ORACLE

Protect Your Organization's Most Valuable Asset with MySQL Enterprise Edition

Nisha Riyaj
MySQL Principal Solution Engineer
nisha.riyaj@oracle.com

Dale Dasker
Manager MySQL Solution Engineering
dale.dasker@oracle.com

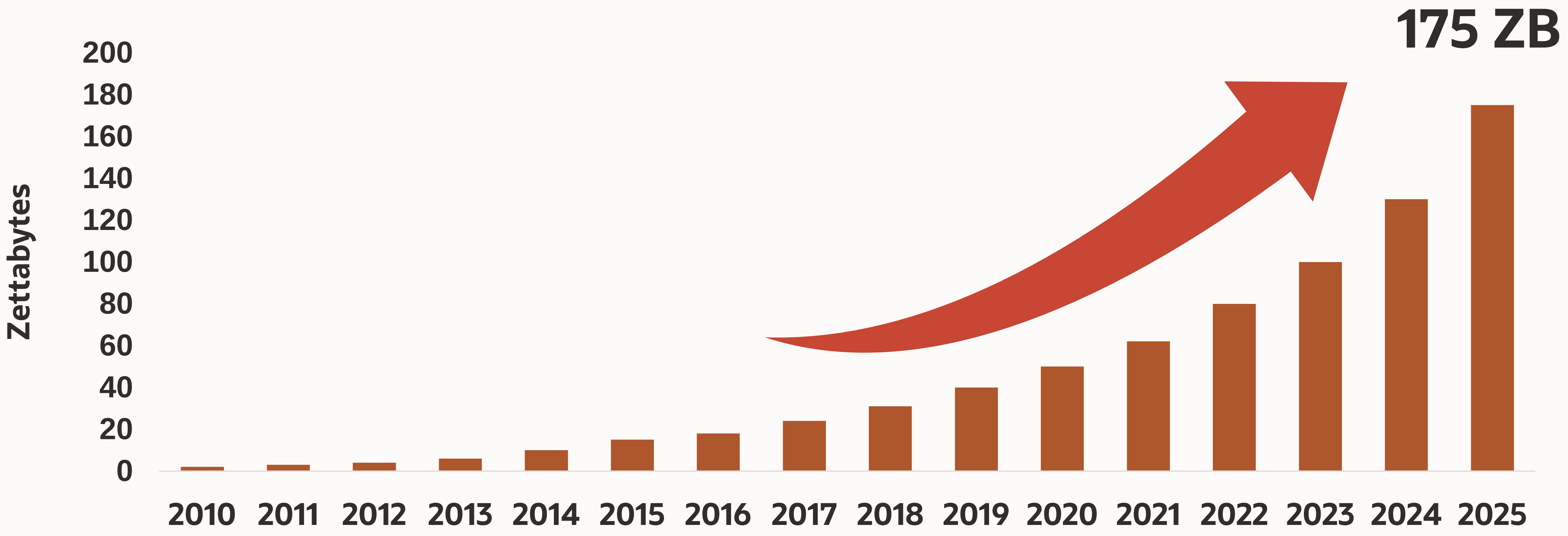
January 25, 2023

Agenda

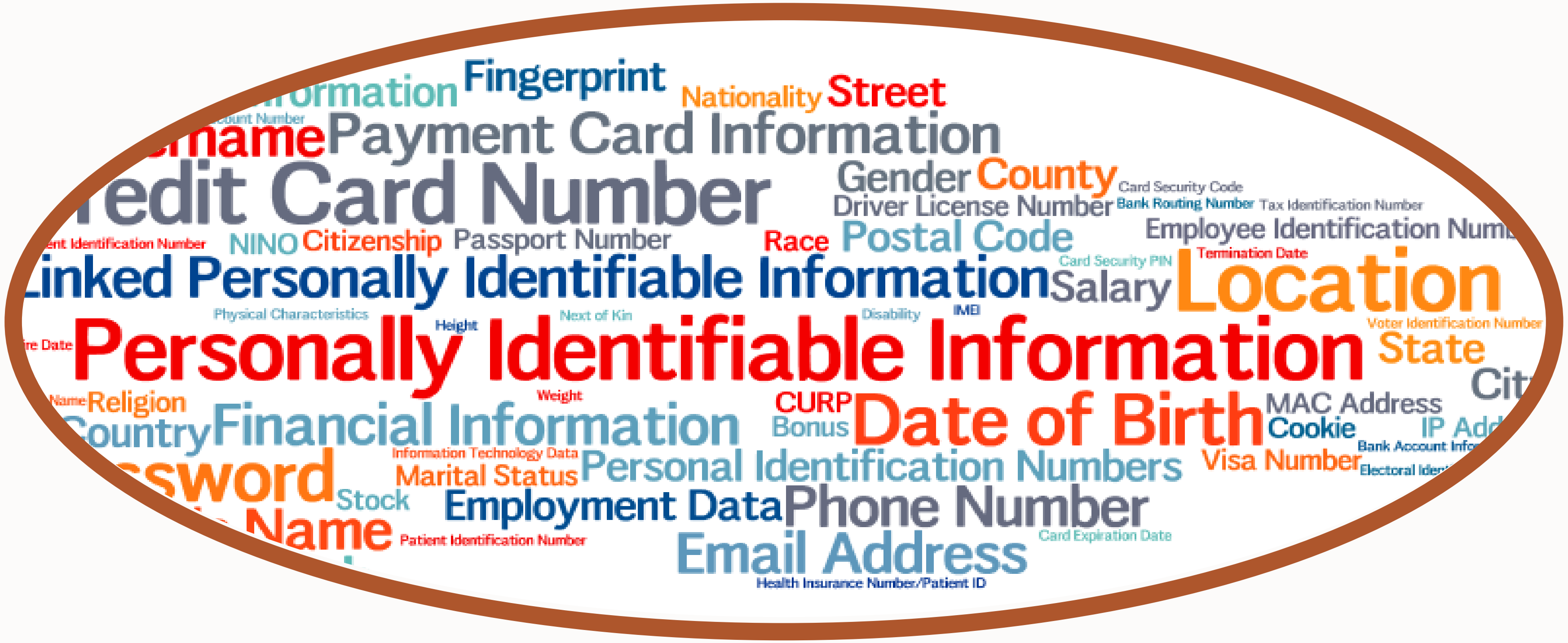
Achieve Compliance with MySQL Enterprise Edition Features

- Workshop Overview
- Setup and Installation of:
  - Enterprise Audit
  - Enterprise Transparent Data Encryption
  - Enterprise FireWall
  - Enterprise Data Masking

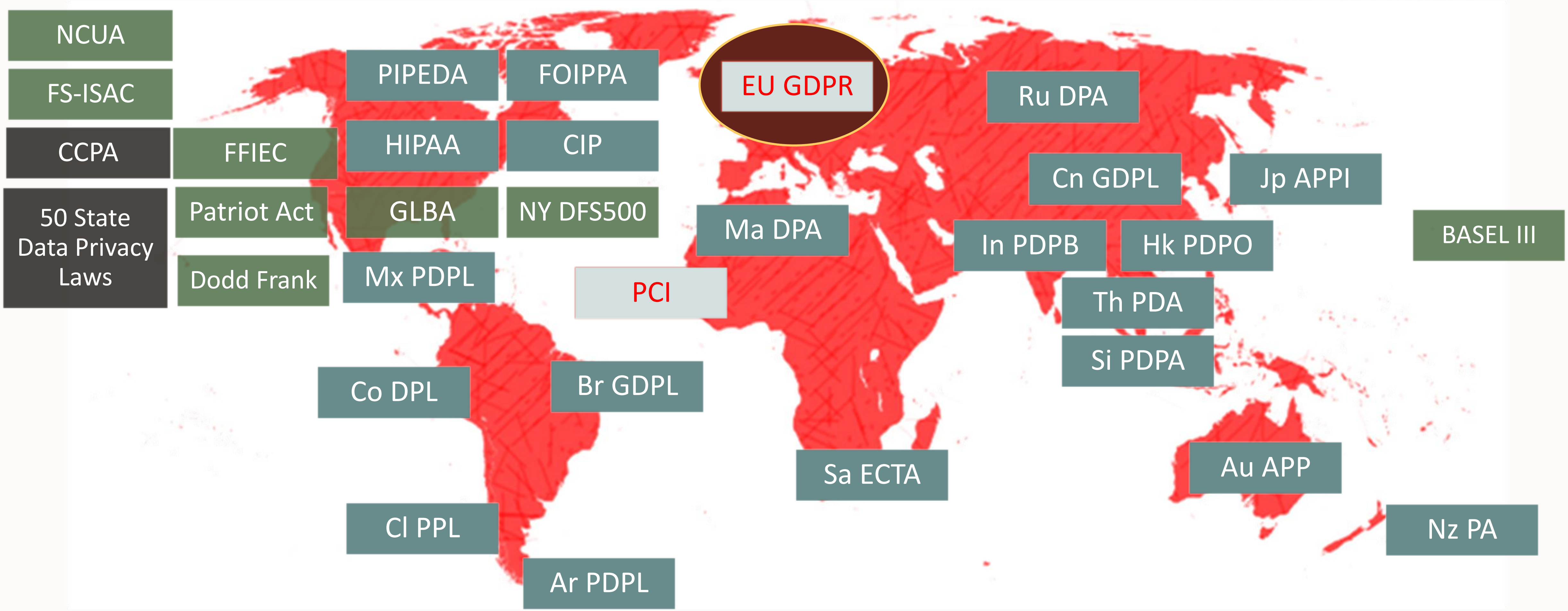
Global Datasphere



Data: Your Most Valuable Asset



Data Security & Privacy Regulations are Prolilferating



EU General Data Protection Regulation (GDPR)

- The E.U. General Data Protection Regulation (GDPR)
- GDPR is a European Union “EU”-wide framework
  - Protection of personal data of EU-based individuals
- Published May 2016, Enforced May 2018
- Fines for GDPR violations are
  - The greater of 20,000,000 Euros or 4% of annual revenue (R150, A83)
- Data must be processed with controls that provide

  “*appropriate security and confidentiality* “
  - Recitals of note - R74-78, R81, R83, R87, R90, A5, A24-25, A28, A32, A35)
- Exact security controls are not specified in the GDPR
  - *WHAT to do*
  - *Not HOW to do it*

EU General Data Protection Regulation (GDPR)

- Data privacy as a fundamental right
- Defines Data protection responsibilities, baselines, principles
- Provides Enforcement Powers

Focus is on 3 Areas

- **Assessment** – Processes, Profiles, Data Sensitivity, Risks
- **Prevention** – **Encryption**, **Anonymization, Access Controls**, Separation of Duties
- **Detection** – **Auditing**, Activity monitoring, Alerting, Reporting

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.)

PCI Security Standards Council™

HIPAA
Health Insurance Portability
and Accountability Act

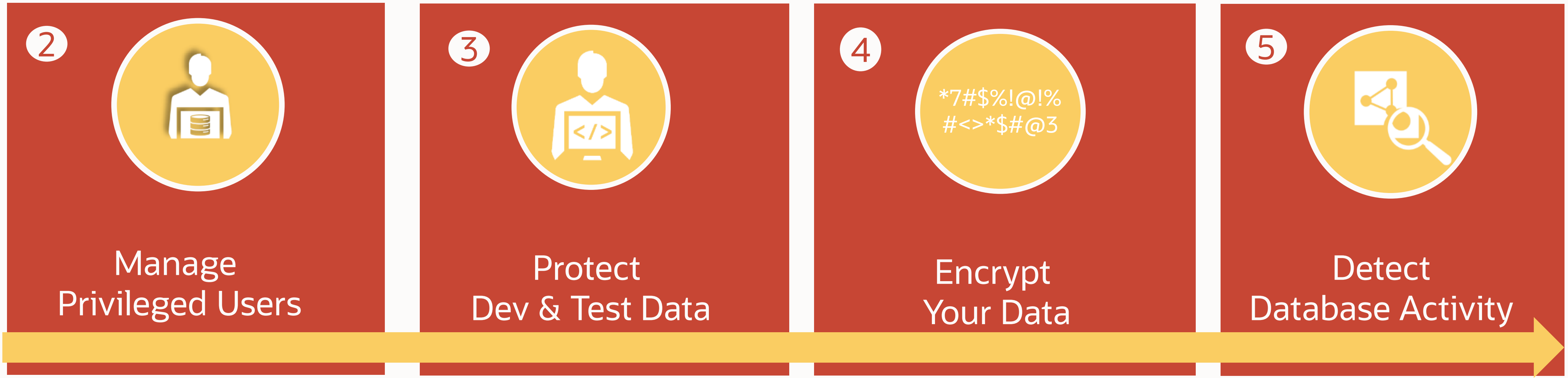
Sarbanes-Oxley
Financial and Accounting Disclosure Information

European Commission
Justice

Data Protection Act 1998

Steps to Database Regulatory Compliance

1 Assess Security Risks: Sensitive Data, Access Privileges, Database Configuration



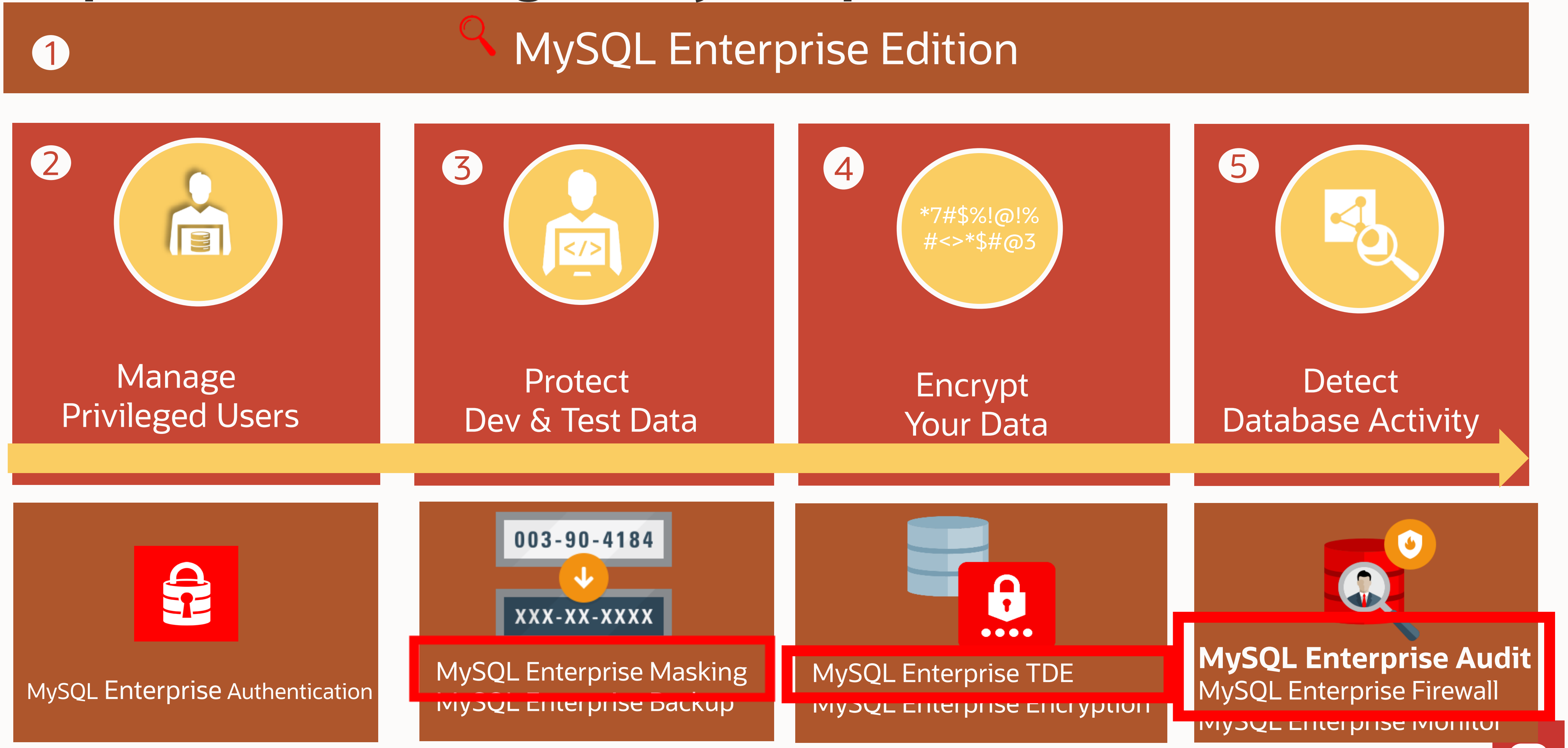
2 Manage Privileged Users

3 Protect Dev & Test Data

4 Encrypt Your Data

5 Detect Database Activity

Steps to Database Regulatory Compliance



1 MySQL Enterprise Edition

2 Manage Privileged Users

3 Protect Dev & Test Data

4 Encrypt Your Data

5 Detect Database Activity

MySQL Enterprise Authentication

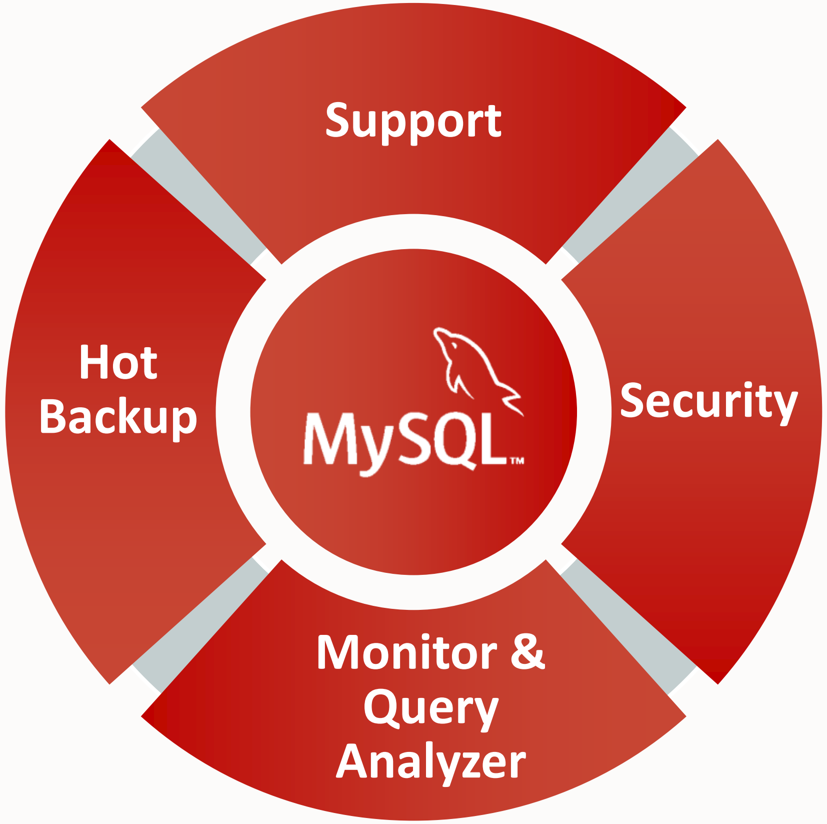
MySQL Enterprise Masking
MySQL Enterprise Backup

MySQL Enterprise TDE
MySQL Enterprise Encryption

MySQL Enterprise Audit
MySQL Enterprise Firewall
MySQL Enterprise Monitor

MySQL Enterprise Audit

MySQL Enterprise Edition



- ✓ Transparent Data Encryption
- ✓ **Audit**
- ✓ MySQL Enterprise Firewall
- ✓ Authentication Plugin
- ✓ Data Masking

MySQL Enterprise Audit

Out-of-the-box logging of connections, logins, and query

Simple to fine grained policies for filtering, and log rotation

Dynamically enabled, disabled: no server restart

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.)

Complete Audit Data

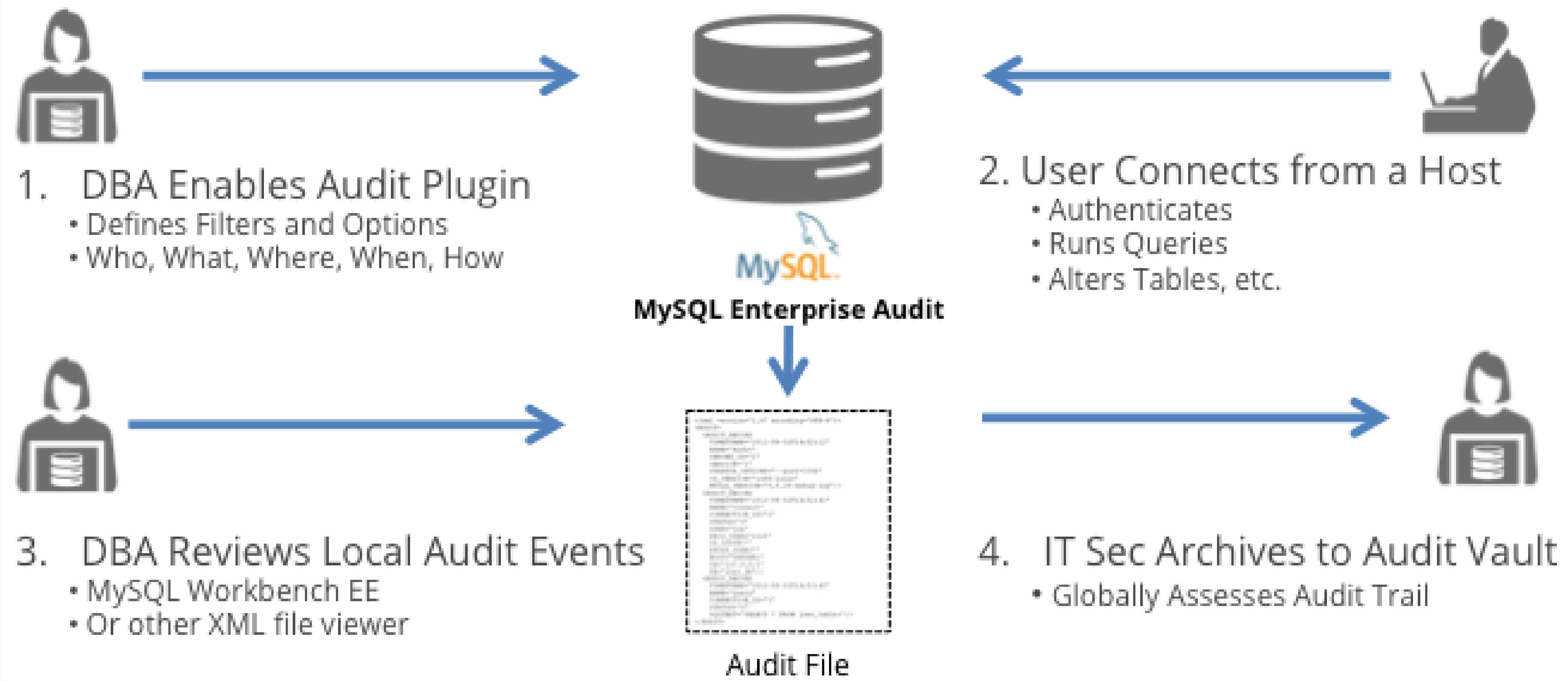
Complete event details

- Who
- What
- When
- How
- Status

- From Where
- 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>
```

MySQL Enterprise Audit - Work Flow



1. DBA Enables Audit Plugin
   - Defines Filters and Options
   - Who, What, Where, When, How

2. User Connects from a Host
   - Authenticates
   - Runs Queries
   - Alters Tables, etc.

MySQL Enterprise Audit

3. DBA Reviews Local Audit Events
   - MySQL Workbench EE
   - Or other XML file viewer

Audit File

4. IT Sec Archives to Audit Vault
   - Globally Assesses Audit Trail

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

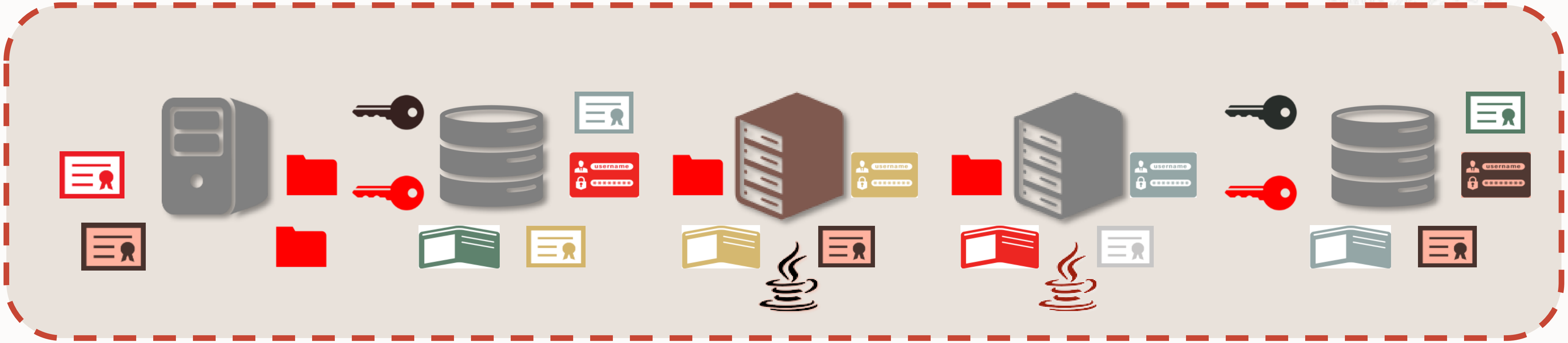
```
{ "timestamp": "2019-10-03 14:21:56",
   "id": 0,
    "class": "audit",
    "event": "startup",
    "connection_id": 0,
    "startup_data": { "server_id": 1,
                              "os_version": "i686-Linux",
                            "mysql_version": "5.7.21-log",
                     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

Audit Log File Formats

Compression and Encryption available

- Compression
  - Based upon gzip
  - audit_log_compression=NONE|GZIP
  - Adds .gz suffix to log files
- Encryption
  - Based upon AES-256-CBC
  - audit_log_encryption=NONE|AES
  - Uses **MySQL keyring plugin**
  - Adds .*pwd_id*.enc suffix to log files

The Challenges of Key Management



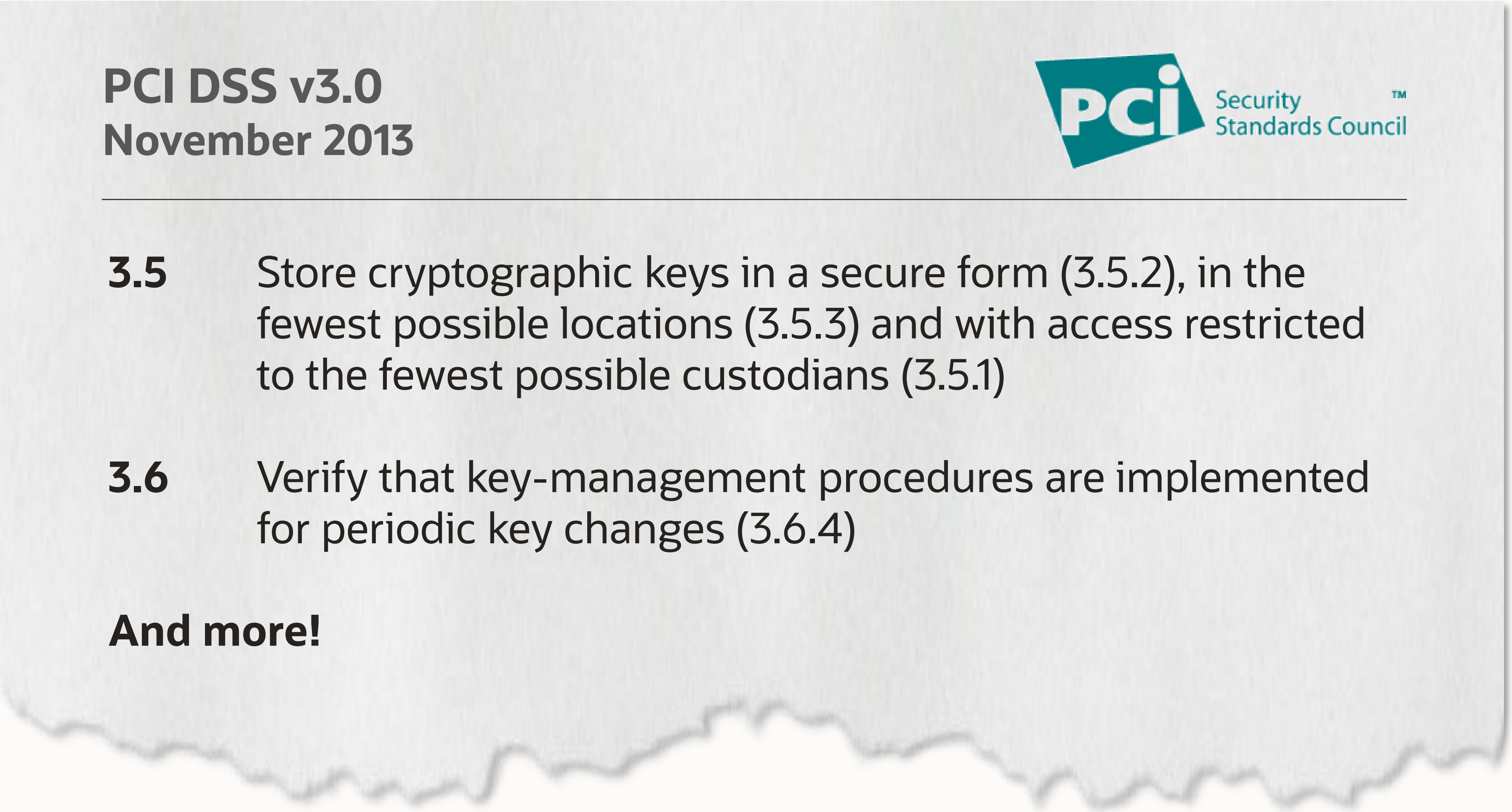
Management

- 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

Regulatory Drivers



PCI DSS v3.0
November 2013

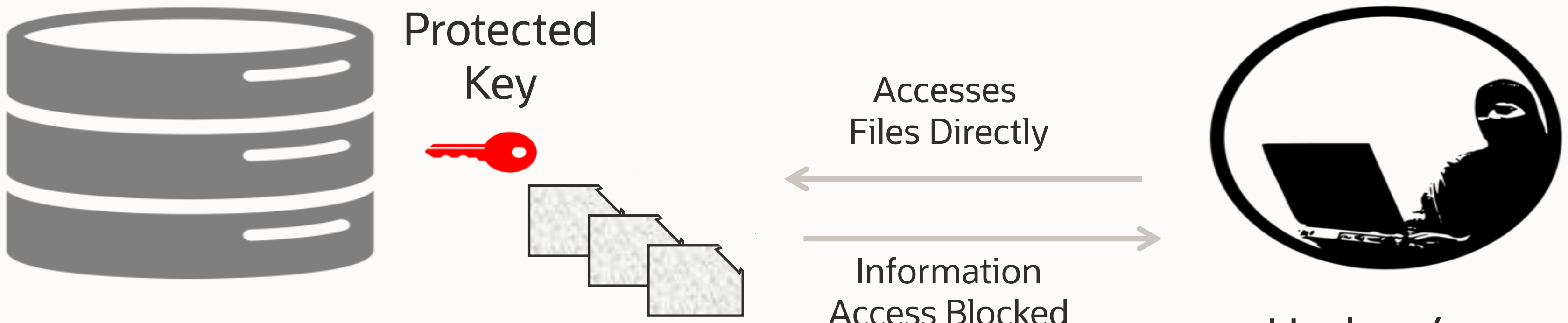
PCI Security Standards Council™

3.5 Store cryptographic keys in a secure form (3.5.2), in the fewest possible locations (3.5.3) and with access restricted to the fewest possible custodians (3.5.1)

3.6 Verify that key-management procedures are implemented for periodic key changes (3.6.4)

And more!

Attack on Files



MySQL Database

Protected Key

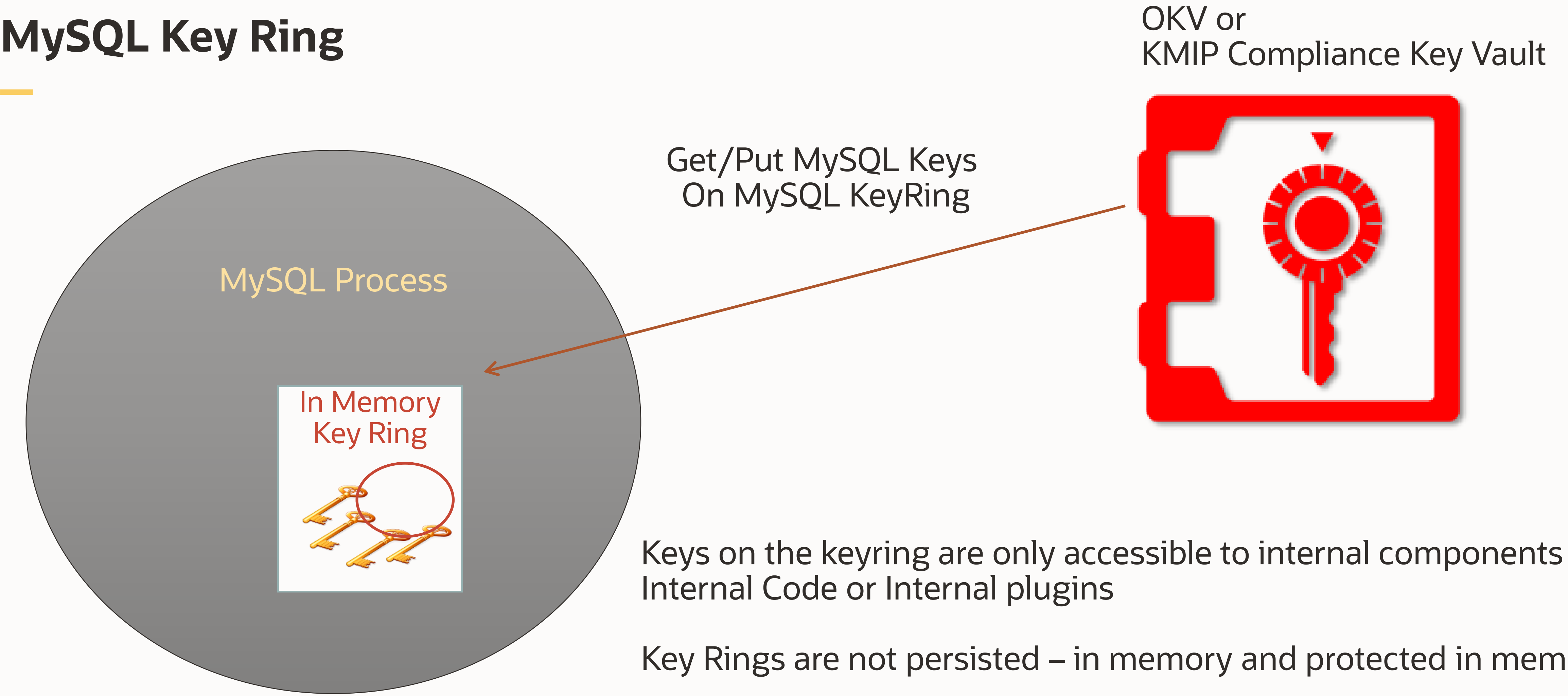
Encrypted Tablespace Files, Redo/Undo, Binlog, System Tables, Audit Logs

Accesses Files Directly

Information Access Blocked By Encryption

Hacker / Dishonest OS User

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

Audit Filtering

Starting with MySQL Enterprise 5.7.13

Allows DBAs to “custom” design audit process

- Use very fine grained rules
  - Reduce audit log file size
  - Reduce File System IO and Storage / Increases performance (less items logged).
  - Increases audit log post processing efficiency – less data to process for immediate answers.
  - Defined using JSON
- Coarse grained rules
  - When you need to watch everything
  - Obsolete. Recommended is to use new audit log filtering.

Audit Log Filters

```
{ "filter": {
    "class": { ”log”: true,
      "name": "connection" } } }
```

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

Connection Event Fields

Name	Type	Description
status	INT	Status of the event: 0: OK, otherwise error state
user.str	STRING	Connecting user string
connection_type	INT	TCP/IP, socket, named pipe, SSL, shared memory
… (many more)		

Table Event Fields

Name	Type	Description
connection_id	STRING	Unique connection id.
sql_command_id	UINT	SQL statement type (SELECT, INSERT…)
query	STRING	Query string accessing the table
table_database	STRING	Database (schema) name
table_name	STRING	Table name
… (many more)		

Filters can be SIMPLE

Log all connection events:

- successful and failed connection attempts
- disconnects
- user change during session (change_user command)

```
{ "filter": {
    "class": { ”log”: true,
      "name": "connection" } } }
```

Filters can be SIMPLE

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

(root@localhost)[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)

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

Filters can be Specific - Log Failed SSL Connects

Log failed SSL connection attempts:

```
{ "filter": {
   "class": {
    "name": "connection",
    "event": {
     "name": "connect",
     "log": {
      "and": [
       { "not": { "field": { "name": "status",
                             "value": 0 } } },
       { "field": { "name": "connection_type",
                    "value": "::ssl" }}]}}}}}}
```

Rules can be Specific related to Tables

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" }}]}}}}}}
```

Comparison Audit to General Log

Connection

Audit Log output:

```
{
    "account": {
        "host": "",
        "user": "root"
    },
    "class": "general",
    "connection_id": 64,
    "event": "status",
    "general_data": {
        "command": "Query",
        "query": "select USER()",
        "sql_command": "select",
        "status": 0
    },
    "id": 2,
    "login": {
        "ip": "10.20.1.1",
        "os": "",
        "proxy": "",
        "user": "root"
    },
    "timestamp": "2019-12-19 00:43:02"
}
```

General Query Log output:

```
2019-12-19T00:43:02.532984Z 64 Connect   root@10.20.1.1 on  using SSL/TLS
2019-12-19T00:43:02.533608Z 64 Query     select @@version_comment limit 1
2019-12-19T00:43:02.551259Z 64 Query     select USER()
2019-12-19T00:43:15.373949Z 60 Quit
```

- *Not as detailed*
- *No means for filtering content*
- *Can be easily disabled*
- *No log management*

Connection Attributes 8.0.19

As of MySQL 8.0.19, events with a class value of connection and event value of connect may include a connection_attributes item to display the connection attributes passed by the client at connect time. (For information about these attributes, which are also exposed in Performance Schema tables, see Section 26.12.9, “Performance Schema Connection Attribute Tables”.)

Example:

```
"connection_attributes": {
  "_pid": "43236",
  "_os": "osx10.14",
  "_platform": "x86_64",
  "_client_version": "8.0.19",
  "_client_name": "libmysql",
  "program_name": "mysqladmin"
}
```

MySQL Transparent Data Encryption

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 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 Enteprise Audit Logs
- MySQL Enterprise Backup Files
- Note: DBAs can optionally force Table Encryption
  - i.e. Users can only create encrypted tables

MySQL Enterprise Security Transparent Data Encryption

Plugin Infrastructure

- New plugin type : keyring
- Ability to load plugin before InnoDB initialization : --early-plugin-load

Keyring plugin

- Used to retrieve keys from Key Stores
- Over Standardized KMIP protocol

Oracle Key Vault (OKV)

Gemalto Safenet KeySecure

Fornetix Key Orchestration Appliance

AWS KMS

SQL

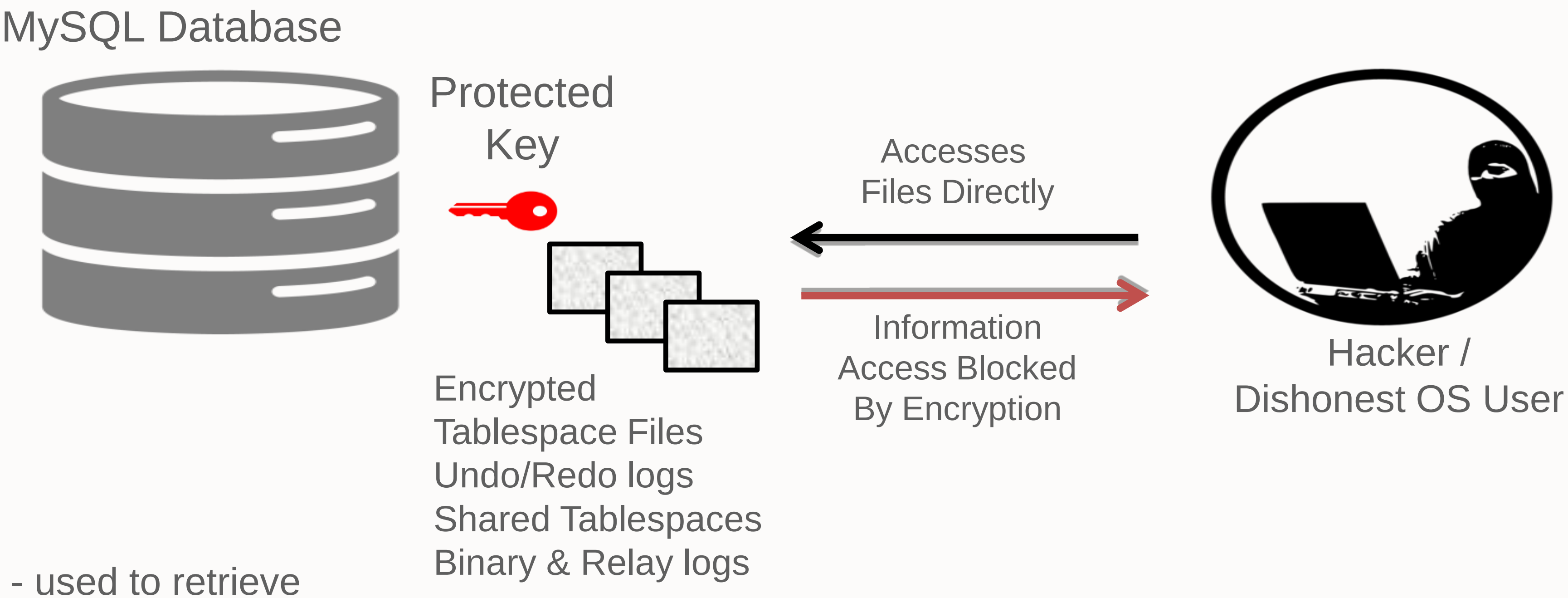
- New option in CREATE TABLE ENCRYPTION=“Y”
- New SQL : ALTER INSTANCE ROTATE INNODB MASTER KEY

InnoDB

- Support for encrypted tables
- IMPORT/EXPORT of encrypted tables
- Support for master key rotation

MySQL Enterprise Transparent Data Encryption (TDE)

Protects against Attacks on Database Files



MySQL Database

Protected Key

Encrypted
Tablespace Files
Undo/Redo logs
Shared Tablespaces
Binary & Relay logs

Accesses Files Directly

Information Access Blocked By Encryption

Hacker / Dishonest OS User

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

MySQL Enterprise Transparent Data Encryption (TDE)

KMIP Compliant

- KMIP – Key Management Interoperability Protocol (Oasis Standard)
  - Keys are protected and secure
- **KMIP** mode tested with the following products
  - Oracle Key Vault (OKV)
  - HashiCorp Vault
  - Gemalto KeySecure
  - Fornetix Key Orchestration Appliance
  - Thales Vormetric Key Management Server
- Enables customers to meet regulatory requirements

- Additional Options
  - Key Ring File
  - Encrypted Key Ring File

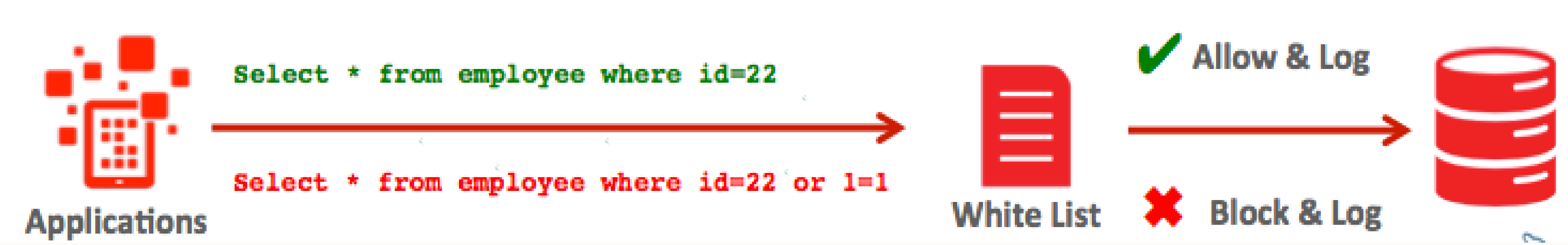
Also

  - Cloud Key Services (AWS)
- https://dev.mysql.com/doc/refman/8.0/en/keyring.html

MySQL Enterprise FireWall

MySQL Enterprise FireWall

- Allow
- Block
- Detect



```
Select * from employee where id=22
```

```
Select * from employee where id=22 or 1=1
```

MySQL Enterprise FireWall

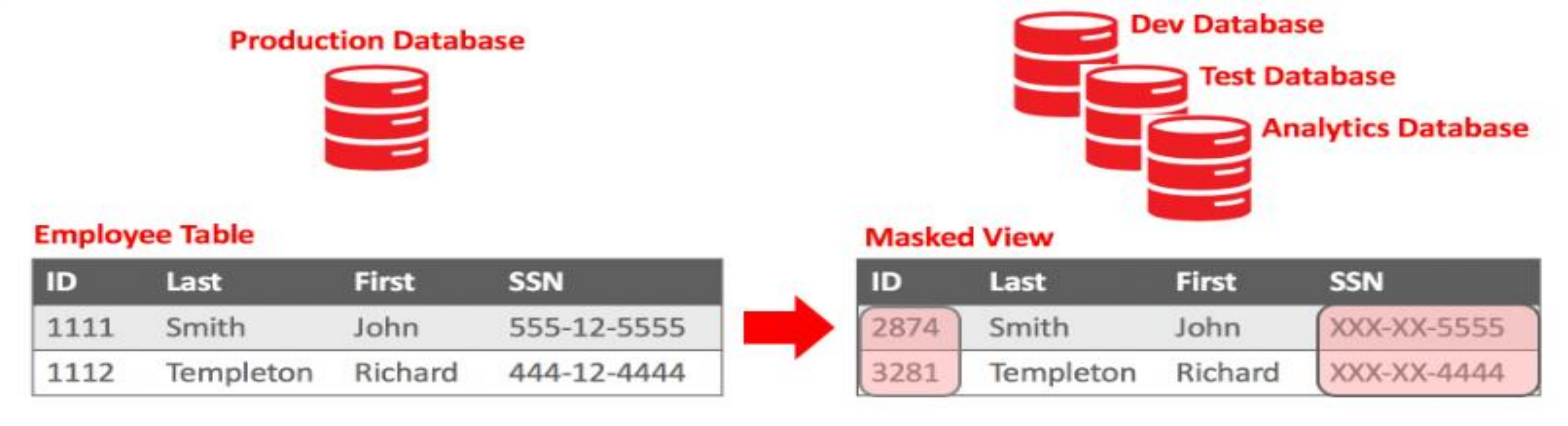
- Group Profiles
- Block SQL Injection Attacks
- Database Intrusion Detection
- Real-time Threat Monitoring

- Block Suspicious Traffic
- Learn and Build allowlists
- Transparent Protection
- High Performance
- Logging

MySQL Enterprise Data Masking

MySQL Enterprise Data Masking

Masking and De-identification of Data



Employee Table

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

Masked View

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

MySQL Enterprise Masking and De-identificaiton protects sensitive data from unauthorized users.

MySQL Enterprise Data Masking

- **Meet regulatory requirements and data privacy laws**
  - Such as GDPR, PCI DSS and HIPPA that require data de-identification.

- **Significantly reduce the risk of a data breach**
  - By preventing unauthorized access to confidential data.

- **Protect confidential information**
  - While improving development, test and analytics environments.

- **Built-in & Easy to Use**
  - Implemented in the MySQL Server itself, so the masking logic is centralized.

- **Robust Data Masking Functions**
  - Can hide or obfuscate sensitive data, by controlling how the data appears.

MySQL Enterprise Edition – SECURITY

MySQL Enterprise TDE

- Data-at-Rest Encryption
- Key Management/Security

MySQL Enterprise Encryption

- Public/Private Key Cryptography
- Asymmetric Encryption

MySQL Enterprise Authentication

- External Authentication Modules
  - Microsoft AD, Linux PAMs, LDAP

MySQL Enterprise Data Masking

MySQL Enterprise Firewall

- Block SQL Injection Attacks

MySQL Enterprise Audit

MySQL Enterprise Monitor

- Changes in Database Configurations, Users Permissions, Database Schema, Passwords

MySQL Enterprise Backup

- Securing Backups, AES 256 encryption

MySQL Enterprise Thread pool

- Attack Hardening

Security Resources

SECURITY MUST READ

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

Also

http://mysqlserverteam.com/

https://www.mysql.com/why-mysql/#en-0-40

https://www.mysql.com/why-mysql/presentations/#en-17-40

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

Workshop Overview

- **Goals**
  - Create a OCI Compute server for hosting MySQL Enterprise Edition
  - Install MySQL Enterprise Edition
  - Overview and Setup of ..
    - Enterprise Audit
    - Enterprise Transparent Data Encryption
    - Enterprise FireWall
    - Enterprise Data Masking
- **What this Workshop is not:**
  - In-depth tutorial on Oracle Cloud Infrastructure
  - MySQL Training Class
- **Lab:**
  - https://bit.ly/MySQL_Workshop_Security

Thank you