ORACLE

Secure Your Data Workshop

MySQL Enterprise Security

Marco Carlessi
MySQL Principal Solutions Engineer
marco.carlessi@oracle.com
November 2024

Agenda

Achieve Compliance with MySQL Enterprise Security Features

- Workshop Overview
- Install sample application
- Setup and Installation of:
  - Enterprise Audit
  - Enterprise Transparent Data Encryption
  - Enterprise Data Masking
  - Enterprise Firewall

Workshop Overview

- **Goals:**
  1. Install MySQL Enterprise Edition
  2. Overview & Setup Enterprise Audit, Enterprise Transparent Data Encryption, Enterprise Data Masking and Enterprise Firewall.

- **Not intended for MySQL Training Class**

- **Lab:**
  https://mc-carlessi.github.io/mysql-enterprise-security/workshops/student/index.html

Why?

Data: Your Most Valuable Asset

Data Breaches – keep increasing…

Number of breaches in December 2023: 1,351

Unprotected Real Estate Wealth Network had more than 1.5 billion records stolen

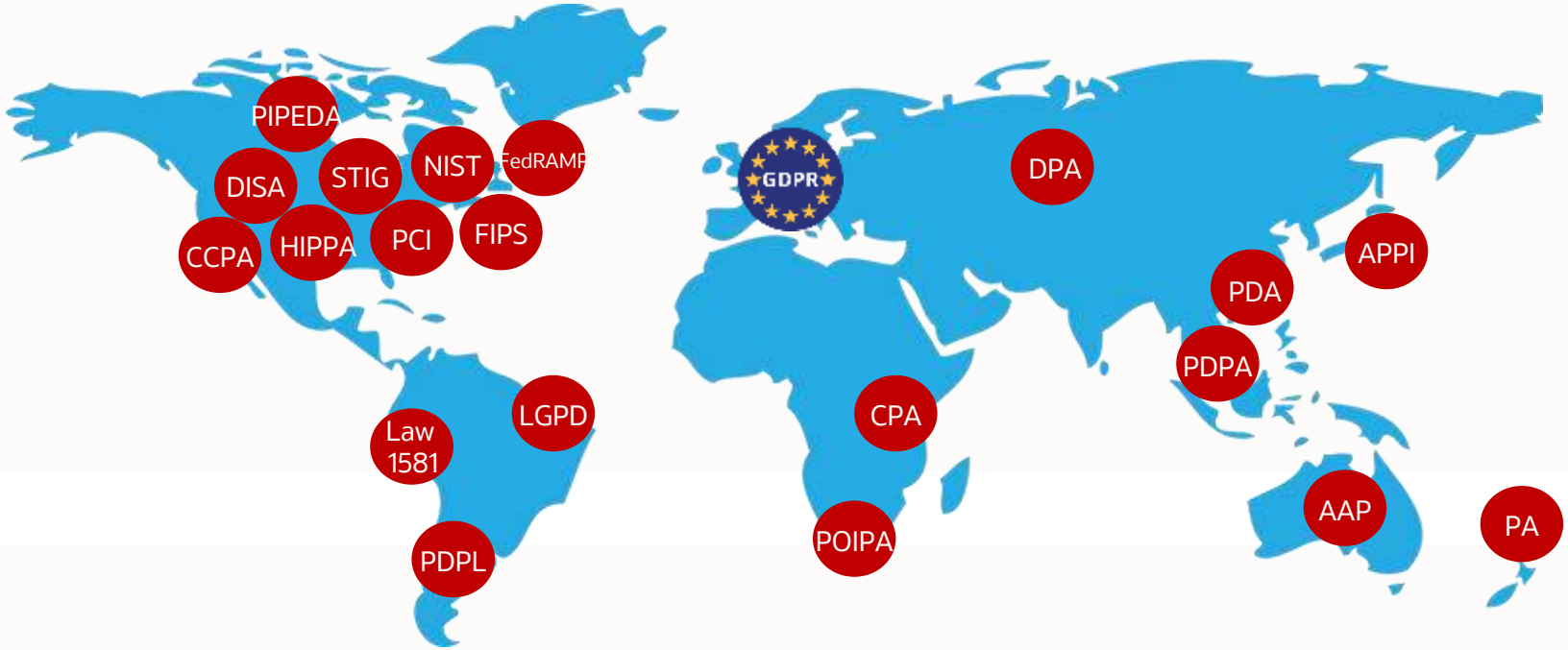
Number of breached records in December 2023: 2,241,916,765

Popular parental control app Kid Security had more than 300 million records exposed

https://www.itgovernance.co.uk/blog/list-of-data-breaches-and-cyber-attacks-in-2023

What?

Data Protection Laws & Regulations



Regulatory Compliance

- **Regulations**
  - PCI – DSS: Payment Card Data
  - HIPAA: Privacy of Health Data
  - Sarbanes Oxley, GLBA, The USA Patriot Act:
    - Financial Data, NPI "personally identifiable financial information"
  - FERPA – Student Data
  - EU General Data Protection Directive: Protection of Personal Data (GDPR)
  - Data Protection Act (UK): Protection of Personal Data
- **Requirements**
  - Continuous Monitoring (Users, Schema, Backups, etc)
  - Data Protection (**Encryption**, Privilege Management, etc.)
  - Data Retention (Backups, User Activity, etc.)
  - Data **Auditing** (User activity, etc.)

How?

MySQL Database Hardening Best Practices

Installation

- Keep MySQL up to date
- MySQL Installer for Windows
- Yum/Apt/other Repositories

Authentication

- Password Policies
- Multi factor authentication
- External Authentication
- X.509

Authorization

- Remove Extra Accounts
- Grant Minimal Privileges
- Audit users and privileges

Configuration

- Firewall
- Auditing and Logging
- Limit Network Access
- Monitor changes

Encryption

- SSL/TLS for Secure Connections
- Data Encryption (AES, RSA)
- TDE
- Masking and De-Identification

Collateral

- Encrypt Backups
- Secure Replications
- Secure HA and DR

Data Protection & Regulatory Compliance

MySQL Enterprise Edition

Manage Privileged Users	Protect Dev & Test Data	Encrypt Your Data	Detect Database Activity
MySQL Enterprise Authentication	MySQL Enterprise Masking MySQL Enterprise Backup	MySQL Enterprise TDE MySQL Enterprise Encryption	MySQL Enterprise Audit MySQL Enterprise Firewall MySQL Plugin for OEM13c

Database and OS Installation, Configuration & Maintenance

- Build on a secure platform
  - OS patched, firewall, etc.
- Use the Secure build
  - https://dev.mysql.com/doc/mysql-secure-deployment-guide/8.0/en/
  - https://dev.mysql.com/doc/mysql-secure-deployment-guide/5.7/en/
- Keep Operating System and MySQL security patches up to date
  - May require a restart (MySQL or operating system) to take effect
- Follow OS vendor specific hardening guidelines
  - For example Tips for Hardening an Oracle Linux Server: http://www.oracle.com/technetwork/articles/servers-storage-admin/tips-harden-oracle-linux-1695888.html
- CIS Benchmark guides

MySQL User Accounts

- Within MySQL, accounts are composed by three elements:
  - Username
    - blank usernames are considered anonymous
  - Host, the reference of the machine where the mysql client run
    - Used transparently by the client/connector
    - May use May use IP or FQDNs or domains with wildcards (e.g. '%' means every host)
  - Password
    - MySQL 8 default plugin is caching_sha2_password
    - MySQL 5.X format is mysql_native_password (it uses the SHA1 algorithm which NIST no longer recommend)
    - It's possible to have users with different password plugin in the same instance
    - Default plugin can be specified with the variable default_authentication_plugin
- Examples

```
CREATE USER 'username1'@'%' IDENTIFIED BY 'VeryComplex1!';
CREATE USER 'username2'@'myhost.oracle.com' IDENTIFIED BY 'VeryComplex2!';
CREATE USER 'username3'@'192.168.1.1' IDENTIFIED BY 'VeryComplex3!';
CREATE USER ''@'%' IDENTIFIED BY 'VeryComplex4!';
CREATE USER 'username4'@localhost IDENTIFIED WITH mysql_native_password BY 'VeryComplex4!';
```

https://dev.mysql.com/doc/refman/8.0/en/user-account-management.html

Table Example - `mysql.user`

- User table contains one row for each account
- Use DESC to check the attributes

```
mysql> SELECT user, host, plugin, user_attributes FROM mysql.user;
+---------+-------------+-----------------------+-----------------------------------------+
| user    | host        | plugin                | User_attributes                         |
+---------+-------------+-----------------------+-----------------------------------------+
| root    | 127.0.0.1   | caching_sha2_password | NULL                                    |
+---------+-------------+-----------------------+-----------------------------------------+
| root    | %           | caching_sha2_password | NULL                                    |
+---------+-------------+-----------------------+-----------------------------------------+
| user1   | 192.168.1.2 | caching_sha2_password | NULL                                    |
+---------+-------------+-----------------------+-----------------------------------------+
| user2   | 192.168.1.3 | caching_sha2_password | {"additional_password": "$A$005$H[}62…"}  |
+---------+-------------+-----------------------+-----------------------------------------+
| user3   | 192.168.1.4 | mysql_native_password | {"metadata": {"comment": "old app user"}} |
+---------+-------------+-----------------------+-----------------------------------------+
```

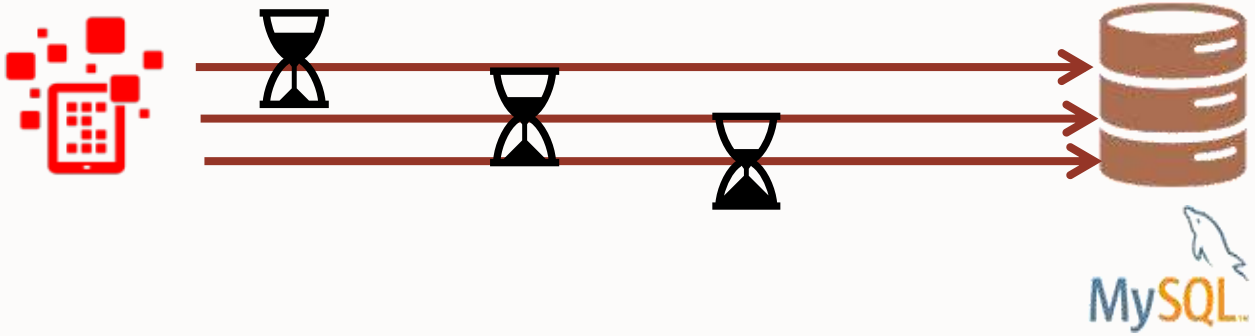

Password Validation Component

- Check robustness of password and prevent usage of weak ones
- It use preconfigured or customized policies (`validate_password_policy` variable)
  - Low: Length
  - Medium: Length; numeric, lowercase/uppercase, and special characters
  - Strong: Length; numeric, lowercase/uppercase, and special characters; dictionary file
- Variables:
  - validate_password_length
  - validate_password_mixed_case_count, validate_password_number_count, validate_password_special_char_count
  - validate_password_dictionary_file
  - validate_password.changed_characters_percentage
  - validate_password.check_user_name
- example

```
mysql> SET PASSWORD = 'abc';
  ERROR 1819 (HY000): Your password does not satisfy the current policy
```


Connection Control Plugin

- Protect from brute force attacks
  - Introduce an increasing delay in server response to clients after a certain number of consecutive failed connection attempts
- CONNECTION_CONTROL plugin
  - checks incoming connections and adds a delay to server responses as necessary
- CONNECTION_CONTROL_FAILED_LOGIN_ATTEMPTS plugin
  - implements an INFORMATION_SCHEMA table that exposes more detailed monitoring information for failed connection attempts



Additional Password Management

- Password expiration
  - To require passwords to be changed periodically
- Password reuse restrictions
  - To prevent old passwords from being chosen again
- Password verification
  - To require that password changes also specify the current password to be replaced
- Dual passwords
  - To enable clients to connect using either a primary or secondary password
- Password strength assessment
  - To require strong passwords. Implemented using the password validation component
- Random password generation
  - An alternative to requiring explicit administrator-specified literal passwords
- Password failure tracking
  - To enable temporary account locking after too many consecutive incorrect-password login failures

mysql_config_editor

- The mysql_config_editor utility enables you to store authentication credentials in an obfuscated login path file named .mylogin.cnf
  - Useful for scripts and automation
  - Enabled by all the mysql utilities and clients
- mylogin.cnf login path file consists of option groups, each one with host, user, password, port and socket

```
[prod1]
user = mydefaultname
password = mydefaultpass
host = 127.0.0.1
…
```

- Integrated with all the mysql utilities and clients
- Usage example

```
mysql_config_editor set --login-path=prod1 --host=127.0.0.1 --user=mydefaultname -password
mysql --login-path=prod1
```

https://dev.mysql.com/doc/refman/8.0/en/mysql-config-editor.html
https://dev.mysql.com/doc/mysql-shell/8.0/en/mysql-shell-pluggable-password-store.html

MySQL Enterprise Authentication

- Simplify your User management, reuse IT passwords
- Multi Factor authentication
- Integrate with Centralized Authentication Infrastructure
  - Centralized Account Management
  - Password Policy Management
  - Groups & Roles
- Pluggable Authentication Modules (PAM)
  - Linux PAM Standard interface (Unix, LDAP, Kerberos, others)
- Kerberos
- Webauthn and FIDO interface (password less authentication)
- May enable Single Sign On (SSO)
  - Plug-in available to access native LDAP service for authentication
  - Plug-in available to access native Windows service

Integrates MySQL with existing security infrastructures

MySQL Privileges

- Grant X privileges on Y to Z [with grant options]

```
GRANT ALL PRIVILEGES ON *.* TO 'test'@'%' WITH GRANT OPTION;
```

- Fine grained controls over user privileges
  - Administrative Privileges: Global, enable users to manage the server
  - Creating, altering and deleting databases tables, columns, indexes
  - Create, execute, or delete stored procedures and with what rights
  - Execute INSERT, SELECT, UPDATE, DELETE queries
- Deny access by username is not possible
  - but you can disable a user
- Control can be disabled with --skip-grant-tables
  - Evaluate skip_networking to use in combination
  - https://dev.mysql.com/doc/refman/8.0/en/resetting-permissions.html
- MySQL roles (collection of privileges) can simplify privileges assignments
  - Assigned and revoked with usual GRANT statements
  - With dedicated functions

Privilege	Meaning and Grantable Levels
ALL [PRIVILEGES]	Grant all privileges at specified access level except GRANT OPTION and PROXY.
ALTER	Enable use of ALTER TABLE. Levels: Global, database, table.
ALTER ROUTINE	Enable stored routines to be altered or dropped. Levels: Global, database, procedur
CREATE	Enable database and table creation. Levels: Global, database, table.
CREATE ROUTINE	Enable stored routine creation. Levels: Global, database.
CREATE TABLESPACE	Enable tablespaces and log file groups to be created, altered, or dropped. Level: Glo
CREATE TEMPORARY TABLES	Enable use of CREATE TEMPORARY TABLE. Levels: Global, database.
CREATE USER	Enable use of CREATE USER, DROP USER, RENAME USER, and REVOKE ALL PRIV
CREATE VIEW	Enable views to be created or altered. Levels: Global, database, table.
DELETE	Enable use of DELETE. Level: Global, database, table.
DROP	Enable databases, tables, and views to be dropped. Levels: Global, database, table.
EVENT	Enable use of events for the Event Scheduler. Levels: Global, database.
EXECUTE	Enable the user to execute stored routines. Levels: Global, database, table.
FILE	Enable the user to cause the server to read or write files. Level: Global.
GRANT OPTION	Enable privileges to be granted to or removed from other accounts. Levels: Global, d
INDEX	Enable indexes to be created or dropped. Levels: Global, database, table.
INSERT	Enable use of INSERT. Levels: Global, database, table, column.
LOCK TABLES	Enable use of LOCK TABLES on tables for which you have the SELECT privilege. Le
PROCESS	Enable the user to see all processes with SHOW PROCESSLIST. Level: Global.
PROXY	Enable user proxying. Level: From user to user.
REFERENCES	Enable foreign key creation. Levels: Global, database, table, column.
RELOAD	Enable use of FLUSH operations. Level: Global.
REPLICATION CLIENT	Enable the user to ask where master or slave servers are. Level: Global.
REPLICATION SLAVE	Enable replication slaves to read binary log events from the master. Level: Global.
SELECT	Enable use of SELECT. Levels: Global, database, table, column.
SHOW DATABASES	Enable SHOW DATABASES to show all databases. Level: Global.
SHOW VIEW	Enable use of SHOW CREATE VIEW. Levels: Global, database, table.
SHUTDOWN	Enable use of mysqladmin shutdown. Level: Global.
SUPER	Enable use of other administrative operations such as CHANGE MASTER TO, KILL,
TRIGGER	Enable trigger operations. Levels: Global, database, table.
UPDATE	Enable use of UPDATE. Levels: Global, database, table, column.
USAGE	Synonym for "no privileges"

MySQL Enterprise Audit

MySQL Enterprise Audit

- Out-of-the-box logging of connections, logins, and queries.
- Simple to fine grained policies for filtering and log rotation.
- Dynamically enabled & disabled.
- Various options for the Audit Logs:
  - XML-based audit stream
  - New 5.7.21+
    - JSON
    - Compression
    - Encryption
    - Remote Read Only SQL statement access
- Send data to a remote server / audit data vault
  - Oracle Audit Vault, Splunk, etc.

Adds regulatory compliance to MySQL applications (HIPAA, Sarbanes-Oxley, GDPR, etc.)

MySQL Enterprise Audit - Work Flow

Audit Log File Formats

Log File Format

XML - audit_log_format=NEW

```
<?xml version="1.0" encoding="utf-8"?>
 <AUDIT>
 <AUDIT_RECORD>
   <TIMESTAMP>2019-10-03T14:06:33 UTC</TIMESTAMP>
    <RECORD_ID>1_2019-10-03T14:06:33</RECORD_ID>
    <NAME>Audit</NAME>
    <SERVER_ID>1</SERVER_ID>
    <VERSION>1</VERSION>
    <STARTUP_OPTIONS>/usr/local/mysql/bin/mysqld --
socket=/usr/local/mysql/mysql.sock --port=3306</STARTUP_OPTIONS>
     <OS_VERSION>i686-Linux</OS_VERSION>
     <MYSQL_VERSION>5.7.21-log</MYSQL_VERSION>
</AUDIT_RECORD>
```

JSON – audit_log_format=JSON

```
{ "timestamp": "2019-10-03 14:21:56",
   "id": 0,
    "class": "audit",
    "event": "startup",
    "connection_id": 0,
    "startup_data": { "server_id": 1,
                          "os_version": "i686-Linux",
                      "mysql_version": "5.7.21-log",
                      "args": ["/usr/local/mysql/bin/mysqld",
                                "--loose-audit-log-format=JSON",
                                "--log-error=log.err",
                                "--pid-file=mysqld.pid",
                                "--port=3306" ] } }
```

Complete Audit Data

Complete event details

- Who
- What
- When
- Where
- How
- Status

- DB version
- OS version
- Options
- and more…

```
<?xml version="1.0" encoding="UTF-8"?>
<AUDIT>
  <AUDIT_RECORD
    TIMESTAMP="2012-08-02T14:52:12"
    NAME="Audit"
    SERVER_ID="1"
    VERSION="1"
    STARTUP_OPTIONS="--port=3306"
    OS_VERSION="i686-Linux"
    MYSQL_VERSION="5.5.28-debug-log"/>
  <AUDIT_RECORD
    TIMESTAMP="2012-08-02T14:52:41"
    NAME="Connect"
    CONNECTION_ID="1"
    STATUS="0"
    USER="joe"
    PRIV_USER="root"
    OS_LOGIN=""
    PROXY_USER=""
    HOST="SERVER1"
    IP="127.0.0.1"
    DB="joes_db"/>
  <AUDIT_RECORD
    TIMESTAMP="2012-08-02T14:53:45"
    NAME="Query"
    CONNECTION_ID="1"
    STATUS="0"
    SQLTEXT="SELECT * FROM joes_table;"/>
</AUDIT>
```

Audit Log Filters

Expanded “Event” model

- Allows for very fine grained auditing

Simple but powerful

- Uses JSON to define filters

Event class	Event subclass
GENERAL	STATUS
CONNECTION	CONNECT
	CHANGE_USER
	DISCONNECT
TABLE_ACCESS	READ
	INSERT
	UPDATE
	DELETE
MESSAGE	INTERNAL
	USER

Filters example 1

```
mysql> SET @f = '{ "filter": { "class": { "name": "connection" } } }';
Query OK, 0 rows affected (0.00 sec)

mysql> SELECT audit_log_filter_set_filter('log_conn_events', @f);
+-----------------------------------------------------+
| audit_log_filter_set_filter('log_conn_events', @f) |
+-----------------------------------------------------+
| OK                                                  |
+-----------------------------------------------------+
1 row in set (0.01 sec)

mysql> SELECT * FROM mysql.audit_log_filter;
+------------------+-------------------------------------------------+
| NAME             | FILTER                                          |
+------------------+-------------------------------------------------+
| log_conn_events | {"filter": {"class": {"name": "connection"}}} |
+------------------+-------------------------------------------------+
1 row in set (0.00 sec)
```

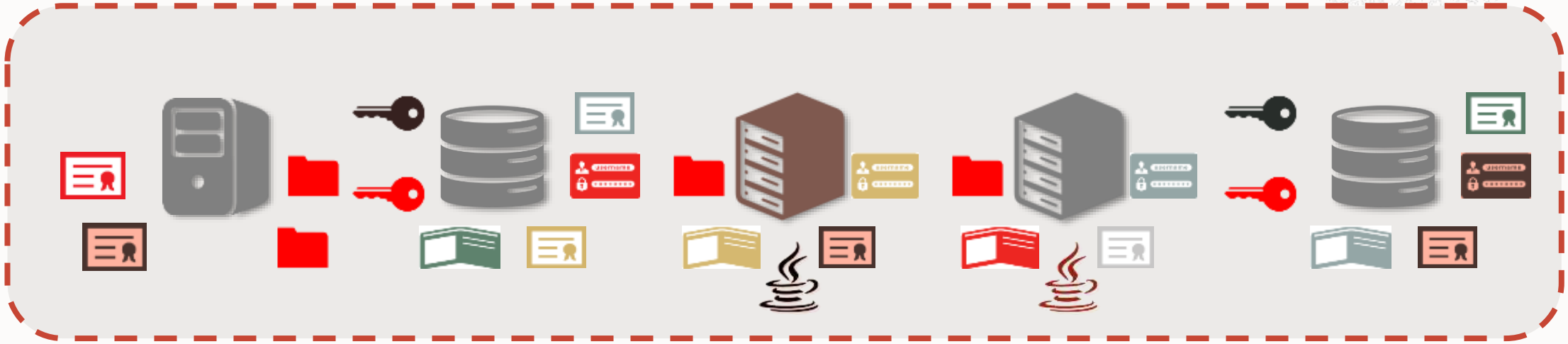
Filters example 2

All deletions, insertions, updates on bank_database.accounts

```
{ "filter":{
   "class": {
    "name": "table_access",
    "event": {
     "name": [ "delete", "insert", "update" ],
     "log": {
       "and": [ { "field": { "name": "table_database.str",
                             "value": "bank_database" } },
                { "field": { "name": "table_name.str",
                             "value": "accounts" }}]}}}}}
```

MySQL Enterprise Transparent Data Encryption

The Challenges of Key Management



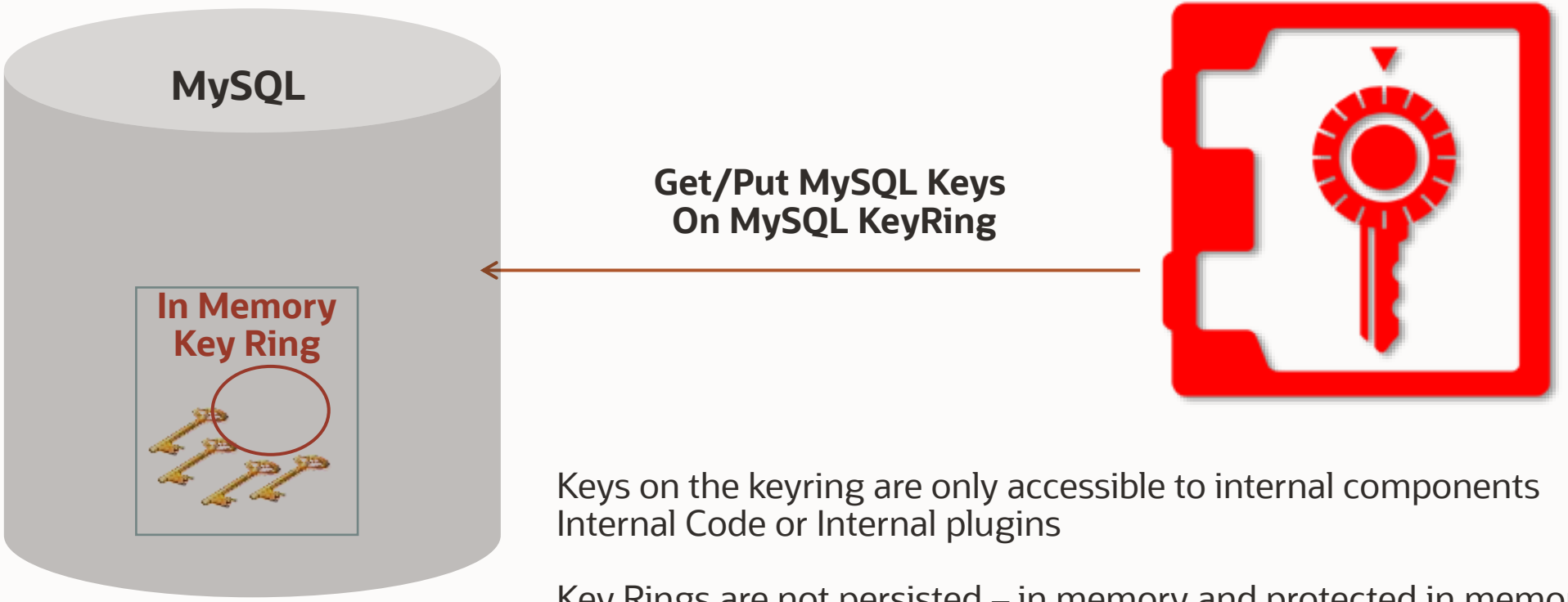
Management

- Proliferation of encryption wallets and keys
- Authorized sharing of keys
- Key availability, retention, and recovery
- Custody of keys and key storage files

Regulations

- Physical separation of keys from encrypted data
- Periodic key rotations
- Monitoring and auditing of keys
- Long-term retention of keys and encrypted data

MySQL Key Ring



Keys on the keyring are only accessible to internal components
Internal Code or Internal plugins

Key Rings are not persisted – in memory and protected in memory

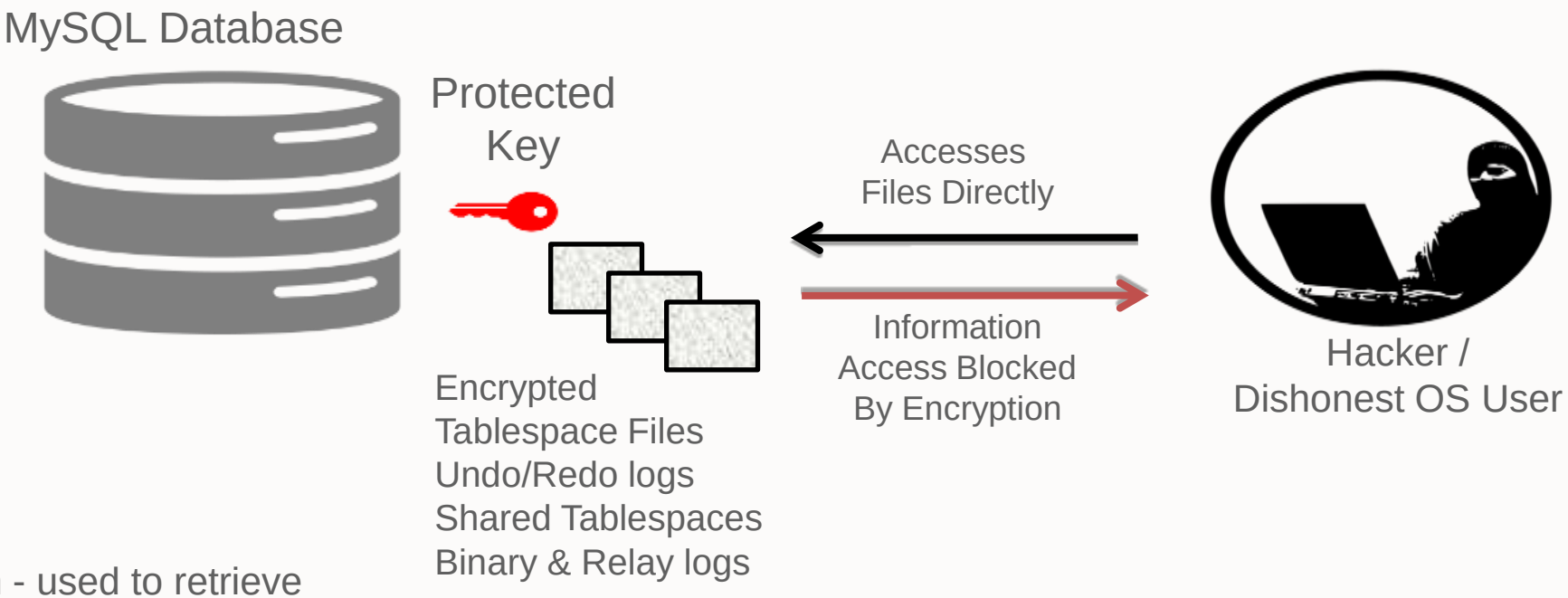
ACLs - who key is for – for example InnoDB Tablespaces

MySQL Enterprise Security Transparent Data Encryption

- Data at Rest Encryption
  - [System | General | Data Dictionary] Tablespaces, Undo/Redo & Binary/Relay logs, Storage, OS File system
  - Policy to enforce table encryption
  - Strong Encryption – AES 256
- Transparent to applications and users
  - No application code, schema or data type changes
- Transparent to DBAs
  - Keys are hidden from DBAs, no configuration changes
- Requires Key Management
  - Protection, rotation, storage, recovery

MySQL Enterprise Transparent Data Encryption (TDE)

Protects against Attacks on Database Files



Keyring plugin - used to retrieve keys from Key Stores over Standardized **KMIP** protocol

MySQL Enterprise Security Transparent Data Encryption

At Rest Encryption Covers

- InnoDB Tables and Tablespace
  - File Per Table Tablespace or General (Multi-Table) Tablespace
- MySQL System Tablespace
  - Data Dictionary Tables
- Binlog Encryption
- MySQL Enterprise Audit Logs
- MySQL Enterprise Backup Files
- Note: DBAs can optionally force Table Encryption
  - i.e. Users can only create encrypted tables

MySQL Enterprise Data Masking

MySQL Enterprise Edition: Masking and De-Identification

De-identify, Anonymize Sensitive Data

Data De-Identification helps database customers improve security

Accelerates compliance for

- Government – GDPR, CHHS
- Financial - PCI
- Healthcare – HIPAA, Clinic Trials Data

Employee Table

ID	Last	First	SSN
1111	Smith	John	555-12-5555
1112	Templeton	Richard	444-12-4444

Random Data Generation **Masked View**

ID	Last	First	SSN
2874	Smith	John	XXX-XX-5555
3281	Templeton	Richard	XXX-XX-4444

MySQL Enterprise Edition: Masking and De-Identification

Data Masking

String data masking

- Mask a substring within a string : ArthXXXXnt
- Mask substrings at the beginning and at the end : XXthurDeXX

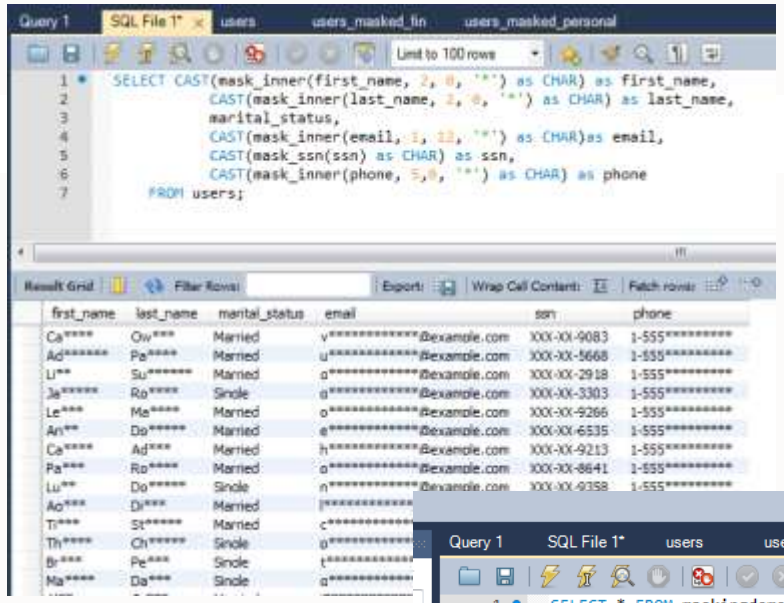
SSN masking : XXXX-XX-1234

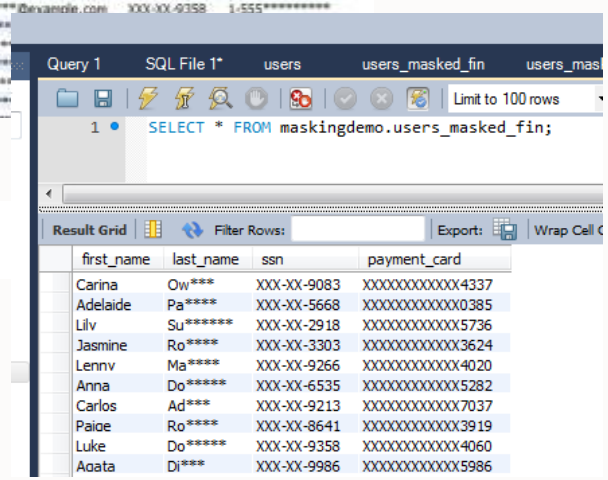
Payment Card masking

- Strict: XXXXXXXXXXXXXXX7395
- Relaxed: 493812XXXXXXXXX7395

Dictionary based masking

- gen_blacklist(“007”, “00designations”, “Cover_identity”) => Universal Exports





MySQL Enterprise Firewall

MySQL Enterprise Firewall

Real-time Database Intrusion Detection

MySQL Enterprise Firewall monitoring

Real Time Protection

- Queries analyzed and matched against Allow List

Blocks SQL Injection Attacks

- Positive Security Model

Block Suspicious Traffic

- Out of Policy Transactions detected & blocked

Learns Allow List

- Automated creation of approved list of SQL command patterns on a per user basis

Transparent

- No changes to application required

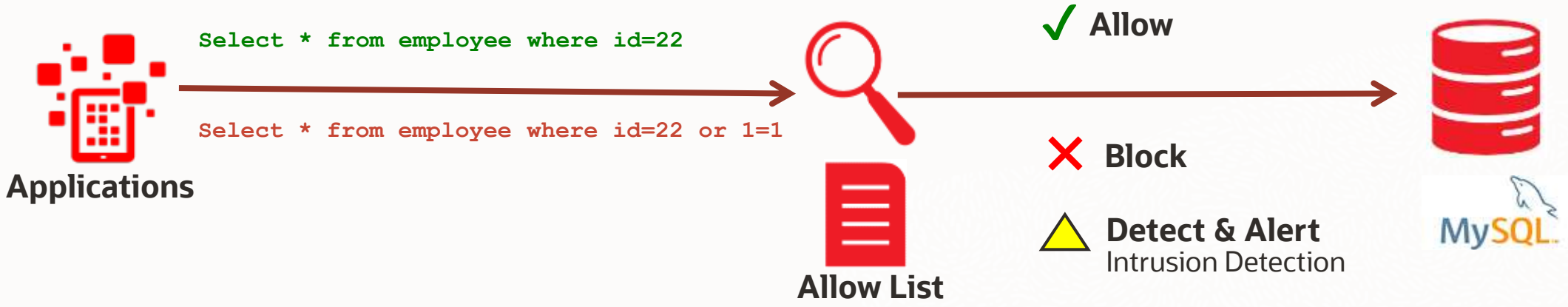
MySQL Enterprise Firewall

Block SQL Injection Attacks

- Allow: SQL Statements that match Allowlist
- Block: SQL statements that are not on Allowlist

Intrusion Detection System

- Detect: SQL statements that are not on Allowlist
  - SQL Statements execute and alert administrators

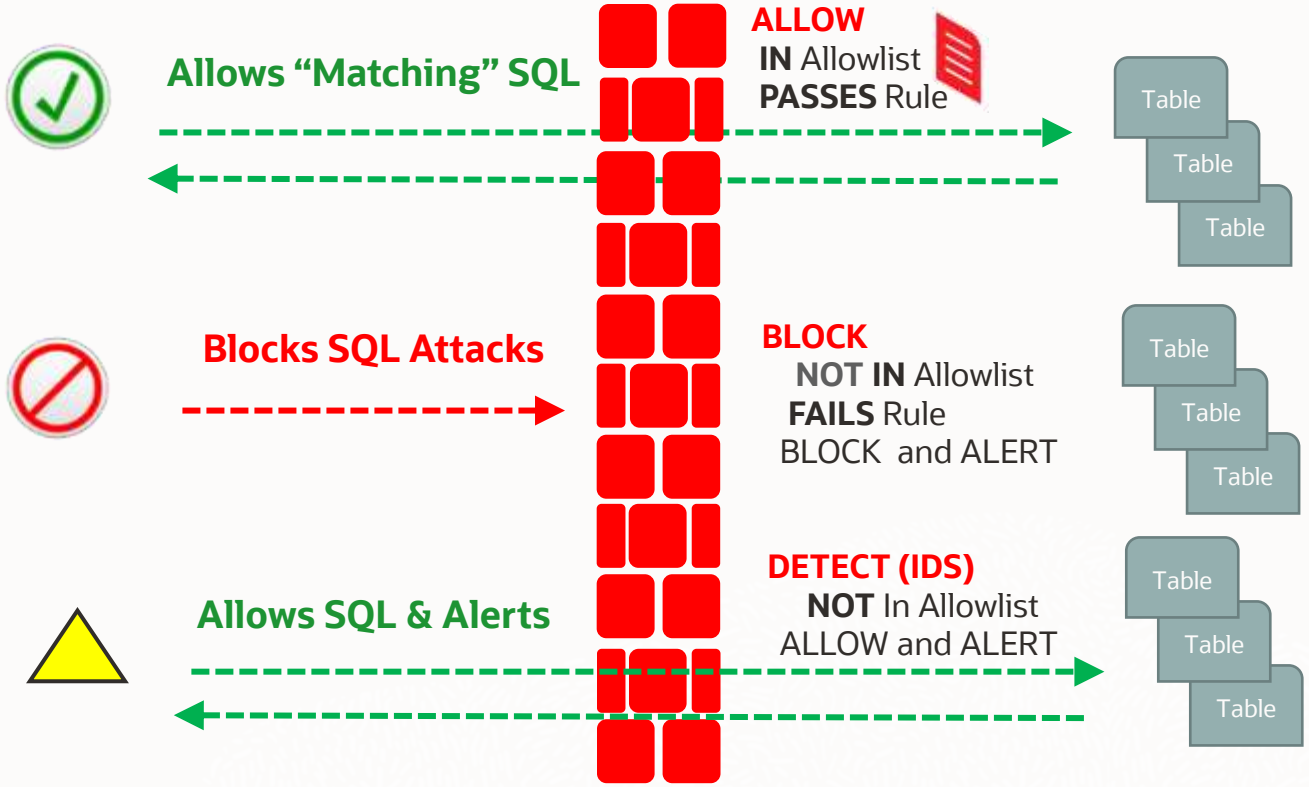


MySQL Enterprise Firewall: Operating Modes

1. ALLOW – Execute SQL
   - SQL Matches Allowlist
   - SQL Passes Rule

2. BLOCK – Block the request
   - Not in Allowlist
   - SQL FAILs Rule
   - In Block Mode

3. DETECT – Execute SQL & Alert
   - Not in Allowlist
   - SQL FAILs Rule
   - In Alert Mode



Database defense

Database Vulnerabilities

- Configurations
  - Set controls and change default setting
- Over Privileged Accounts
  - Use privilege policies
- Weak Access Control
  - use dedicated administrative accounts
- Weak Authentication
  - use strong password enforcement
- Weak Auditing
  - enforce compliance & audit policies
- Lack of Encryption
  - use data, backup, & network encryption
- Proper Credential & Key Management
  - protect passwords, use key vaults
- Unsecured Backups
  - encrypt backups
- No Monitoring
  - security monitoring for users & objects
- Poorly Coded Applications
  - Use database firewall

Database Attacks

- SQL Injection
  - DB Firewall (allow list), Input Validation
- Buffer Overflow
  - Frequently apply Database Software updates, DB Firewall (allow list), Input Validation
- Insider Abuse
  - Tight Access Controls, User specific authentication (no general accounts), Auditing, Monitoring, Encryption
- Brute Force Attack
  - lock out accounts after a defined number of incorrect attempts
- Network Eavesdropping
  - Require SSL/TLS for all Connections and Transport
- Malware
  - Tight Access Controls, Limited Network IP access, Change default settings, Encryption

Database Malicious Actions

- Information Disclosure: obtain credit card and other personal information
  - Encryption – Data and Network, implement Tighter Access Controls
- Denial of Service: run resource intensive queries
  - Resource Usage Limits – Set various limits, e.g. Max Connections, Sessions, Timeouts, …
- Elevation of Privilege: retrieve and use administrator credentials
  - Stronger authentication, Access Controls, Auditing
- Spoofing: retrieve and use other credentials
  - Stronger account and password policies
- Tampering: change data in the database; delete transaction records
  - Tighter Access Controls, Auditing, Monitoring, Backups

Resources

MySQL Secure Deployment Guide

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

60+ blogs to dive into specific topics and features

- https://blogs.oracle.com/mysql/search.html?contentType=Blog-Post&default=security*
- https://dev.mysql.com/blog-archive/?cat=Security

Whitepapers

- https://www.mysql.com/why-mysql/white-papers/#en-22-40

On Demand Webinars

- https://www.mysql.com/news-and-events/on-demand-webinars/#en-20-40

Forums

- https://forums.mysql.com/

Thank you!