the the `mysqld` start up command line.
- Ensure `allow-suspicious-udfs` is set to `FALSE` in the MySQL configuration:

  ```
  my_print_defaults mysqld | grep allow-suspicious-udfs
  ```

  No results returned is a pass.

Remediation:

Perform the following to establish the recommended state:

- Remove `--allow-suspicious-udfs` from the `mysqld` start up command line.
- Remove `allow-suspicious-udfs` from the MySQL option file.

Default Value:

`OFF`

References:

1. https://dev.mysql.com/doc/extending-mysql/8.0/en/adding-udf.html#udf-security
2. http://dev.mysql.com/doc/refman/8.0/en/server-options.html#option_mysqld_allow-suspicious-udfs

Additional Information:

This option has no corresponding state in `SHOW VARIABLES`.

4.4 Harden Usage for 'local_infile' on MySQL Clients (Automated)

Profile Applicability:

- Level 1 - MySQL RDBMS
- Level 1 - MySQL RDBMS on Linux

Description:

The `local_infile` parameter dictates whether files located on the MySQL client's computer can be loaded or selected via `LOAD DATA INFILE` or `SELECT local_file`.

Rationale:

For MySQL client programs and connectors prior to 8.0.21, disabling `local_infile` reduces an attacker's ability to read sensitive files off the affected server via an SQL injection vulnerability.

Impact:

Disabling `local_infile` will impact the functionality of solutions that rely on it.

Audit:

Check the version of MySQL clients and connectors.
For example:

```
$ mysqlsh --version
$ mysql --version
```

The version should be 8.0.21 or higher.

For connectors inspect the library in use.

Most connectors provide functions which return version information.

For C - `libmysqlclient` has:

```
const char *mysql_get_client_info(void)
```

If clients have not been upgraded to 8.0.21 check the value of `local_infile`.
Execute the following SQL statement:

```
SHOW VARIABLES WHERE Variable_name = 'local_infile';
```

If clients are older than 8.0.21 or if `local_infile` is not in use, ensure the value returned is `0`.

Remediation:

Upgrade all MySQL clients and connectors to 8.0.21 or higher.

In the case where using `local_infile` is needed, the following changes further harden security:

On client side, secure by:

Limiting the location from where data can be read using `--load-data-local-dir`.

```
mysql --local-infile=0 --load-data-local-dir=/my/local/data
```

Adding TLS connection to assure server identity by requiring verification.

```
mysql --local-infile=0 --load-data-local-dir=/my/local/data --ssl-
mode=VERIFY_IDENTITY
```

If `local_infile` is not in use or if clients are not upgraded - add the following line to the `[mysqld]` section of the MySQL configuration file and restart the MySQL service:

```
local-infile=0
```

Default Value:

`0` (OFF)

References:

1. http://dev.mysql.com/doc/refman/8.0/en/string-functions.html#function_load-file
2. http://dev.mysql.com/doc/refman/8.0/en/load-data.html

CIS Controls:

Version 7

4.7 Limit Access to Script Tools
Limit access to scripting tools (such as Microsoft PowerShell and Python) to only administrative or development users with the need to access those capabilities.

4.5 Ensure 'mysqld' is Not Started With '--skip-grant-tables' (Automated)

Profile Applicability:

- Level 1 - MySQL RDBMS on Linux

Description:

This option causes `mysqld` to start without using the privilege system.

Rationale:

If this option is used, all clients of the affected server will have unrestricted access to all databases.

Audit:

Perform the following to determine if the recommended state is in place:

- Open the MySQL configuration (e.g., `my.cnf`) file and search for skip-grant-tables
- Ensure `skip-grant-tables` is set to `FALSE`

Remediation:

Perform the following to establish the recommended state:

- Open the MySQL configuration (e.g., `my.cnf`) file and set:

```
skip-grant-tables = FALSE
```

References:

1. http://dev.mysql.com/doc/refman/8.0/en/server-options.html#option_mysqld_skip-grant-tables

Additional Information:

This option has no `SHOW VARIABLES` counterpart.

CIS Controls:

Version 7

14.6 Protect Information through Access Control Lists
Protect all information stored on systems with file system, network share, claims, application, or database specific access control lists. These controls will enforce the principle that only authorized individuals should have access to the information based on their need to access the information as a part of their responsibilities.

4.6 Ensure Symbolic Links are Disabled (Automated)

Profile Applicability:

- Level 1 - MySQL RDBMS

- Level 1 - MySQL RDBMS on Linux

Description:

The `symbolic-links` and `skip-symbolic-links` options for MySQL determine whether symbolic link support is available. When use of symbolic links is enabled, they have different effects depending on the host platform. When symbolic links are disabled, then symbolic links stored in files or entries in tables are not used by the database.

Rationale:

Prevents symbolic links from being used for database files. This is especially important when MySQL is executing as root as arbitrary files may be overwritten. The `symbolic-links` option might allow someone to direct actions by the MySQL server to other files and/or directories.

Audit:

Execute the following SQL statement to assess this recommendation:

```
SHOW variables LIKE 'have_symlink';
```

Ensure the `Value` returned is `DISABLED`.

Remediation:

Perform the following actions to remediate this setting:

- Open the MySQL configuration file (`my.cnf`)
- Locate `skip_symbolic_links` in the configuration
- Set the `skip_symbolic_links` to `YES`

Note: If `skip_symbolic_links` does not exist, add it to the configuration file in the `mysqld` section.

References:

1. http://dev.mysql.com/doc/refman/8.0/en/symbolic-links.html

2. http://dev.mysql.com/doc/refman/8.0/en/server-options.html#option_mysqld_symbolic-links

CIS Controls:

Version 7

13 Data Protection
Data Protection

4.7 Ensure the 'daemon_memcached' Plugin is Disabled (Automated)

Profile Applicability:

- Level 1 - MySQL RDBMS
- Level 1 - MySQL RDBMS on Linux

Description:

The `InnoDB` `memcached` Plugin allows users to access data stored in `InnoDB` with the `memcached` protocol.

Rationale:

By default, the plugin doesn't do authentication, which means that anyone with access to the TCP/IP port of the plugin can access and modify the data. However, not all data is exposed by default.

Audit:

Execute the following SQL statement to assess this recommendation:

```
SELECT * FROM information_schema.plugins WHERE
PLUGIN_NAME='daemon_memcached';
```

Ensure that no rows are returned.

Remediation:

To remediate this setting, issue the following command in the MySQL command-line client:

```
uninstall plugin daemon_memcached;
```

This uninstalls the `memcached` plugin from the MySQL server.

Default Value:

`disabled`

References:

1. http://dev.mysql.com/doc/refman/8.0/en/innodb-memcached-security.html

CIS Controls:

Version 7

9.2 Ensure Only Approved Ports, Protocols and Services Are Running
Ensure that only network ports, protocols, and services listening on a system with validated business needs, are running on each system.

4.8 Ensure the 'secure_file_priv' is Configured Correctly (Automated)

Profile Applicability:

- Level 1 - MySQL RDBMS
- Level 1 - MySQL RDBMS on Linux

Description:

The `secure_file_priv` option restricts to paths used by `LOAD DATA INFILE` or `SELECT local_file`. It is recommended that this option be set to a file system location that contains only resources expected to be loaded by MySQL. Even better, if data import/export using `LOAD DATA INFILE` or `SELECT local_file` is not used, the functionality should be disabled entirely by setting `--secure-file-priv` to `NULL`.

Rationale:

Setting `secure_file_priv` reduces an attacker's ability to read sensitive files off the affected server via a SQL injection vulnerability.

Impact:

Solutions that rely on loading data from various sub-directories may be negatively impacted by this change. Consider consolidating load directories under a common parent directory.

The server checks the value of `secure_file_priv` at startup and writes a warning to the error log if the value is insecure. A non-NULL value is considered insecure if it is empty, or the value is the data directory or a subdirectory of it, or a directory that is accessible by all users.

Audit:

Execute the following SQL statement and ensure one row is returned:

```
SHOW GLOBAL VARIABLES WHERE Variable_name = 'secure_file_priv';
```

The Value should either contain `NULL` (thus is disabled entirely) or a valid path.

Remediation:

If you are not going to use this feature, remove `secure_file_priv` from the `[mysqld]` section of the MySQL configuration file and restart the MySQL service.

If you need this feature add the following line to the [mysqld] section of the MySQL configuration file and restart the MySQL service:

```
secure_file_priv=<path_to_load_directory>
```

Default Value:

No value set.

References:

1. http://dev.mysql.com/doc/refman/8.0/en/server-system-variables.html#sysvar_secure_file_priv

CIS Controls:

Version 7

13 Data Protection
Data Protection

4.9 Ensure 'sql_mode' Contains 'STRICT_ALL_TABLES' (Automated)

Profile Applicability:

- Level 2 - MySQL RDBMS
- Level 2 - MySQL RDBMS on Linux

Description:

When data changing statements are made (i.e., `INSERT`, `UPDATE`), MySQL can handle invalid or missing values differently depending on whether strict SQL mode is enabled. When strict SQL mode is enabled, data may not be truncated or otherwise "adjusted" to make the data changing statement work.

Rationale:

Without strict mode the server tries to proceed with the action when an error might have been a more secure choice. For example, by default MySQL will truncate data if it does not fit in a field, which can lead to unknown behavior, or be leveraged by an attacker to circumvent data validation.

Impact:

Applications relying on the MySQL database should be aware that `STRICT_ALL_TABLES` is in use, such that error conditions are handled appropriately.

Audit:

To audit for this recommendation, execute the following query:

```
SHOW VARIABLES LIKE 'sql_mode';
+---------------+------------------------------------------------------------+
| Variable_name | Value                                                      |
+---------------+------------------------------------------------------------+
| sql_mode      | ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,NO_ZERO_IN_DATE,    |
|               | NO_ZERO_DATE,ERROR_FOR_DIVISION_BY_ZERO,                   |
|               | NO_ENGINE_SUBSTITUTION                                     |
+---------------+------------------------------------------------------------+
```

If `STRICT_ALL_TABLES` is not in the list returned, this is a fail.

Remediation:

Add `STRICT_ALL_TABLES` to the `sql_mode` in the server's configuration file, for example:

```
SET GLOBAL sql_mode
='STRICT_ALL_TABLES,ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,NO_ZERO_IN_DATE,NO
_ZERO_DATE,ERROR_FOR_DIVISION_BY_ZERO,NO_ENGINE_SUBSTITUTION';
```

Default Value:

```
ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,NO_ZERO_IN_DATE,NO_ZERO_DATE,ERROR_FOR
_DIVISION_BY_ZERO,NO_ENGINE_SUBSTITUTION
```

References:

1. http://dev.mysql.com/doc/refman/8.0/en/server-sql-mode.html

Additional Information:

The `sql_mode` is a set and might contain more elements than just `STRICT_ALL_TABLES`.

There is a global `sql_mode` and a per session `sql_mode`. The per session `sql_mode` is based on the global `sql_mode` on initialization and might be changed by the application.

4.10 Use MySQL TDE for At-Rest Data Encryption (Automated)

Profile Applicability:

- Level 2 - MySQL RDBMS
- Level 2 - MySQL RDBMS on Linux

Description:

Transparent Data Encryption (TDE) at-rest encryption protects your critical data by enabling data-at-rest encryption in the database. It protects the privacy of your information, prevents data breaches and helps meet regulatory requirements.

Rationale:

File system based encryption does a good job of protecting against data theft on devices unable to limit physical access. It does not, however, protect against users who have or gain access to the operating system, backups, over the network copies, etc. Encrypting data from mysqld adds an additional layer of data protection.

Audit:

Check for the types of at-rest encryption enabled.

```
SELECT VARIABLE_NAME, VARIABLE_VALUE, 'AT REST ENCRYPTION' as Note,
IF(VARIABLE_VALUE = 'AES', 'Encrypted', 'Not Encrypted') as IsEncrypted
FROM performance_schema.global_variables where
variable_name in ('audit_log_encryption', 'binlog_encryption',
'innodb_redo_log_encrypt', 'innodb_undo_log_encrypt',
'table_encryption_privilege_check');
```

`OFF` or `NONE` indicate at-rest encryption is not enabled.

To check which tables or tablespaces are not encrypted

```
SELECT
    INNODB_TABLESPACES.NAME,
    INNODB_TABLESPACES.ENCRYPTION
FROM information_schema.INNODB_TABLESPACES
WHERE NAME NOT IN ('innodb_temporary','sys/sys_config');
```

If any tables or tablespaces show encryption as `N` and are therefore not encrypted, this is a fail.

Backup data should be encrypted at rest as well. If you are running the MySQL Enterprise

Backup verify that the backup uses `--encrypt`.

For example:

```
$ mysqlbackup --defaults-file=/home/dbadmin/my.cnf --backup-
image=/home/admin/backups/my.mbi \
  --backup-dir=/home/admin/backup-tmp --encrypt-password backup-to-image
```

If `--encrypt-password` is not included the backup is not encrypted, this is a fail.

Remediation:

Edit `my.cnf`:

```
# AUDIT LOG
sudo vi /etc/my.cnf
[mysqld]
audit-log=FORCE_PLUS_PERMANENT
audit-log-format=JSON
audit-log-encryption=AES
```

Execute these commands:

```
#### BINLOG
>set persist binlog_encryption=ON;

##### REDO and UNDO
>set persist innodb_redo_log_encrypt=ON;
>set persist innodb_undo_log_encrypt=ON;

# DO NOT USE GENERAL LOG OR SLOW LOGS - USE AUDIT AND PERFORMANCE_SCHEMA.
>SET PERSIST general_log = 'OFF';
```

Run `ALTER` to enable encryption (note will lock table as table is encrypted).

```
# TABLESPACES, TABLES
ALTER TABLESPACE <tablespacename> ENCRYPTION = 'Y';
// if innodb file per table (indicated by schemaname/tablename in report)
ALTER TABLE <tablename> ENCRYPTION = 'Y';
#Encrypt the system tablespace
ALTER TABLESPACE mysql ENCRYPTION = 'Y';
```

Run MySQL Enterprise Backup with encryption.

For example:

```
$ mysqlbackup --defaults-file=/home/dbadmin/my.cnf --backup-
image=/home/admin/backups/my.mbi \
  --backup-dir=/home/admin/backup-tmp --encrypt-password backup-to-image
```

Default Value:

At rest encryption is off by default.

Administrators can force tables or tablespaces to be encrypted for all schemas by default by setting in `my.cnf`.

```
default-table-encryption=ON
```

or Per schema by defining `DEFAULT ENCRYPTION`:

```
CREATE {DATABASE | SCHEMA} ...
  | DEFAULT ENCRYPTION [=] {'Y' | 'N'}
```

References:

1. https://dev.mysql.com/doc/refman/8.0/en/innodb-data-encryption.html
2. https://dev.mysql.com/doc/refman/8.0/en/server-system-variables.html#sysvar_table_encryption_privilege_check
3. https://dev.mysql.com/doc/refman/8.0/en/create-database.html
4. https://dev.mysql.com/doc/refman/8.0/en/replication-binlog-encryption.html
5. https://dev.mysql.com/doc/refman/8.0/en/audit-log-logging-configuration.html#audit-log-file-encryption
6. https://dev.mysql.com/doc/mysql-enterprise-backup/8.0/en/meb-encrypted-innodb.html

CIS Controls:

Version 7

14.8 Encrypt Sensitive Information at Rest
Encrypt all sensitive information at rest using a tool that requires a secondary authentication mechanism not integrated into the operating system, in order to access the information.

5 MySQL Permissions

This section contains recommendations about user privileges.

5.1 Ensure Only Administrative Users Have Full Database Access (Manual)

Profile Applicability:

- Level 1 - MySQL RDBMS
- Level 1 - MySQL RDBMS on Linux

Description:

The `mysql.user`, `mysql.db`, and other `mysql` tables ending in `_priv` list a variety of privileges that can be granted (or denied) to MySQL users. Some of the privileges of concern include: `Select_priv`, `Insert_priv`, `Update_priv`, `Delete_priv`, `Drop_priv`, and so on. Typically, these privileges should not be available to every MySQL user and often are reserved for administrative use only. The `information_schema.user_privileges` provides a consolidated view of all user privileges.

Rationale:

Limiting the accessibility of the `mysql` database will protect the confidentiality, integrity, and availability of the data housed within MySQL. A user which has direct access to `mysql.*` might view password hashes, change permissions, or alter or destroy information intentionally or unintentionally.

Audit:

Execute the following SQL statement(s) to assess this recommendation:

```
select * from information_schema.user_privileges
where grantee not like ('\'mysql.%localhost\'');
```

Ensure all users returned are administrative users with minimal privileges required.

The above query ignores MySQL internal `reserved accounts`.

Remediation:

Perform the following actions to remediate this setting:

1. Enumerate non-administrative users resulting from the audit procedure.
2. For each non-administrative user, use the `REVOKE` statement to remove privileges as appropriate.

References:

1. https://dev.mysql.com/doc/refman/8.0/en/reserved-accounts.html

Additional Information:

Consideration should be made for which privileges are required by each user requiring interactive database access.

CIS Controls:

Version 7

4.1 Maintain Inventory of Administrative Accounts
Use automated tools to inventory all administrative accounts, including domain and local accounts, to ensure that only authorized individuals have elevated privileges.

14.6 Protect Information through Access Control Lists
Protect all information stored on systems with file system, network share, claims, application, or database specific access control lists. These controls will enforce the principle that only authorized individuals should have access to the information based on their need to access the information as a part of their responsibilities.

5.2 Ensure 'file_priv' is Not Set to 'Y' for Non-Administrative Users (Manual)

Profile Applicability:

- Level 1 - MySQL RDBMS

- Level 1 - MySQL RDBMS on Linux

Description:

The `File_priv` privilege found in the `mysql.user` table is used to allow or disallow a user from reading and writing files on the server host. Any user with the `File_priv` right granted has the ability to:

- Read files from the local file system that are readable by the MySQL server (this includes world-readable files).
- Write files to the local file system where the MySQL server has write access.

Rationale:

The `File_priv` right allows `mysql` users to read files from disk and to write files to disk. This may be leveraged by an attacker to further compromise MySQL. It should be noted that the MySQL server should not overwrite existing files.

Audit:

Execute the following SQL statement to audit this setting:

```
select user, host from mysql.user where File_priv = 'Y';
```

Ensure only administrative users are returned in the result set.

Remediation:

Perform the following steps to remediate this setting:

1. Enumerate the non-administrative users found in the result set of the audit procedure.
2. For each user, issue the following SQL statement (replace *`<user>`* with the non-administrative user):

```
REVOKE FILE ON *.* FROM '<user>';
```

References:

1. http://dev.mysql.com/doc/refman/8.0/en/privileges-provided.html#priv_file

Additional Information:

See also: `secure_file_priv` system settings.

CIS Controls:

Version 7

14.6 Protect Information through Access Control Lists
Protect all information stored on systems with file system, network share, claims, application, or database specific access control lists. These controls will enforce the principle that only authorized individuals should have access to the information based on their need to access the information as a part of their responsibilities.

5.3 Ensure 'process_priv' is Not Set to 'Y' for Non-Administrative Users (Manual)

Profile Applicability:

- Level 2 - MySQL RDBMS
- Level 2 - MySQL RDBMS on Linux

Description:

The `PROCESS` privilege found in the `mysql.user` table determines whether a given user can see statement execution information for all sessions.

Rationale:

The `PROCESS` privilege allows principals to view currently executing MySQL statements beyond their own, including statements used to manage passwords. This may be leveraged by an attacker to compromise MySQL or to gain access to potentially sensitive data.

Impact:

Users denied the `PROCESS` privilege may also be denied use of `SHOW ENGINE`.

Audit:

Execute the following SQL statement to audit this setting:

```
select user, host from mysql.user where Process_priv = 'Y';
```

Ensure only administrative users are returned in the result set.

Remediation:

Perform the following steps to remediate this setting:

1. Enumerate the non-administrative users found in the result set of the audit procedure
2. For each user, issue the following SQL statement (replace `<user>` with the non-administrative user:

    ```
    REVOKE PROCESS ON *.* FROM '<user>';
    ```

References:

1. http://dev.mysql.com/doc/refman/8.0/en/privileges-provided.html#priv_process

CIS Controls:

Version 7

14.6 Protect Information through Access Control Lists
Protect all information stored on systems with file system, network share, claims, application, or database specific access control lists. These controls will enforce the principle that only authorized individuals should have access to the information based on their need to access the information as a part of their responsibilities.

5.4 Ensure 'SUPER' is Not Granted to Non-Administrative Users (Manual)

Profile Applicability:

- Level 1 - MySQL RDBMS

- Level 1 - MySQL RDBMS on Linux

Description:

The `SUPER` privilege is a powerful and far-reaching privilege and should not be granted lightly. In MySQL 8.0, `SUPER` is deprecated and will be removed in a future version of MySQL.

The `SUPER` privilege shown in the `INFORMATION_SCHEMA.USER_PRIVILEGES` table governs the use of a variety of MySQL features. These features include, `CHANGE MASTER TO`, `KILL`, `mysqladmin kill` option, `PURGE BINARY LOGS`, `SET GLOBAL`, `mysqladmin debug` option, logging control, and more.

In MySQL 8.0, `SUPER` is deprecated and will be removed in a future version of MySQL. Migrating Accounts from SUPER to Dynamic Privileges is recommended.

Rationale:

The `SUPER` privilege allows principals to perform many actions, including view and terminate currently executing MySQL statements (including statements used to manage passwords). This privilege also provides the ability to configure MySQL, such as enable/disable logging, alter data, disable/enable features. Limiting the accounts that have the `SUPER` privilege reduces the chances that an attacker can exploit these capabilities.

It is more secure to migrate administrative users off `SUPER` and instead assign the specific and minimal set of mysql Dynamic Privileges needed to perform their tasks.

Impact:

When the `SUPER` privilege is denied to a given user, that user will be unable to take advantage of certain capabilities, such as certain `mysqladmin` options.

Audit:

Execute the following SQL statement to audit this setting:

```
SELECT GRANTEE FROM INFORMATION_SCHEMA.USER_PRIVILEGES
WHERE PRIVILEGE_TYPE = 'SUPER’;
```

Ensure only administrative users are returned in the result set.

Remediation:

Perform the following steps to remediate this setting:

1. Enumerate the non-administrative users found in the result set of the audit procedure
2. For each user, issue the following SQL statement (replace *<user>* with the non-administrative user:

   ```
   REVOKE SUPER ON *.* FROM '<user>';
   ```

   Next minimize administrator rights

   1. Assess the minimal set of Dynamic Permissions needed by a user to perform their duties.
   2. For each user assign the appropriate Dynamic Permission and then revoke that `<user>` `SUPER` capability.
      For example, if administrator `'u1'@'localhost'` requires `SUPER` for binary log purging and system variable modification, these statements make the required changes to the account thus limiting rights to what is needed:

      ```
      GRANT BINLOG_ADMIN, SYSTEM_VARIABLES_ADMIN ON *.* TO
      'u1'@'localhost';
      REVOKE SUPER ON *.* FROM 'u1'@'localhost';
      ```

References:

1. http://dev.mysql.com/doc/refman/8.0/en/privileges-provided.html#priv_super
2. https://dev.mysql.com/doc/refman/8.0/en/privileges-provided.html#privileges-provided-summary

CIS Controls:

Version 7

14.6 Protect Information through Access Control Lists
Protect all information stored on systems with file system, network share, claims, application, or database specific access control lists. These controls will enforce the principle that only authorized individuals should have access to the information based on their need to access the information as a part of their responsibilities.

5.5 Ensure 'shutdown_priv' is Not Set to 'Y' for Non-Administrative Users (Manual)

Profile Applicability:

- Level 1 - MySQL RDBMS
- Level 1 - MySQL RDBMS on Linux

Description:

The `SHUTDOWN` privilege simply enables use of the `shutdown` option to the `mysqladmin` command, which allows a user with the `SHUTDOWN` privilege the ability to shut down the MySQL server.

Rationale:

The `SHUTDOWN` privilege allows principals to shutdown MySQL. This may be leveraged by an attacker to negatively impact the availability of MySQL.

Audit:

Execute the following SQL statement to audit this setting:

```
SELECT user, host FROM mysql.user WHERE Shutdown_priv = 'Y';
```

Ensure only administrative users are returned in the result set.

Remediation:

Perform the following steps to remediate this setting:

1. Enumerate the non-administrative users found in the result set of the audit procedure.
2. For each user, issue the following SQL statement (replace *`<user>`* with the non-administrative user):

```
REVOKE SHUTDOWN ON *.* FROM '<user>';
```

References:

1. http://dev.mysql.com/doc/refman/8.0/en/privileges-provided.html#priv_shutdown

CIS Controls:

Version 7

14.6 Protect Information through Access Control Lists

Protect all information stored on systems with file system, network share, claims, application, or database specific access control lists. These controls will enforce the principle that only authorized individuals should have access to the information based on their need to access the information as a part of their responsibilities.

5.6 Ensure 'create_user_priv' is Not Set to 'Y' for Non-Administrative Users (Manual)

Profile Applicability:

- Level 1 - MySQL RDBMS
- Level 1 - MySQL RDBMS on Linux

Description:

The `CREATE USER` privilege governs the right of a given user to add or remove users, change existing users' names, or revoke existing users' privileges.

Rationale:

Reducing the number of users granted the `CREATE USER` right minimizes the number of users able to add/drop users, alter existing users' names, and manipulate existing users' privileges.

Impact:

Users that are denied the `CREATE USER` privilege will not only be unable to create a user, but they may be unable to drop a user, rename a user, or otherwise revoke a given user's privileges.

Audit:

Execute the following SQL statement to audit this setting:

```
SELECT user, host FROM mysql.user WHERE Create_user_priv = 'Y';
```

Ensure only administrative users are returned in the result set.

Remediation:

Perform the following steps to remediate this setting:

1. Enumerate the non-administrative users found in the result set of the audit procedure
2. For each user, issue the following SQL statement (replace `<user>` with the non-administrative user):

```
REVOKE CREATE USER ON *.* FROM '<user>';
```

CIS Controls:

Version 7

14.6 Protect Information through Access Control Lists
Protect all information stored on systems with file system, network share, claims, application, or database specific access control lists. These controls will enforce the principle that only authorized individuals should have access to the information based on their need to access the information as a part of their responsibilities.

5.7 Ensure 'grant_priv' is Not Set to 'Y' for Non-Administrative Users (Manual)

Profile Applicability:

- Level 1 - MySQL RDBMS
- Level 1 - MySQL RDBMS on Linux

Description:

The `GRANT OPTION` privilege exists in different contexts (`mysql.user`, `mysql.db`) for the purpose of governing the ability of a privileged user to manipulate the privileges of other users.

Rationale:

The `GRANT` privilege allows a principal to grant other principals additional privileges. This may be used by an attacker to compromise MySQL.

Audit:

Execute the following SQL statements to audit this setting:

```
SELECT user, host FROM mysql.user WHERE Grant_priv = 'Y';
SELECT user, host FROM mysql.db WHERE Grant_priv = 'Y';
```

Ensure only administrative users are returned in the result set.

Remediation:

Perform the following steps to remediate this setting:

1. Enumerate the non-administrative users found in the result sets of the audit procedure
2. For each user, issue the following SQL statement (replace *`<user>`* with the non-administrative user:

```
REVOKE GRANT OPTION ON *.* FROM <user>;
```

References:

1. http://dev.mysql.com/doc/refman/8.0/en/privileges-provided.html#priv_grant-option

CIS Controls:

Version 7

14.6 Protect Information through Access Control Lists
Protect all information stored on systems with file system, network share, claims, application, or database specific access control lists. These controls will enforce the principle that only authorized individuals should have access to the information based on their need to access the information as a part of their responsibilities.

5.8 Ensure 'repl_slave_priv' is Not Set to 'Y' for Non-Replica Users (Manual)

Profile Applicability:

- Level 1 - MySQL RDBMS
- Level 1 - MySQL RDBMS on Linux

Description:

The `REPLICATION SLAVE` privilege governs whether a given user (in the context of the source server) can request updates that have been made on the source server.

Rationale:

The `REPLICATION SLAVE` privilege allows a principal to fetch `binlog` files containing all data changing statements and/or changes to table data from the source. This may be used by an attacker to read/fetch sensitive data from MySQL.

Audit:

Execute the following SQL statement to audit this setting:

```
SELECT user, host FROM mysql.user WHERE Repl_slave_priv = 'Y';
```

Ensure only accounts designated for replica users are granted this privilege.

Remediation:

Perform the following steps to remediate this setting:

1. Enumerate the non-replica users found in the result set of the audit procedure
2. For each user, issue the following SQL statement (replace *`<user>`* with the non-replica user):

   ```
   REVOKE REPLICATION SLAVE ON *.* FROM <user>;
   ```

   Use the `REVOKE` statement to remove the `SUPER` privilege from users who shouldn't have it.

References:

1. http://dev.mysql.com/doc/refman/8.0/en/privileges-provided.html#priv_replication-slave

CIS Controls:

Version 7

14.6 Protect Information through Access Control Lists
Protect all information stored on systems with file system, network share, claims, application, or database specific access control lists. These controls will enforce the principle that only authorized individuals should have access to the information based on their need to access the information as a part of their responsibilities.

5.9 Ensure DML/DDL Grants are Limited to Specific Databases and Users (Manual)

Profile Applicability:

- Level 1 - MySQL RDBMS
- Level 1 - MySQL RDBMS on Linux

Description:

DML/DDL includes the set of privileges used to modify or create data structures. This includes `INSERT`, `SELECT`, `UPDATE`, `DELETE`, `DROP`, `CREATE`, and `ALTER` privileges.

Rationale:

`INSERT`, `SELECT`, `UPDATE`, `DELETE`, `DROP`, `CREATE`, and `ALTER` are powerful privileges in any database. Such privileges should be limited only to those users requiring such rights. By limiting the users with these rights and ensuring that they are limited to specific databases, the attack surface of the database is reduced.

Audit:

Execute the following SQL statement to audit this setting:

```
SELECT User,Host,Db
FROM mysql.db
WHERE Select_priv='Y'
  OR Insert_priv='Y'
  OR Update_priv='Y'
  OR Delete_priv='Y'
  OR Create_priv='Y'
  OR Drop_priv='Y'
  OR Alter_priv='Y';
```

Ensure all users returned are permitted to have these privileges on the indicated databases.

Note: Global grants are covered in Recommendation 4.1.

Remediation:

Perform the following steps to remediate this setting:

1. Enumerate the unauthorized users, hosts, and databases returned in the result set of the audit procedure

2. For each user, issue the following SQL statement (replace *<user>* with the unauthorized user, *<host>* with host name, and *<database>* with the database name):

```
REVOKE SELECT ON <host>.<database> FROM <user>;
REVOKE INSERT ON <host>.<database> FROM <user>;
REVOKE UPDATE ON <host>.<database> FROM <user>;
REVOKE DELETE ON <host>.<database> FROM <user>;
REVOKE CREATE ON <host>.<database> FROM <user>;
REVOKE DROP ON <host>.<database> FROM <user>;
REVOKE ALTER ON <host>.<database> FROM <user>;
```

CIS Controls:

Version 7

14.6 Protect Information through Access Control Lists
Protect all information stored on systems with file system, network share, claims, application, or database specific access control lists. These controls will enforce the principle that only authorized individuals should have access to the information based on their need to access the information as a part of their responsibilities.

5.10 Securely Define Stored Procedures and Functions DEFINER and INVOKER (Manual)

Profile Applicability:

- Level 1 - MySQL RDBMS

- Level 1 - MySQL RDBMS on Linux

Description:

Stored procedure and stored function declarations include a definition of permissions which can be used to escalate permissions. It's important to inspect these settings to ensure they do not unnecessarily escalate privileges.

Rationale:

A stored procedure or function that improperly escalates privileges may provide unintended access rights which can be improperly used.

Audit:

Run the following:

```
SHOW PROCEDURE STATUS;
SHOW FUNCTION STATUS;
```

Inspect Definer and Invoker security types.

If DEFINER is a powerful user consider that user's permissions.

If INVOKER then the rights for the stored procedure or function are that of the user executing these.

Review code using

```
SHOW CREATE PROCEDURE <name>;
SHOW CREATE FUNCTION <name>;
```

For more details on Procedures and Functions

```
SELECT * FROM information_schema.routines;
```

For more details on Procedures and Functions input and output parameters.

```
SELECT * FROM information_schema.parameters;
```

Remediation:

Drop and recreate stored procedures and functions using proper DEFINER and INVOKER settings, or other code changes.

References:

1. https://dev.mysql.com/doc/refman/8.0/en/create-procedure.html

CIS Controls:

Version 7

14.6 Protect Information through Access Control Lists
Protect all information stored on systems with file system, network share, claims, application, or database specific access control lists. These controls will enforce the principle that only authorized individuals should have access to the information based on their need to access the information as a part of their responsibilities.

6 Auditing and Logging

This section provides guidance with respect to MySQL's logging behavior.

6.1 Ensure 'log_error' is Not Empty (Automated)

Profile Applicability:

- Level 1 - MySQL RDBMS

- Level 1 - MySQL RDBMS on Linux

Description:

The error log contains information about events such as `mysqld` starting and stopping, when a table needs to be checked or repaired, and, depending on the host operating system, stack traces when `mysqld` fails.

Rationale:

Enabling error logging may increase the ability to detect malicious attempts against MySQL, and other critical messages, such as if the error log is not enabled then connection error might go unnoticed.

Audit:

Execute the following SQL statement to audit this setting:

```
SHOW variables LIKE 'log_error';
```

Ensure the `Value` returned is not empty.

Remediation:

Perform the following actions to remediate this setting:

1. Open the MySQL configuration file (`my.cnf` or `my.ini`).
2. Set the `log-error` option to the path for the error log.

References:

1. http://dev.mysql.com/doc/refman/8.0/en/error-log.html

CIS Controls:

Version 7

6.2 Activate audit logging
Ensure that local logging has been enabled on all systems and networking devices.

6.2 Ensure Log Files are Stored on a Non-System Partition (Automated)

Profile Applicability:

- Level 1 - MySQL RDBMS on Linux

Description:

MySQL log files can be set in the MySQL configuration to exist anywhere on the filesystem. It is common practice to ensure that the system filesystem is left uncluttered by application logs. System filesystems include the root, `/var`, or `/usr`.

Rationale:

Moving the MySQL logs off the system partition will reduce the probability of denial of service via the exhaustion of available disk space to the operating system.

Audit:

Execute the following SQL statement to assess this recommendation:

```
SELECT @@global.log_bin_basename;
```

Ensure the value returned does not indicate root (`/`), `/var`, or `/usr`.

Remediation:

Perform the following actions to remediate this setting:

1. Open the MySQL configuration file (`my.cnf`)
2. Locate the `log-bin` entry and set it to a file not on root (`/`), `/var`, or `/usr`

References:

1. http://dev.mysql.com/doc/refman/8.0/en/binary-log.html
2. http://dev.mysql.com/doc/refman/8.0/en/replication-options-binary-log.html

CIS Controls:

Version 7

6.4 Ensure adequate storage for logs
Ensure that all systems that store logs have adequate storage space for the logs generated.

6.3 Ensure 'log_error_verbosity' is Set to '2' (Automated)

Profile Applicability:

- Level 2 - MySQL RDBMS

- Level 2 - MySQL RDBMS on Linux

Description:

The `log_error_verbosity` system variable, set to `2` by default, specifies the verbosity of events sent to the MySQL error log. A value of `2` enables logging of error and warning messages, a value of `3` also includes informational logging, a value of `1` logs only errors.

Rationale:

This might help to detect malicious behavior by logging communication errors and aborted connections.

Audit:

Execute the following SQL statement to assess this recommendation:

```
SHOW GLOBAL VARIABLES LIKE 'log_error_verbosity';
```

Ensure the `Value` returned equals `2`.

Remediation:

Perform the following actions to remediate this setting:

- Open the MySQL configuration file (`my.cnf`)
- Ensure the following line is found in the `mysqld` section

```
log_error_verbosity = 2
```

Default Value:

The option is enabled (`2`) - errors and warning events are logged - by default.

References:

1. https://dev.mysql.com/doc/refman/8.0/en/server-system-variables.html#sysvar_log_error_verbosity

CIS Controls:

Version 7

6.3 Enable Detailed Logging

Enable system logging to include detailed information such as an event source, date, user, timestamp, source addresses, destination addresses, and other useful elements.

6.4 Ensure 'log-raw' is Set to 'OFF' (Automated)

Profile Applicability:

- Level 1 - MySQL RDBMS on Linux

Description:

The `log-raw` MySQL option determines whether passwords are rewritten by the server so as not to appear in log files as plain text. If `log-raw` is enabled, then passwords are written to the various log files (`general query log`, `slow query log`, and `binary log`) in plain text.

Rationale:

With raw logging of passwords enabled someone with access to the log files might see plain text passwords.

Audit:

Perform the following actions to assess this recommendation:

- Open the MySQL configuration file (`my.cnf`)
- Ensure the `log-raw` entry is present
- Ensure the `log-raw` entry is set to `OFF`

Remediation:

Perform the following actions to remediate this setting:

- Open the MySQL configuration file (`my.cnf`)
- Find the `log-raw` entry and set it as follows

```
log-raw = OFF
```

Default Value:

`OFF`

References:

1. https://dev.mysql.com/doc/refman/8.0/en/password-logging.html
2. https://dev.mysql.com/doc/refman/8.0/en/server-options.html#option_mysqld_log-raw

CIS Controls:

Version 7

13.2 Remove Sensitive Data or Systems Not Regularly Accessed by Organization
Remove sensitive data or systems not regularly accessed by the organization from the network. These systems shall only be used as stand alone systems (disconnected from the network) by the business unit needing to occasionally use the system or completely virtualized and powered off until needed.

6.5 Ensure Audit Filters Capture Connection Attempts (Manual)

Profile Applicability:

- Level 1 - MySQL RDBMS
- Level 1 - MySQL RDBMS on Linux

Description:

The functions `audit_log_filter_set_filter()` and `audit_log_filter_set_user()` are used to define rules for auditing. With this feature you can easily audit successful and/or failed connection events and write to the audit log file.

Rationale:

The `audit_log_filter_set_filter` function which defines auditing filters. The users for which filter(s) apply is defined by `audit_log_filter_set_user`. One or more filters can be created to log connections success and/or failure.

Impact:

If the audit rule and application of the rule to targeted or all users is not properly configured, it will not log failed connections, successful connections or any other connection related events.

Audit:

To assess this recommendation, execute the following SQL statements:
Evaluate the filters:

```
SELECT * FROM mysql.audit_log_filter;
```

and to which accounts the filters are assigned:

```
SELECT * FROM mysql.audit_log_user;
```

Determine whether the filters and users assigned to filter meet your security, business, and regulatory requirements. If they do not, this is a fail.

Test your filters by attempting successful and failed connections or other events that should be captured in the audit trail and review the audit log to confirm those events were captured.

For example:

Successful Connections will have json fields:

"class": "connection", "event": "connect"

Failed Connections will have a failed status value - "status": 1045

```
{ "timestamp": "2020-03-19 20:13:40", "id": 0, "class": "connection",
"event": "connect", "connection_id": 30, "account": { "user": "", "host":
"localhost" }, "login": { "user": "app_developer", "os": "", "ip":
"127.0.0.1", "proxy": "" }, "connection_data": { "connection_type": "ssl",
"status": 1045, "db": "" } }
```

Successful Connections will show a successful status value - "status": 0

```
{ "timestamp": "2020-03-19 21:04:13", "id": 1, "class": "connection",
"event": "connect", "connection_id": 38, "account": { "user": "newuser",
"host": "localhost" }, "login": { "user": "newuser", "os": "", "ip":
"127.0.0.1", "proxy": "" }, "connection_data": { "connection_type": "ssl",
"status": 0, "db": "", "connection_attributes": { "_pid": "4971", "_os":
"macos10.14", "_platform": "x86_64", "_client_version": "8.0.18",
"_client_name": "libmysql", "program_name": "MySQLWorkbench" } } },
```

Remediation:

To remediate this configuration setting, execute one of the following SQL statements:
Log All connections – Successful and Failed:

```
SET @f = '{ "filter": { "class": { "name": "connection" } } }';
SELECT audit_log_filter_set_filter('log__all_conn_events', @f);
SELECT audit_log_filter_set_user('%', 'log_all_conn_events');
```

Or Log Only Failed Connections:

```
SET @f='
{
  "filter": {
    "log": false,
    "class": {
        "name": "connection",
        "event": [
          { "name": "connect", "log" : { "not": { "field": { "name":
"status", "value": 0 } } } },
          { "name": "disconnect", "log": false }
        ]
    }
 }
 }';
select @f;
SELECT audit_log_filter_set_filter('log_conn_events', @f);
SELECT audit_log_filter_set_user('%', 'log_conn_events');
```

Default Value:

The default value for `audit_log_connection_policy` is `ALL`.

References:

1. https://dev.mysql.com/doc/refman/8.0/en/audit-log-filtering.html
2. https://dev.mysql.com/doc/refman/8.0/en/audit-log.html

Additional Information:

Prior legacy modes of defining audit filters, although simple to use, were not specific enough to define precise auditing rules - and thus required too much storage - resulting in "over" auditing. Additionally MySQL audit filters can not only log events but act as firewall rules by using an abort() definition in a filter.

See:

https://dev.mysql.com/doc/refman/8.0/en/audit-log-legacy-filtering.html

CIS Controls:

Version 7

6.3 Enable Detailed Logging
Enable system logging to include detailed information such as an event source, date, user, timestamp, source addresses, destination addresses, and other useful elements.

6.6 Ensure ALL Events are Audited (Automated)

Profile Applicability:

- Level 2 - MySQL RDBMS
- Level 2 - MySQL RDBMS on Linux

Description:

This filter defines all events to be written to the audit log.

Rationale:

This filter to log all, and binding to all, users must be set to ensure all event information is written to the audit log.

Impact:

Logging all events can result in very large audit files. In the case where the database is extremely active it may be more appropriate to be more selective when defining audit filters.

Audit:

Show all filters.

```
SELECT * FROM mysql.audit_log_filter;
```

Must return a filter name with a filter defined to log everything, for example:

```
+---------+---------------------------+
| NAME    | FILTER                    |
+---------+---------------------------+
| log_all | {"filter": {"log": true}} |
+---------+---------------------------+
```

Show users to filter binding.

```
SELECT * FROM mysql.audit_log_user;
```

Ensure the filter to `log_all` (name can vary) is applied to all users.

```
+------+------+------------+
| USER | HOST | FILTERNAME |
+------+------+------------+
| %    |      | log_all    |
+------+------+------------+
```

Remediation:

Create Log All Filter:

```
SELECT audit_log_filter_set_filter('log_all', '{ "filter": { "log": true }
}');
```

Apply to all logins:

```
SELECT audit_log_filter_set_user('%', 'log_all');
```

References:

1. https://dev.mysql.com/doc/refman/8.0/en/audit-log-filtering.html

Additional Information:

If an appropriate filter and binding of the filter to users is not in place then audit events related to all events won't be written to the audit log file.

CIS Controls:

Version 7

6.3 Enable Detailed Logging
Enable system logging to include detailed information such as an event source, date, user, timestamp, source addresses, destination addresses, and other useful elements.

6.7 Set audit_log_strategy to SYNCHRONOUS or SEMISYNCRONOUS (Automated)

Profile Applicability:

- Level 2 - MySQL RDBMS
- Level 2 - MySQL RDBMS on Linux

Description:

The `audit_log_strategy` must be set to `SYNCHRONOUS` or `SEMISYNCHRONOUS`

Rationale:

This setting controls how information is written to the audit log. It can be set to `SYNCHRONOUS` to make it fully durable or other settings which are less durable but have less performance overhead.

Impact:

If this setting is set to `PERFORMANCE` or `ASYNCHRONOUS` audit events might be lost in case of a crash or when the server somehow can't write to the audit log file.

Audit:

To assess this recommendation, execute the following SQL statement:

```
SHOW GLOBAL VARIABLES LIKE 'audit_log_strategy';
```

The result should be `SYNCHRONOUS` or `SEMISYNCHRONOUS`

Remediation:

To remediate this configuration:

1. Open the MySQL configuration file (`my.cnf`)
2. Navigate to the `mysqld` section of the configuration file
3. Set `audit_log_strategy='SEMISYNCHRONOUS'` (or `SYNCHRONOUS`)

Default Value:

`ASYNCHRONOUS`

References:

1. https://dev.mysql.com/doc/refman/8.0/en/audit-log-reference.html#sysvar_audit_log_strategy

Additional Information:

Also consider the load strategy.

```
audit-log= ON, FORCE, or FORCE_PLUS_PERMANENT
```

For example, most secure is to set `--audit-log=FORCE_PLUS_PERMANENT`

This tells the server to load the plugin and prevent it from being removed while the server is running.

6.8 Ensure the Audit Plugin Can't be Unloaded (Automated)

Profile Applicability:

- Level 1 - MySQL RDBMS
- Level 1 - MySQL RDBMS on Linux

Description:

Set `audit_log` to `FORCE_PLUS_PERMANENT`

Rationale:

This disables unloading on the plugin.

Impact:

If someone can unload the plugin it would be possible to perform actions on the database without audit events being logged to the audit log. If the audit log plugin can be unloaded the audit log can be temporarily or permanently disabled.

Audit:

To assess this recommendation, execute the following SQL statement:

```
SELECT LOAD_OPTION FROM information_schema.plugins WHERE
PLUGIN_NAME='audit_log';
```

The result must be `FORCE_PLUS_PERMANENT`

Remediation:

To remediate this setting, follow these steps:

1. Open the MySQL configuration file (`my.cnf`)
2. Ensure the following line is found in the `mysqld` section

   ```
   audit_log = 'FORCE_PLUS_PERMANENT'
   ```

Default Value:

`ON`

References:

1. https://dev.mysql.com/doc/refman/8.0/en/audit-log-reference.html#option_mysqld_audit-log

CIS Controls:

Version 7

6.2 Activate audit logging
Ensure that local logging has been enabled on all systems and networking devices.

7 Authentication

This section contains configuration recommendations that pertain to the authentication mechanisms of MySQL.

7.1 Ensure default_authentication_plugin is Set to a Secure Option (Automated)

Profile Applicability:

- Level 1 - MySQL RDBMS

- Level 1 - MySQL RDBMS on Linux

Description:

The `–default-authentication-plugin` system variable governs two things:

- Authentication plugin used by a new user account if a plugin is not specified explicitly through `CREATE USER` statement
- Initial authentication data payload generated by server in case of a new connection.

Caching SHA-2 Authentication is the new default in MySQL 8.0. It provides stronger password protection than the prior Native Authentication and provides better performance than SHA2 Authentication. Alternatively, there are additional methods to securely connect using Lightweight Directory Access Protocol (LDAP) and Active Directory authentication.

Rationale:

MySQL Native Authentication relies on the Secure Hash Algorithm 1 (SHA1) algorithm and the National Institute of Standards and Technology (NIST) has suggested to stop using it.

The MySQL Native Authentication plugin leverages this weak hashing algorithm that can be quickly brute forced.

Audit:

Execute the following SQL statement to assess this recommendation:

```
SHOW VARIABLES WHERE Variable_name = 'default_authentication_plugin';
```

Ensure the `Value` field is not set to `mysql_native_password`.

Remediation:

Configure mysql to default to the `caching_sha2_password` plugin.

Require `caching_sha2_password` plugin to be used by default for new accounts.

Edit `my.cnf`, in the section `[mysqld]` add:

```
default_authentication_plugin=caching_sha2_password
```

Determine if any users are using `mysql_native_password`.

```
select host, user, plugin from mysql.user;
```

Migrate these users from `mysql_native_password`.

```
ALTER USER user
  IDENTIFIED WITH caching_sha2_password IDENTIFIED BY RANDOM PASSWORD
PASSWORD EXPIRE;
```

Provide users the random password value through a secure mechanism - on next login they will be forced to change the password.

Default Value:

New users default to `caching_sha2_password`. Migrated users will initially be `mysql_native` or other authentication method.

References:

1. https://dev.mysql.com/doc/refman/8.0/en/upgrading-from-previous-series.html#upgrade-caching-sha2-password-compatibility-issues
2. https://dev.mysql.com/doc/refman/8.0/en/authentication-plugins.html

CIS Controls:

Version 7

16.4 Encrypt or Hash all Authentication Credentials
Encrypt or hash with a salt all authentication credentials when stored.

7.2 Ensure Passwords are Not Stored in the Global Configuration (Automated)

Profile Applicability:

- Level 1 - MySQL RDBMS on Linux

Description:

The `[client]` section of the MySQL configuration file allows setting a `user` and `password` to be used. Verify the `password` option is not used in the global configuration file (`my.cnf`).

Rationale:

Using the `password` parameter may negatively impact the confidentiality of the user's password.

Impact:

The global configuration is by default readable for all users on the system. This is needed for global defaults (prompt, port, socket, etc.). If a password is present in this file then all users on the system may be able to access it.

Audit:

To assess this recommendation, perform the following steps:

- Open the MySQL configuration file (e.g., `my.cnf`)
- Examine the `[client]` section of the MySQL configuration file and ensure `password` is not employed.

Remediation:

Use the `mysql_config_editor` to store authentication credentials in `.mylogin.cnf` in encrypted form.
If not possible, use the user-specific options file, `.my.cnf.`, and restricting file access permissions to the user identity.

References:

1. https://dev.mysql.com/doc/refman/8.0/en/mysql-config-editor.html

Additional Information:

There must not be a password in any of the sections of the global configuration.

CIS Controls:

Version 7

16.4 Encrypt or Hash all Authentication Credentials
Encrypt or hash with a salt all authentication credentials when stored.

7.3 Ensure Passwords are Set for All MySQL Accounts (Automated)

Profile Applicability:

- Level 1 - MySQL RDBMS

- Level 1 - MySQL RDBMS on Linux

Description:

Blank passwords allow a user to login without using a password.

Rationale:

Without a password only knowing the username and the list of allowed hosts will allow someone to connect to the server and assume the identity of the user. This, in effect, bypasses authentication mechanisms.

Audit:

Execute the following SQL query to determine if any users have a blank password:

```
SELECT User,host
FROM mysql.user
WHERE (plugin IN('mysql_native_password', 'mysql_old_password',''))
 AND (LENGTH(authentication_string) = 0
 OR authentication_string IS NULL))
 OR (plugin='sha256_password' AND LENGTH(authentication_string) = 0);
```

No rows will be returned if all accounts have a password set.

Remediation:

For each row returned from the audit procedure, reset the password for the given user using the following statement (as an example):

```
ALTER USER
       <user>@<host> IDENTIFIED BY RANDOM PASSWORD PASSWORD EXPIRE;
```

This resets the password temporarily to a RANDOM string and returns that temporary password as a result.

The user can then use this temporary password to login and is forced to set the password to one of their choosing upon login.

Note: Replace `<user>`, `<host>` with appropriate values.

CIS Controls:

Version 7

4.4 Use Unique Passwords
Where multi-factor authentication is not supported (such as local administrator, root, or service accounts), accounts will use passwords that are unique to that system.

7.4 Set 'default_password_lifetime' to Require a Yearly Password Change (Automated)

Profile Applicability:

- Level 1 - MySQL RDBMS
- Level 1 - MySQL RDBMS on Linux

Description:

Password expiry provides passwords with a time bounded lifetime.

Rationale:

The 'default_password_lifetime' global variable prevents a password being set for an indefinite period. Excessive password expiration requirements do more harm than good, because these requirements make users select predictable passwords, composed of sequential words and numbers that are closely related to each other. More importantly, when events occur that could compromise password security account passwords should be expired immediately.

Impact:

Scripted clients or users dependent on automated login in a controlled environment will need to consider their authentication procedures. The server will accept the user but the user is placed in restricted mode. In restricted mode, operations performed within the session result in an error until the user establishes a new account password.

Audit:

```
SHOW VARIABLES LIKE 'default_password_lifetime';
```

The result should not be `0` and less than or equal to `365`.

Remediation:

To remediate this recommendation, execute the following command:

```
SET GLOBAL default_password_lifetime=365
```

Default Value:

`360`

References:

1. https://dev.mysql.com/doc/refman/8.0/en/password-management.html
2. https://dev.mysql.com/doc/refman/8.0/en/expired-password-handling.html

Additional Information:

Research: Is it true, upon connection, a client can reset its own password after expiry regardless of the variable `disconnect_on_expired_password`? https://dev.mysql.com/doc/refman/5.7/en/server-system-variables.html#sysvar_disconnect_on_expired_password `disconnect_on_expired_password` is set to `ON` by default, so therefore, doesn't place the user into sandbox mode and should just disconnect the user. The sandbox mode allows the user the opportunity to change their own password.

CIS Controls:

Version 7

16.10 Ensure All Accounts Have An Expiration Date
Ensure that all accounts have an expiration date that is monitored and enforced.

7.5 Ensure Password Complexity Policies are in Place (Automated)

Profile Applicability:

- Level 1 - MySQL RDBMS on Linux

Description:

Password complexity includes password characteristics such as length, case, numerical, and character sets.

Rationale:

Complex passwords help mitigate dictionary, brute forcing, and other password attacks. This recommendation prevents users from choosing weak passwords which can easily be guessed.

Impact:

Remediation for this recommendation requires a server restart.

Audit:

Execute the following to see if the password validation component is installed:

```
select * from mysql.component where component_urn like '%validate_password';
```

If no rows are returned, this check fails.

Execute the following SQL statements to assess this recommendation:

```
SHOW VARIABLES LIKE 'validate_password%';
```

The result set from the above statement should show:

- `validate_password_length` should be `14` or more
- `validate_password_check_user_name` should be `ON`
- `validate_password_policy` should be `STRONG` - checks length; numeric, lowercase/uppercase, and special characters; dictionary file

New passwords should be checked against a dictionary file that contains values known to be commonly-used, expected, or compromised. For example, the list should include, but is not limited to:

- Passwords obtained from previous breaches

- Dictionary words
- Repetitive or sequential characters (e.g., `aaaaaa`, `1234abcd`)
- Context-specific words, such as the name of the service, the username, and derivatives thereof
- `validate_password_dictionary_file` should point to a dictionary file of common words used in passwords.

The following may make the password complexity too difficult, use sparingly.

- `validate_password_mixed_case_count` not more than `1`
- `validate_password_number_count` not more than `1`
- `validate_password_special_char_count` not more than `1`

The following lines should be present in the global configuration:

```
plugin-load=validate_password.so
validate_password=FORCE_PLUS_PERMANENT
```

Remediation:

Add to the global configuration:

```
plugin-load=validate_password.so
validate_password=FORCE_PLUS_PERMANENT
validate_password_length=14
validate_password_check_user_name=ON
validate_password_dictionary_file=<path to dictionary file>
validate_password_policy=STRONG
```

Optionally set one or more of these - ensuring complexity is not overly onerous

```
validate_password_mixed_case_count=1
validate_password_number_count=1
validate_password_special_char_count=1
```

And change passwords for users which have passwords which are identical to their username.

Default Value:

Default `component_validate_password` is not installed.

```
validate_password_length=8
validate_password_mixed_case_count=1
validate_password_number_count=1
validate_password_policy=MEDIUM
validate_password_special_char_count=1
```

References:

1. https://dev.mysql.com/doc/refman/8.0/en/validate-password.html

CIS Controls:

Version 7

4.4 Use Unique Passwords
Where multi-factor authentication is not supported (such as local administrator, root, or service accounts), accounts will use passwords that are unique to that system.

7.6 Ensure No Users Have Wildcard Hostnames (Automated)

Profile Applicability:

- Level 1 - MySQL RDBMS
- Level 1 - MySQL RDBMS on Linux

Description:

MySQL can make use of host wildcards when granting permissions to users on specific databases. For example, you may grant a given privilege to `'<user>'@'%'`.

Rationale:

Avoiding the use of wildcards within hostnames helps control the specific locations from which a given user may connect to and interact with the database.

Audit:

Execute the following SQL statement to assess this recommendation:

```
SELECT user, host FROM mysql.user WHERE host = '%';
```

Ensure no rows are returned.

Remediation:

Perform the following actions to remediate this setting:

1. Enumerate all users returned after running the audit procedure.
2. Either `ALTER` the user's host to be specific or `DROP` the user.

CIS Controls:

Version 7

14.6 Protect Information through Access Control Lists
Protect all information stored on systems with file system, network share, claims, application, or database specific access control lists. These controls will enforce the principle that only authorized individuals should have access to the information based on their need to access the information as a part of their responsibilities.

7.7 Ensure No Anonymous Accounts Exist (Automated)

Profile Applicability:

- Level 1 - MySQL RDBMS
- Level 1 - MySQL RDBMS on Linux

Description:

Anonymous accounts are users with empty usernames (''). Anonymous accounts have no passwords, so anyone can use them to connect to the MySQL server.

Rationale:

Removing anonymous accounts will help ensure that only identified and trusted principals are capable of interacting with MySQL.

Impact:

Any applications relying on anonymous database access will be adversely affected by this change.

Audit:

Execute the following SQL query to identify anonymous accounts:

```
SELECT user,host FROM mysql.user WHERE user = '';
```

The above query will return zero rows if no anonymous accounts are present.

Remediation:

Perform the following actions to remediate this setting:

1. Enumerate the anonymous users returned from executing the audit procedure.
2. For each anonymous user, `DROP` or assign them a name.

Note: As an alternative, you may execute the `mysql_secure_installation` utility.

References:

1. https://dev.mysql.com/doc/refman/8.0/en/mysql-secure-installation.html
2. https://dev.mysql.com/doc/refman/8.0/en/default-privileges.html

3. https://dev.mysql.com/doc/refman/8.0/en/proxy-users.html#proxy-users-conflicts

CIS Controls:

Version 7

16.8 Disable Any Unassociated Accounts
Disable any account that cannot be associated with a business process or business owner.

8 Network

This section contains recommendations related to how the MySQL server uses the network.

8.1 Ensure 'require_secure_transport' is Set to 'ON' and/or 'have_ssl' is Set to 'YES' (Automated)

Profile Applicability:

- Level 1 - MySQL RDBMS
- Level 1 - MySQL RDBMS on Linux

Description:

All network traffic must use SSL/TLS when traveling over untrusted networks.

Rationale:

The SSL/TLS-protected MySQL protocol helps to prevent eavesdropping and man-in-the-middle attacks.

Impact:

Enabling Secure Sockets Layer (SSL) will allow clients to encrypt network traffic and verify the identity of the server. This could have impact on network traffic inspection.

Audit:

Execute the following SQL statements to assess this recommendation:

Check the global default.

```
select @@require_secure_transport;
```

Ensure the returned `Value` is `ON` or '1'

```
SHOW variables WHERE variable_name = 'have_ssl';
```

Or if MySQL is built with OpenSSL:

```
SHOW variables WHERE variable_name = 'have_openssl';
```

Ensure the Value returned is `YES`.

Note: `have_openssl` is an alias for `have_ssl` when MySQL is built with OpenSSL.

Remediation:

Follow the procedures as documented in the MySQL 8.0 Reference Manual to setup SSL. Set global policy to force SSL for all connections:

```
set persist require_secure_transport=ON;
```

Default Value:

`DISABLED`

References:

1. https://dev.mysql.com/doc/refman/8.0/en/using-encrypted-connections.html
2. https://dev.mysql.com/doc/refman/8.0/en/connection-options.html
3. https://dev.mysql.com/doc/refman/8.0/en/server-system-variables.html#sysvar_require_secure_transport

CIS Controls:

Version 7

14.4 Encrypt All Sensitive Information in Transit
Encrypt all sensitive information in transit.

8.2 Ensure 'ssl_type' is Set to 'ANY', 'X509', or 'SPECIFIED' for All Remote Users (Automated)

Profile Applicability:

- Level 1 - MySQL RDBMS
- Level 1 - MySQL RDBMS on Linux

Description:

All network traffic must use SSL/TLS when traveling over untrusted networks.

SSL/TLS should be enforced on a per-user basis for users which enter the system through the network.

Rationale:

The SSL/TLS-protected MySQL protocol helps to prevent eavesdropping and man-in-the-middle attacks.

Impact:

When SSL/TLS is enforced then clients which do not use SSL will not be able to connect. If the server is not configured for SSL/TLS then accounts for which SSL/TLS is mandatory will not be able to connect.

Audit:

Execute the following SQL statements to assess this recommendation:

```
SELECT user, host, ssl_type FROM mysql.user
WHERE NOT HOST IN ('::1', '127.0.0.1', 'localhost');
```

Ensure the `ssl_type` for each user returned is equal to `X509`, or `SPECIFIED`.

Note: `ANY` means the connection must be using TLS and could optionally provide a client-side certificate.

Remediation:

Use the ALTER USER statement to require the use of SSL:

```
ALTER USER 'my_user'@'app1.example.com' REQUIRE X509;
```

Note: `REQUIRE SSL` only enforces SSL. There are additional options `REQUIRE ISSUER`, `REQUIRE SUBJECT` which can be used to further restrict the connection.

Default Value:

On the server-side SSL is `ON` by default `--ssl` (permits but does not require secure connections) and `require_secure_transport` is `OFF` (turning `ON` allows only secure connections)

References:

1. https://dev.mysql.com/doc/refman/8.0/en/using-encrypted-connections.html
2. https://dev.mysql.com/doc/refman/8.0/en/alter-user.html#alter-user-tls
3. https://dev.mysql.com/doc/refman/8.0/en/connection-options.html#option_general_ssl

CIS Controls:

Version 7

14.4 Encrypt All Sensitive Information in Transit
Encrypt all sensitive information in transit.

8.3 Set Maximum Connection Limits for Server and per User (Manual)

Profile Applicability:

- Level 1 - MySQL RDBMS
- Level 1 - MySQL RDBMS on Linux

Description:

Limiting concurrent connections to a MySQL server can be used to reduce risk of Denial of Service (DoS) attacks performed by exhausting connection resources.

Rationale:

Limiting the number of concurrent sessions at the server and per user level helps to reduce the risk of DoS attacks. MySQL provides mechanisms to limit the number of simultaneous connections that can be made at the server level or by any given account.

Audit:

To check global (default) concurrent-sessions settings in the MySQL database server, to check the per user default run the query:

```
SELECT VARIABLE_NAME, VARIABLE_VALUE
FROM performance_schema.global_variables
WHERE VARIABLE_NAME LIKE 'max_%connections';
```

If the value of `max_user_connections` is `0` this means there is “no limit”.

If the value of `max_connections` is not set, there is no limit.

Also check the values on a per user basis run the following

```
select user, host, max_user_connections from mysql.user where user not like
'mysql.%' and user not like 'root';
```

If the value is `0` this means the global value of `max_user_connections` applies.

If no limits are configured this is a fail.

Remediation:

Connect to the MySQL Database as an administrator

For example, to set the global default per user to `50` run the command:

```
SET PERSIST max_user_connections=50;
```

To control the maximum number of clients the server permits to connect simultaneously, set the max_connections system variable:

```
SET PERSIST max_connections=1000;
```

Additionally, this max user connections can be set per user as well as for a given period of time period using CREATE or ALTER

For example:

```
ALTER USER 'fred'@'localhost'
WITH MAX_CONNECTIONS_PER_HOUR 5
MAX_USER_CONNECTIONS 2;
```

Default Value:

The default value of max_connections is 151, max_user_connections is 0 (unlimited, thus limited by max_connections).

References:

1. https://dev.mysql.com/doc/refman/8.0/en/user-resources.html
2. https://dev.mysql.com/doc/refman/8.0/en/connection-interfaces.html#connection-interfaces-volume-management

9 Replication

Everything related to replicating data from one server to another.

9.1 Ensure Replication Traffic is Secured (Manual)

Profile Applicability:

- Level 1 - MySQL RDBMS on Linux

Description:

The replication traffic between servers should be secured. Security measures should include ensuring the confidentiality and integrity of the traffic, and performing mutual authentication between the servers before performing replication.

Rationale:

The replication traffic should be secured as it gives access to all transferred information and might leak passwords.

Impact:

When the replication traffic is not secured someone might be able to capture passwords and other sensitive information when sent to the replica.

Audit:

Check if the replication traffic is using one or more of the following to provide confidentiality and integrity for the traffic, and mutual authentication for the servers:

- A private network
- A VPN
- SSL/TLS
- A SSH Tunnel

Remediation:

Secure the network traffic using one or more technologies to provide confidentiality and integrity for the traffic, and mutual authentication for the servers.

CIS Controls:

Version 7

14.4 Encrypt All Sensitive Information in Transit
Encrypt all sensitive information in transit.

9.2 Ensure 'SOURCE_SSL_VERIFY_SERVER_CERT' is Set to 'YES' or '1' (Automated)

Profile Applicability:

- Level 1 - MySQL RDBMS
- Level 1 - MySQL RDBMS on Linux

Description:

In the MySQL `REPLICA` (`SLAVE` is deprecated as of 8.0.22) context the setting `SOURCE_SSL_VERIFY_SERVER_CERT` (`MASTER_SSL_VERIFY_SERVER_CERT` is deprecated as of 8.0.22) indicates whether the `REPLICA` should verify the `SOURCE`'s certificate. This configuration item may be set to `Yes` or `No`, and unless SSL has been enabled on the `REPLICA`, the value will be ignored.

Rationale:

When SSL is in use certificate verification is important to authenticate the party to which a connection is being made. In this case, the `REPLICA` (client) should verify the SOURCE's (server's) certificate to authenticate the `SOURCE` prior to continuing the connection.

Impact:

When using `CHANGE REPLICATION SOURCE TO`, (`CHANGE MASTER` is deprecated as of 8.0.23) be aware of the following:

- `REPLICA` processes need to be stopped prior to executing `CHANGE SOURCE TO`
- Use of `CHANGE REPLICATION SOURCE TO` starts new relay logs without keeping the old ones unless explicitly told to keep them
- When `CHANGE REPLICATION SOURCE TO` is invoked, some information is dumped to the error log (previous values for `SOURCE_HOST`, `SOURCE_PORT`, `SOURCE_LOG_FILE`, and `SOURCE_LOG_POS`)
- Invoking `CHANGE REPLICATION SOURCE TO` will implicitly commit any ongoing transactions in the session where the `CHANGE REPLICATION SOURCE` was run, but not all ongoing transactions on the database.

Audit:

To assess this recommendation, issue the following statement:

For releases from 8.0.23:

```
select ssl_verify_server_cert from mysql.replica_source_info;
```

The term slave is changed to replica and deprecated in 8.0.23

For releases prior to 8.0.23, run the statement:

```
select ssl_verify_server_cert from mysql.slave_source_info;
```

Verify the value of `ssl_verify_server_cert` is `1`.

Remediation:

To remediate this setting, you must use the `CHANGE SOURCE TO` command.

From 8.0.23:

```
STOP REPLICA; -- required if replication was already running
CHANGE REPLICATION SOURCE TO SOURCE_SSL_VERIFY_SERVER_CERT=1;
START REPLICA; -- required if you want to restart replication
```

Prior to 8.0.23:

```
STOP SLAVE; -- required if replication was already running
CHANGE MASTER TO MASTER_SSL_VERIFY_SERVER_CERT=1;
START SLAVE; -- required if you want to restart replication
```

References:

1. https://dev.mysql.com/doc/refman/8.0/en/change-replication-source-to.html
2. https://dev.mysql.com/doc/refman/8.0/en/change-master-to.html

9.3 Ensure 'source_info_repository' is Set to 'TABLE' (Automated)

Profile Applicability:

- Level 2 - MySQL RDBMS
- Level 2 - MySQL RDBMS on Linux

Description:

The `source_info_repository` (`master_info_repository` is deprecated as of 8.0.22) setting determines to where a `REPLICA` (`slave` is deprecated as of 8.0.22) logs `SOURCE` status and connection information. The options are `FILE` or `TABLE`. Note also that this setting is associated with the `sync_source_info` (`sync_master_info` is deprecated in 8.0.22) setting as well.

Rationale:

The password which the client uses is stored in the `SOURCE` info repository, which by default is a plaintext file. The `TABLE` `SOURCE` info repository is a bit safer, but with filesystem access it's still possible to gain access to the password the `REPLICA` is using.

Audit:

Execute the following SQL statement to assess this recommendation:

```
SHOW GLOBAL VARIABLES LIKE 'source_info_repository';
```

Prior to 8.0.22

```
SHOW GLOBAL VARIABLES LIKE 'master_info_repository';
```

The result should be `TABLE` instead of `FILE`.

Note: There also should not be a `source.info` or `master.info` file in the `datadir`.

Remediation:

Perform the following actions to remediate this setting:

1. Open the MySQL configuration file (`my.cnf`)
2. Locate `source_info_repository` (`master_info_repository`)
3. Set the `source_info_repository` (`master_info_repository`) value to `TABLE`

Note: If `source_info_repository` (`master_info_repository`) does not exist, add it to the configuration file.

Default Value:

```
TABLE
```

References:

1. https://dev.mysql.com/doc/refman/8.0/en/replication-options-replica.html#sysvar_source_info_repository
2. https://dev.mysql.com/doc/refman/8.0/en/replication-options-slave.html#sysvar_master_info_repository

9.4 Ensure 'super_priv' is Not Set to 'Y' for Replication Users (Automated)

Profile Applicability:

- Level 1 - MySQL RDBMS
- Level 1 - MySQL RDBMS on Linux

Description:

The `SUPER` privilege found in the `mysql.user` table governs the use of a variety of MySQL features. These features include, `CHANGE MASTER TO`, `KILL`, `mysqladmin kill` option, `PURGE BINARY LOGS`, `SET GLOBAL`, `mysqladmin debug` option, logging control, and more.

Rationale:

The `SUPER` privilege allows principals to perform many actions, including view and terminate currently executing MySQL statements (including statements used to manage passwords). This privilege also provides the ability to configure MySQL, such as enable/disable logging, alter data, disable/enable features. Limiting the accounts that have the `SUPER` privilege reduces the chances that an attacker can exploit these capabilities.

Impact:

When the `SUPER` privilege is denied to a given user, that user will be unable to take advantage of certain capabilities, such as certain `mysqladmin` options.

Audit:

Execute the following SQL statement to audit this setting:

```
select user, host from mysql.user where user='repl' and Super_priv = 'Y';
```

No rows should be returned.

If you wish to validate permissions in more detail:

```
select * from mysql.user where user='repl'\G
```

The following columns should return `Y`.

```
*************************** 1. row ***************************
             Select_priv: Y
             Reload_priv: Y
```

```
           Shutdown_priv: Y
            Process_priv: Y
               File_priv: Y
              Grant_priv: Y
            Execute_priv: Y
         Repl_slave_priv: Y
        Repl_client_priv: Y
        Create_user_priv: Y

1 row in set (0.0007 sec)
```

Check Dynamic Privileges :

```
select PRIV from mysql.global_grants where user like 'repl'\G
```

Expected results are:

```
BACKUP_ADMIN
CLONE_ADMIN
PERSIST_RO_VARIABLES_ADMIN
REPLICATION_SLAVE_ADMIN
SYSTEM_VARIABLES_ADMIN
```

Note: Substitute your replication user's name for `repl` in the above queries.

Remediation:

Execute the following steps to remediate this setting:

1. Enumerate the replication users found in the result set of the audit procedure
2. For each replication user, issue the following SQL statement (replace `repl` with your replication user's name):

```
REVOKE SUPER ON *.* FROM 'repl';
```

Note: Prior to 8.0.21 if MySQL Replica Set was used to create the replications administrator (call to `dba.configureReplicaSetInstance` in MySQL Shell) after performing the above revoke you will need to grant the following dynamic privilege.

```
GRANT REPLICATION_SLAVE_ADMIN ON *.* TO `repl WITH GRANT OPTION;
```

References:

1. https://dev.mysql.com/doc/refman/8.0/en/privileges-provided.html#priv_super
2. https://dev.mysql.com/doc/refman/8.0/en/deploying-innodb-replicasets.html

CIS Controls:

Version 7

4.7 Limit Access to Script Tools
Limit access to scripting tools (such as Microsoft PowerShell and Python) to only administrative or development users with the need to access those capabilities.

9.5 Ensure No Replication Users Have Wildcard Hostnames (Automated)

Profile Applicability:

- Level 1 - MySQL RDBMS
- Level 1 - MySQL RDBMS on Linux

Description:

MySQL can make use of host wildcards when granting permissions to users on specific databases. For example, you may grant a given privilege to `'<user>'@'%'`.

Rationale:

Avoiding the use of wildcards within hostnames helps control the specific locations from which a given user may connect to and interact with the database.

Audit:

Execute the following SQL statement to assess this recommendation:

```
SELECT user, host FROM mysql.user WHERE user='repl' AND host = '%';
```

Ensure no rows are returned.

Remediation:

Perform the following actions to remediate this setting:

1. Enumerate all users returned after running the audit procedure.
2. Either `ALTER` the user's host to be specific or `DROP` the user.

References:

1. https://dev.mysql.com/doc/refman/8.0/en/deploying-innodb-replicasets.html

Additional Information:

When creating a user for replication manually with the `CREATE USER` command or using the MySQL Replica Set command `dba.configureReplicaSetInstance` limit hosts initially.

For example:

```
mysql-js> dba.configureReplicaSetInstance('root@rs-1:3306', {clusterAdmin:
"'rsadmin'@'rs-1%'"});
```

CIS Controls:

Version 7

14.6 Protect Information through Access Control Lists
Protect all information stored on systems with file system, network share, claims, application, or database specific access control lists. These controls will enforce the principle that only authorized individuals should have access to the information based on their need to access the information as a part of their responsibilities.

10 MySQL InnoDB Cluster / Group Replication

MySQL InnoDB cluster provides a complete high availability solution for MySQL. MySQL Shell includes AdminAPI which enables you to easily configure and administer a group of at least three MySQL server instances to function as an InnoDB cluster. Each MySQL server instance runs MySQL Group Replication, which provides the mechanism to replicate data within InnoDB clusters, with built-in failover. AdminAPI removes the need to work directly with Group Replication in InnoDB clusters.

Various things can be configured to enhance the security of MySQL InnoDB Cluster.

10.1 Ensure All Group Replication Traffic is Secured (Manual)

Profile Applicability:

- Level 1 - MySQL RDBMS
- Level 1 - MySQL RDBMS on Linux

Description:

MySQL Group communication connections and distributed recovery connections can be secured using SSL.

Rationale:

SSL encryption ensures data cannot be seen over the network for Group Replication.

Audit:

Using MySQL InnoDB Cluster admin api:
Run shell

```
mysql-js> var cluster = dba.getCluster()
mysql-js> cluster.status()
{
    "clusterName": "testCluster",
    "defaultReplicaSet": {
        "name": "default",
        "primary": "localhost:3320",
        "ssl": "REQUIRED",
        "status": "OK",
```

`ssl` should **not** be `DISABLED`.

Run the following statement from mysql client:

```
select @@group_replication_ssl_mode;
```

If DISABLED communication is not secure it is a finding.

REQUIRED, VERIFY_CA, or VERIFY_IDENTITY - indicate SSL is required.

Remediation:

Edit my.cnf and set group_replication_ssl_mode, for example:

```
group_replication_ssl_mode=REQUIRED
```

Acceptable values are:

- REQUIRED - Establish a secure connection if the server supports secure connections.
- VERIFY_CA - Like REQUIRED, but additionally verify the server TLS certificate against the configured Certificate Authority (CA) certificates.
- VERIFY_IDENTITY - Like VERIFY_CA, but additionally verify that the server certificate matches the host to which the connection is being established.

References:

1. https://dev.mysql.com/doc/refman/8.0/en/group-replication-secure-socket-layer-support-ssl.html

CIS Controls:

Version 7

14.4 Encrypt All Sensitive Information in Transit
Encrypt all sensitive information in transit.

10.2 Allowlist Approved Servers Belonging to a MySQL InnoDB Cluster (Manual)

Profile Applicability:

- Level 2 - MySQL RDBMS
- Level 2 - MySQL RDBMS on Linux

Description:

Optionally, specify an allowlist of approved servers that belong to the MySQL InnoDB Cluster.

Rationale:

When using MySQL InnoDB Cluster by specifying the allowlist explicitly, you can increase the security of your cluster as only servers in the allowlist are allowed to connect to the cluster.

Audit:

Open MySQL Shell and execute the following command to crate the allowlist of servers. This list is comma separated list, surrounded by quotes. For example:

- From 8.0.22 on:

```
select @@group_replication_ip_allowlist
```

- Prior to 8.0.22:

```
select @@group_replication_ip_whitelist;
```

The result set from the above statement should be the IPv4, IPv6, or host names allowed to join the MySQL InnoDB Cluster (Group).

Remediation:

Example - to configure a cluster to only accept connections from servers at addresses `203.0.113.0/24` and `198.51.100.110`. The whitelist can also include host names, which are resolved only when a connection request is made by another server.

- From 8.0.22:

```
mysql-js> cluster.addInstance("icadmin@ic-3:3306", {ipAllowlist:
"203.0.113.0/24, 198.51.100.110"})
```

- Prior to 8.0.22:

```
mysql-js> cluster.addInstance("icadmin@ic-3:3306", {ipWhitelist:
"203.0.113.0/24, 198.51.100.110"})
```

References:

1. https://dev.mysql.com/doc/refman/8.0/en/mysql-innodb-cluster-working-with-cluster.html#mysql-innodb-cluster-securing
2. https://dev.mysql.com/doc/refman/8.0/en/group-replication-ip-address-permissions.html

Appendix: Summary Table

Control		Set Correctly	
		Yes	No
1	**Operating System Level Configuration**		
1.1	Place Databases on Non-System Partitions (Manual)	□	□
1.2	Use Dedicated Least Privileged Account for MySQL Daemon/Service (Automated)	□	□
1.3	Disable MySQL Command History (Automated)	□	□
1.4	Verify That the MYSQL_PWD Environment Variables is Not in Use (Automated)	□	□
1.5	Ensure Interactive Login is Disabled (Automated)	□	□
1.6	Verify That 'MYSQL_PWD' is Not Set in Users' Profiles (Automated)	□	□
2	**Installation and Planning**		
2.1	**Backup and Disaster Recovery**		
2.1.1	Backup Policy in Place (Manual)	□	□
2.1.2	Verify Backups are Good (Manual)	□	□
2.1.3	Secure Backup Credentials (Manual)	□	□
2.1.4	The Backups Should be Properly Secured (Manual)	□	□
2.1.5	Point-in-Time Recovery (Automated)	□	□
2.1.6	Disaster Recovery (DR) Plan (Manual)	□	□
2.1.7	Backup of Configuration and Related Files (Manual)	□	□
2.2	**Data Encryption**		
2.2.1	Ensure Binary and Relay Logs are Encrypted (Automated)	□	□
2.3	Dedicate the Machine Running MySQL (Manual)	□	□
2.4	Do Not Specify Passwords in the Command Line (Manual)	□	□
2.5	Do Not Reuse Usernames (Manual)	□	□
2.6	Ensure Non-Default, Unique Cryptographic Material is in Use (Manual)	□	□
2.7	Ensure 'password_lifetime' is Less Than or Equal to '365' (Automated)	□	□
2.8	Ensure Password Complexity is Configured (Automated)	□	□
2.9	Ensure Password Resets Require Strong Passwords (Automated)	□	□
2.10	Require Current Password for Password Reset (Automated)	□	□
2.11	Use Dual Passwords to Enable Higher Frequency Password Rotation (Manual)	□	□
2.12	Lock Out Accounts if Not Currently in Use (Manual)	□	□

2.13	Ensure AES Encryption Mode for AES_ENCRYPT/AES_DECRYPT is Configured Correctly (Automated)	□	□
2.14	Ensure Socket Peer-Credential Authentication is Used Appropriately (Manual)	□	□
2.15	Ensure MySQL is Bound to an IP Address (Automated)	□	□
2.16	Limit Accepted Transport Layer Security (TLS) Versions (Automated)	□	□
2.17	Require Client-Side Certificates (X.509) (Automated)	□	□
2.18	Ensure Only Approved Ciphers are Used (Automated)	□	□
2.19	Implement Connection Delays to Limit Failed Login Attempts (Automated)	□	□
3	**File Permissions**		
3.1	Ensure 'datadir' Has Appropriate Permissions (Automated)	□	□
3.2	Ensure 'log_bin_basename' Files Have Appropriate Permissions (Automated)	□	□
3.3	Ensure 'log_error' Has Appropriate Permissions (Automated)	□	□
3.4	Ensure 'slow_query_log' Has Appropriate Permissions (Automated)	□	□
3.5	Ensure 'relay_log_basename' Files Have Appropriate Permissions (Automated)	□	□
3.6	Ensure 'general_log_file' Has Appropriate Permissions (Automated)	□	□
3.7	Ensure SSL Key Files Have Appropriate Permissions (Automated)	□	□
3.8	Ensure Plugin Directory Has Appropriate Permissions (Automated)	□	□
3.9	Ensure 'audit_log_file' Has Appropriate Permissions (Automated)	□	□
3.10	Secure MySQL Keyring (Automated)	□	□
4	**General**		
4.1	Ensure the Latest Security Patches are Applied (Manual)	□	□
4.2	Ensure Example or Test Databases are Not Installed on Production Servers (Automated)	□	□
4.3	Ensure 'allow-suspicious-udfs' is Set to 'OFF' (Automated)	□	□
4.4	Harden Usage for 'local_infile' on MySQL Clients (Automated)	□	□
4.5	Ensure 'mysqld' is Not Started With '--skip-grant-tables' (Automated)	□	□
4.6	Ensure Symbolic Links are Disabled (Automated)	□	□
4.7	Ensure the 'daemon_memcached' Plugin is Disabled (Automated)	□	□

4.8	Ensure the 'secure_file_priv' is Configured Correctly (Automated)	□	□
4.9	Ensure 'sql_mode' Contains 'STRICT_ALL_TABLES' (Automated)	□	□
4.10	Use MySQL TDE for At-Rest Data Encryption (Automated)	□	□
5	**MySQL Permissions**		
5.1	Ensure Only Administrative Users Have Full Database Access (Manual)	□	□
5.2	Ensure 'file_priv' is Not Set to 'Y' for Non-Administrative Users (Manual)	□	□
5.3	Ensure 'process_priv' is Not Set to 'Y' for Non-Administrative Users (Manual)	□	□
5.4	Ensure 'SUPER' is Not Granted to Non-Administrative Users (Manual)	□	□
5.5	Ensure 'shutdown_priv' is Not Set to 'Y' for Non-Administrative Users (Manual)	□	□
5.6	Ensure 'create_user_priv' is Not Set to 'Y' for Non-Administrative Users (Manual)	□	□
5.7	Ensure 'grant_priv' is Not Set to 'Y' for Non-Administrative Users (Manual)	□	□
5.8	Ensure 'repl_slave_priv' is Not Set to 'Y' for Non-Replica Users (Manual)	□	□
5.9	Ensure DML/DDL Grants are Limited to Specific Databases and Users (Manual)	□	□
5.10	Securely Define Stored Procedures and Functions DEFINER and INVOKER (Manual)	□	□
6	**Auditing and Logging**		
6.1	Ensure 'log_error' is Not Empty (Automated)	□	□
6.2	Ensure Log Files are Stored on a Non-System Partition (Automated)	□	□
6.3	Ensure 'log_error_verbosity' is Set to '2' (Automated)	□	□
6.4	Ensure 'log-raw' is Set to 'OFF' (Automated)	□	□
6.5	Ensure Audit Filters Capture Connection Attempts (Manual)	□	□
6.6	Ensure ALL Events are Audited (Automated)	□	□
6.7	Set audit_log_strategy to SYNCHRONOUS or SEMISYNCRONOUS (Automated)	□	□
6.8	Ensure the Audit Plugin Can't be Unloaded (Automated)	□	□
7	**Authentication**		
7.1	Ensure default_authentication_plugin is Set to a Secure Option (Automated)	□	□
7.2	Ensure Passwords are Not Stored in the Global Configuration (Automated)	□	□
7.3	Ensure Passwords are Set for All MySQL Accounts (Automated)	□	□

7.4	Set 'default_password_lifetime' to Require a Yearly Password Change (Automated)	□	□
7.5	Ensure Password Complexity Policies are in Place (Automated)	□	□
7.6	Ensure No Users Have Wildcard Hostnames (Automated)	□	□
7.7	Ensure No Anonymous Accounts Exist (Automated)	□	□
8	**Network**		
8.1	Ensure 'require_secure_transport' is Set to 'ON' and/or 'have_ssl' is Set to 'YES' (Automated)	□	□
8.2	Ensure 'ssl_type' is Set to 'ANY', 'X509', or 'SPECIFIED' for All Remote Users (Automated)	□	□
8.3	Set Maximum Connection Limits for Server and per User (Manual)	□	□
9	**Replication**		
9.1	Ensure Replication Traffic is Secured (Manual)	□	□
9.2	Ensure 'SOURCE_SSL_VERIFY_SERVER_CERT' is Set to 'YES' or '1' (Automated)	□	□
9.3	Ensure 'source_info_repository' is Set to 'TABLE' (Automated)	□	□
9.4	Ensure 'super_priv' is Not Set to 'Y' for Replication Users (Automated)	□	□
9.5	Ensure No Replication Users Have Wildcard Hostnames (Automated)	□	□
10	**MySQL InnoDB Cluster / Group Replication**		
10.1	Ensure All Group Replication Traffic is Secured (Manual)	□	□
10.2	Allowlist Approved Servers Belonging to a MySQL InnoDB Cluster (Manual)	□	□

Appendix: Change History

Date	Version	Changes for this version
Apr 14, 2021	1.0.0	Least Priv MySQL Account should not have shell access (Ticket 10283)
Apr 14, 2021	1.0.0	Fix links (Ticket 10284)
Apr 14, 2021	1.0.0	Typo in Description (Ticket 9912)
Apr 14, 2021	1.0.0	Why don't you use 'validate_password_check_user_name'? (Ticket 9896)
Apr 14, 2021	1.0.0	Set to null if import/export not used. (Ticket 10319)
Apr 14, 2021	1.0.0	Add detailed permissions guide to reference (Ticket 10311)
Apr 14, 2021	1.0.0	permissions intended should be more restrictive (Ticket 10315)
Apr 14, 2021	1.0.0	Add Section - Securely define Store Procedures and Functions DEFINER and SQL SECURITY clauses (Ticket 10385)
Apr 14, 2021	1.0.0	Binlog Strategy additional guidance (Ticket 10289)
Apr 14, 2021	1.0.0	Enhance description - refer to new versus old technology and product names (Ticket 10291)
Apr 14, 2021	1.0.0	Add mysql-auto.cnf (.. where SET PERSIST values go) (Ticket 10292)
Apr 14, 2021	1.0.0	Add MySQL Shell (Ticket 10285)

Apr 14, 2021	1.0.0	Add Section - Require current password for password replacement (Ticket 10357)
Apr 14, 2021	1.0.0	Add section - Use dual password to increase frequency of application password rotation. (Ticket 10358)
Apr 14, 2021	1.0.0	If connection delays are enable monitor (Ticket 10362)
Apr 14, 2021	1.0.0	Add section - limit host IP using bind (Ticket 10363)
Apr 14, 2021	1.0.0	Require specific TLS versions (Ticket 10364)
Apr 14, 2021	1.0.0	Add Section on - Enabling TDE encryption (Ticket 10365)
Apr 14, 2021	1.0.0	Add to dual password audit the following SQL (Ticket 10373)
Apr 14, 2021	1.0.0	Add section - Set Password Reuse Policy (Ticket 10356)
Apr 14, 2021	1.0.0	Add section - Lock accounts if not in current use. (Ticket 10359)
Apr 14, 2021	1.0.0	Add section - Implement Connection Delays to defend against brute authentication attacks (Ticket 10361)
Apr 14, 2021	1.0.0	Create Section - Timeout authentication for interactive and non-interactive sessions (Ticket 10383)
Apr 14, 2021	1.0.0	Add Section - Set Account Resource Limits (Ticket 10384)
Apr 14, 2021	1.0.0	Add section - Ensure MySQL Cluster requires SSL Mode (Ticket 10386)

Apr 14, 2021	1.0.0	Add Section - Optionally specify a whitelist of approved servers that belong to the MySQL InnoDB Cluster (Ticket 10393)
Apr 14, 2021	1.0.0	Change title and update content (Ticket 10323)
Apr 14, 2021	1.0.0	Should mention reserved users (new to 8.0) (Ticket 10294)
Apr 14, 2021	1.0.0	Shouldn't this be level 1? (Ticket 10295)
Apr 14, 2021	1.0.0	Not sufficient to just check location of the data directory (Ticket 10282)
Apr 14, 2021	1.0.0	Update References (Ticket 10286)
Apr 14, 2021	1.0.0	Update Env Ref (Ticket 10287)
Apr 14, 2021	1.0.0	check require_secure_transport (Ticket 10327)
Apr 14, 2021	1.0.0	Need to rewrite this entirely. Old and dated. (Ticket 10312)
Apr 14, 2021	1.0.0	general_log_file should be off. (Ticket 10313)
Apr 14, 2021	1.0.0	Should show all files for ssl (Ticket 10314)
Apr 14, 2021	1.0.0	Add encryption to this (Ticket 10316)
Apr 14, 2021	1.0.0	keyring_file_data is not a best practice (Ticket 10317)
Apr 14, 2021	1.0.0	Remove section - there is no test database in 5.7 nor 8 (Ticket 10318)
Apr 14, 2021	1.0.0	In 8 super replaced by dynamic privy (Ticket 10321)

Apr 14, 2021	1.0.0	No longer in 8.0 - remove section Ensure 'secure_auth' is set to 'ON' (Ticket 10324)
Apr 14, 2021	1.0.0	need to rewrite 6.5-6.7 - this is legacy modes - too course grained etc. (Ticket 10322)
Apr 14, 2021	1.0.0	Remove section 'sql_mode' contains ... not in 8 (Ticket 10325)
Apr 14, 2021	1.0.0	Consider a more comprehsive SQL statement to audit admin privy (Ticket 10320)
Apr 14, 2021	1.0.0	Should Group Replication be added to this section? (Ticket 10328)
Apr 14, 2021	1.0.0	Better to set global policy than per user. (Ticket 10296)
Apr 14, 2021	1.0.0	MySQL Shell needs to be added (Ticket 10293)
Apr 14, 2021	1.0.0	MySQL Enterprise Backup (Ticket 10288)
Apr 14, 2021	1.0.0	First reference link is not existing (Ticket 12331)
Apr 14, 2021	1.0.0	MySQL 8.0 Benchmark Final Version (Ticket 10405)