ORACLE

# Level UP to Enterprise Level Security with MySQL

## Best Practices

**박혜선**

Oracle, MySQL

Jan, 2023

# Agenda

1. Trends and Challenges
2. MySQL Security Overview
3. MySQL Enterprise Edition Security
4. Security Tips

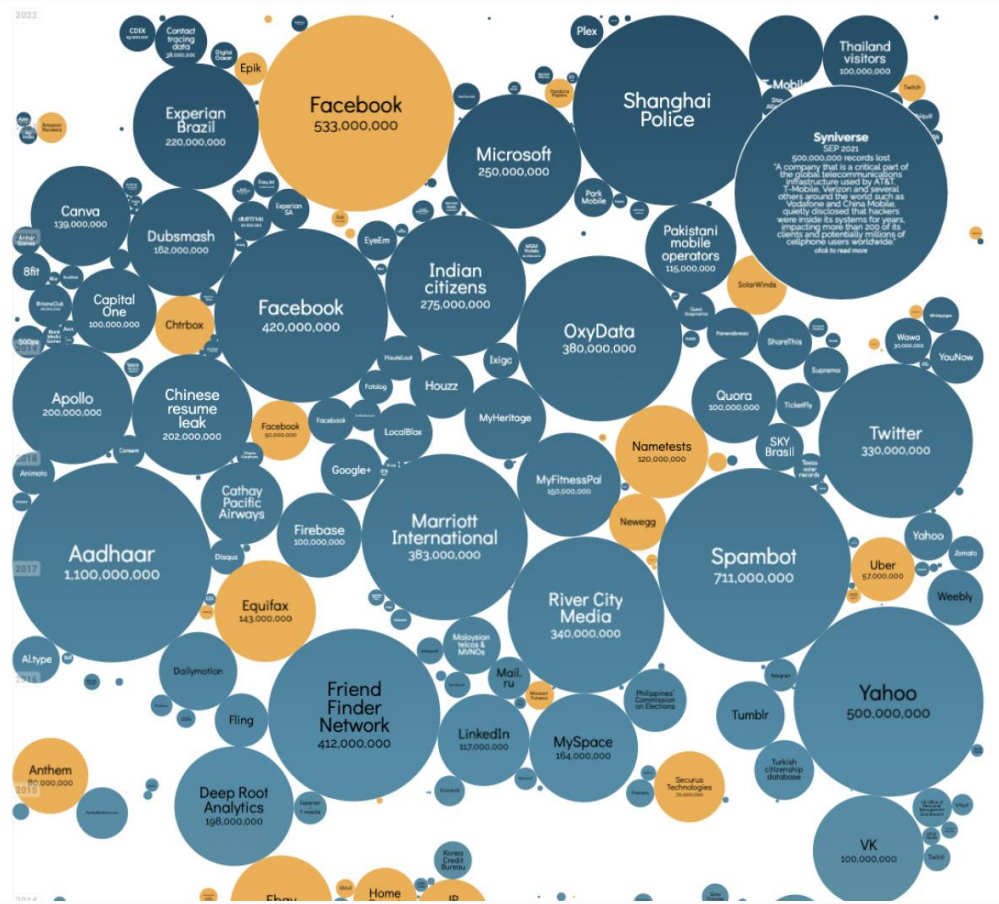



# Trends & Challenges




# 세계에서 가장 큰 데이터 침해 및 해킹

https://informationisbeautiful.net/visualizations/worlds-biggest-data-breaches-hacks/



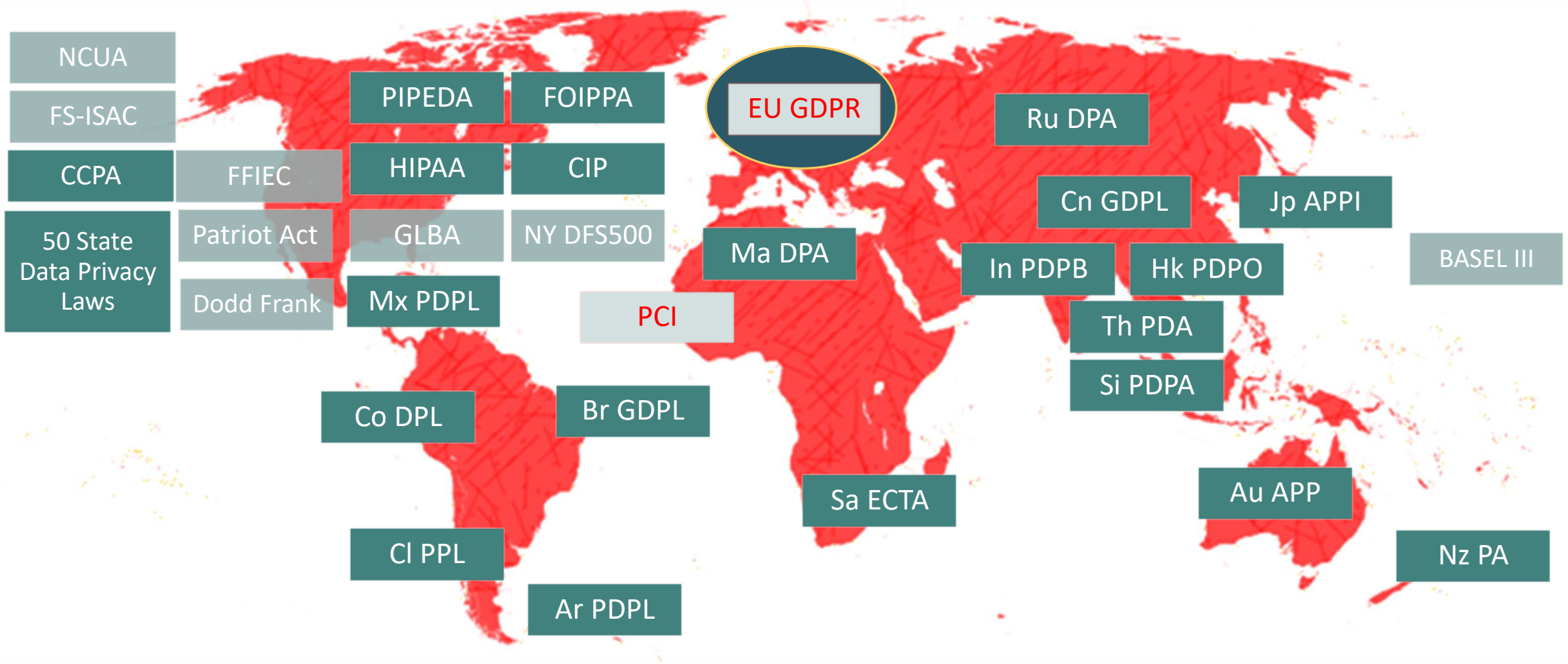

**$4.35 million** 연평균 데이터 침해 비용

**83%** 데이터 유출 피해를 1회 이상 경험

**$4.54 million** 랜섬웨어 공격의 평균 비용,
랜섬웨어 요구액은 포함되지 않음

**93%** 의 보안 침해는 예방 가능하였음
-- Online Trust Alliance (Internet Society)

# Data Security & Privacy Regulations are Proliferating



# Security is Job #1
# Data is the Most Valuable Asset

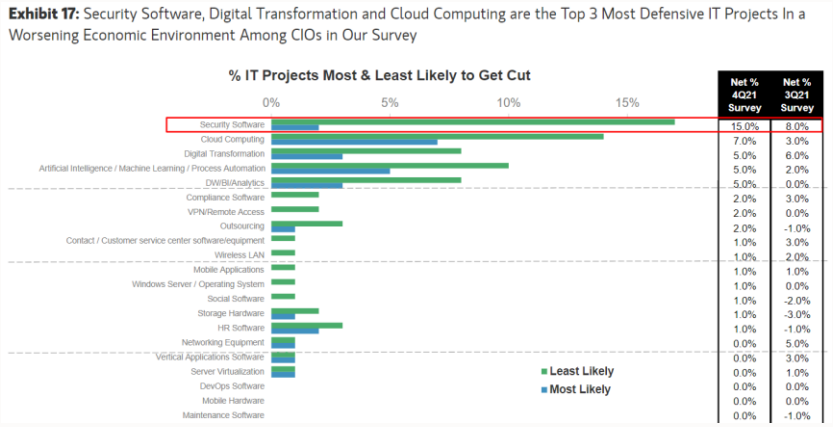
https://www.cio.com/article/303215/8-top-priorities-for-cios-in-2022.html

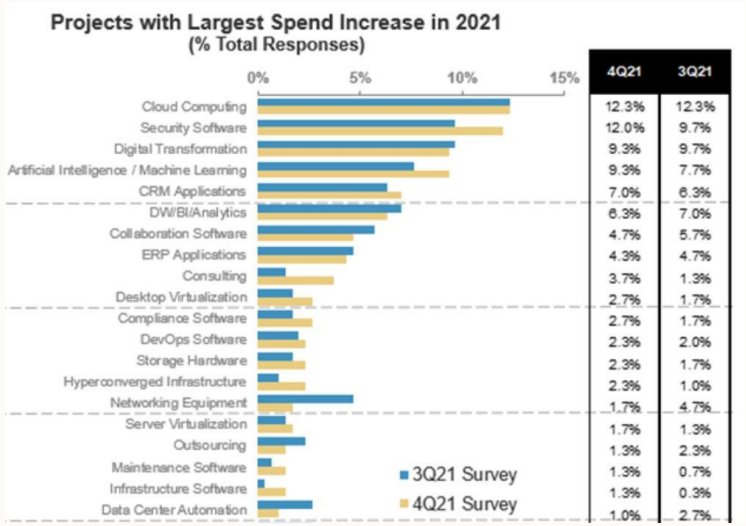

*"Keep the organization safe (cybersecurity/cyber resilience/GDPR compliance/data protection compliance) "*

Was
#1 – Security – in 2019

2 years later

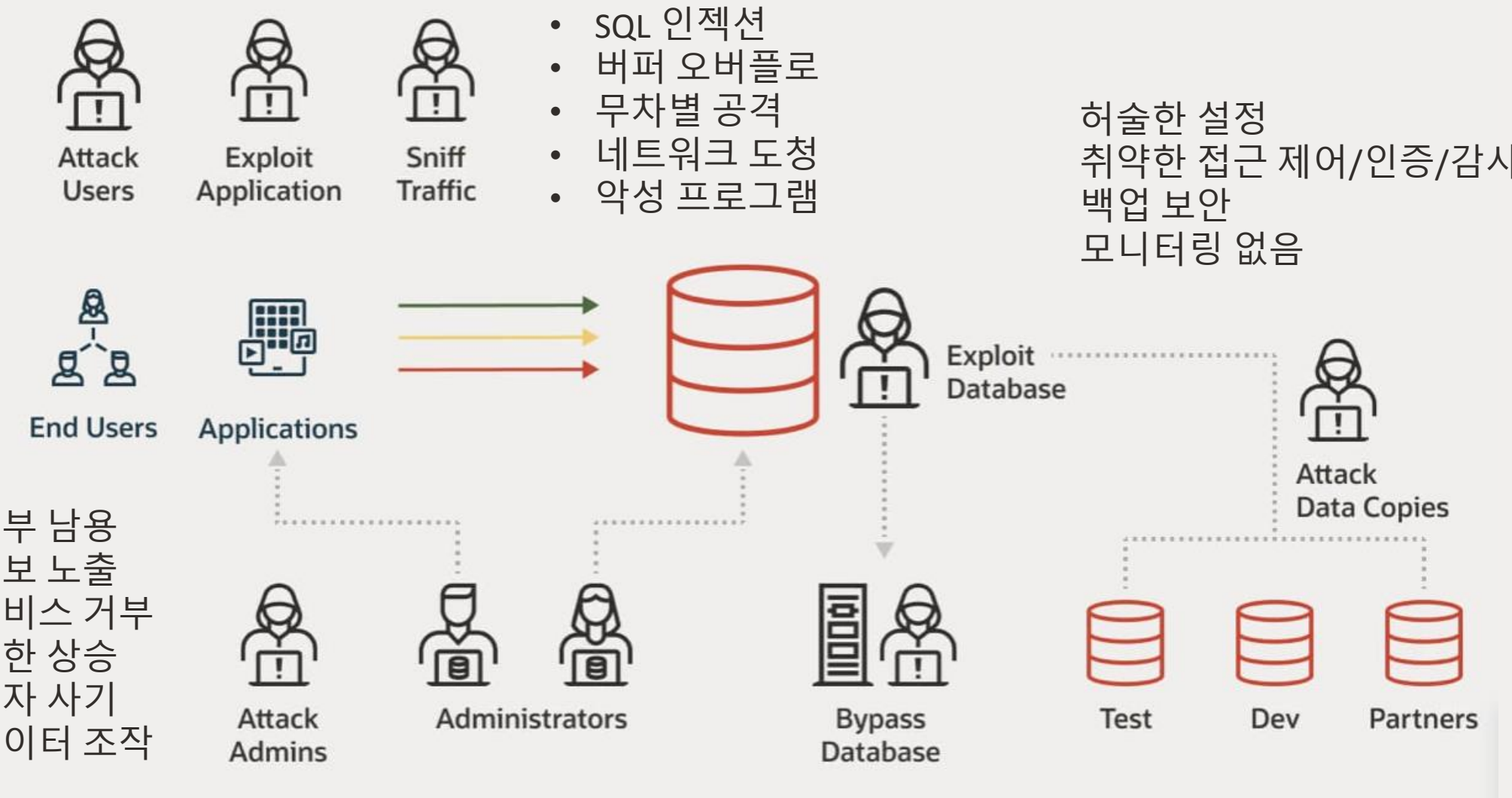
#1 – Security




And on par for spending
Increase with Cloud

# Where are databases attacked?




- SQL 인젝션
- 버퍼 오버플로
- 무차별 공격
- 네트워크 도청
- 악성 프로그램

허술한 설정
취약한 접근 제어/인증/감사
백업 보안
모니터링 없음

- 내부 남용
- 정보 노출
- 서비스 거부
- 권한 상승
- 전자 사기
- 데이터 조작

# 보안 규정에서 요구되는 보안 단계

## 평가

- 리스크와 취약성을 찾아내고 필요한 보안 통제가 있는지 확인합니다.

## 방어

- 암호화 방식, 사용자 제어, 접근 제어 등을 사용합니다.

## 감지

- 여전히 위반의 가능성이 있습니다. 감사, 모니터링, 경고 혹은 알람이 있는지 확인합니다.

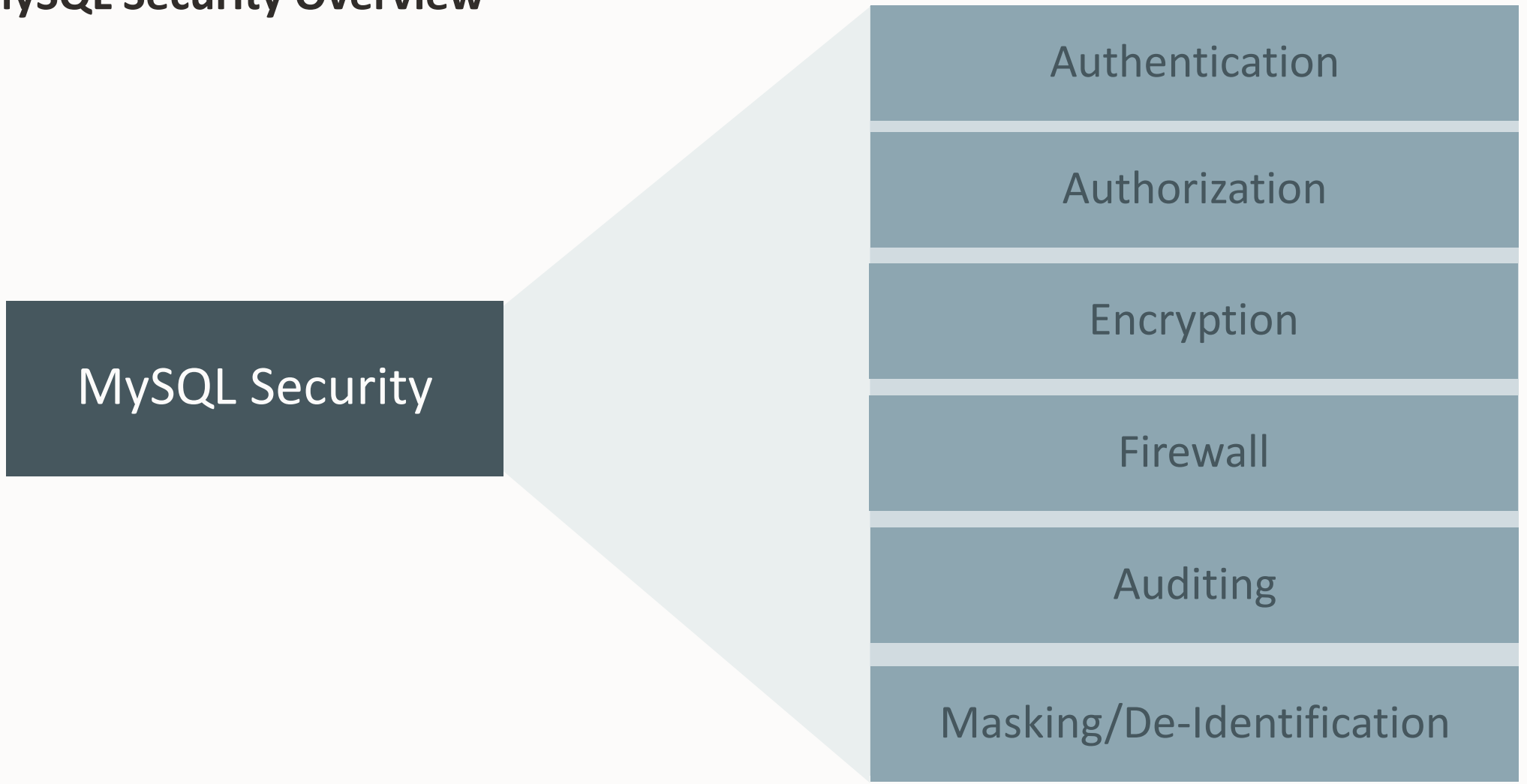
## 복구

- 보안 사고로 인해 서비스, 및 메인 데이터베이스가 중단되지 않도록 합니다.
- 증거 보류 – 사후 분석 –보안 취약점 수정 합니다.

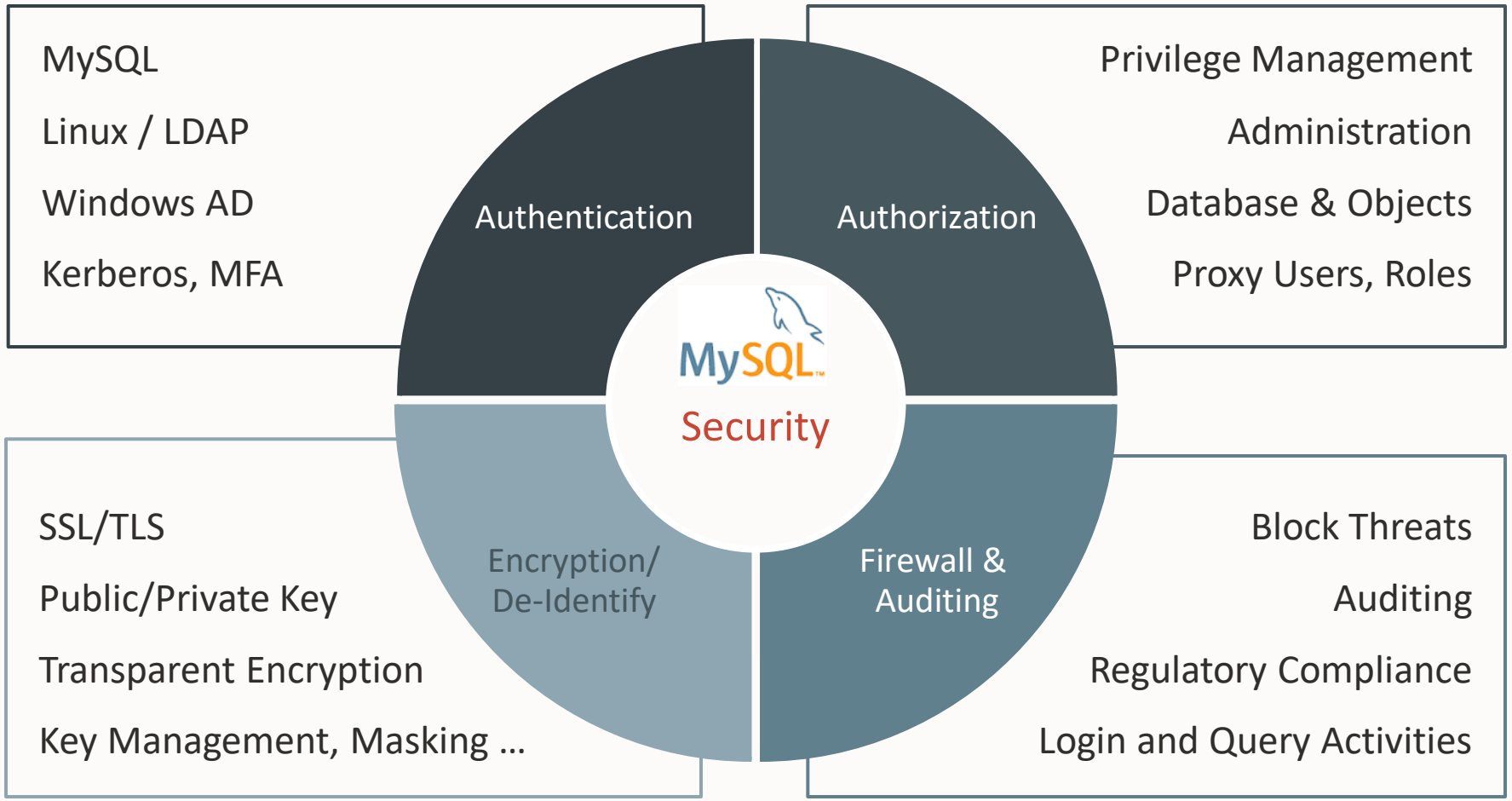
# MySQL Security Overview




# MySQL Security Overview



# MySQL Security Overview



# How to

Examples




# Controlling Access and Rights

## Authentication & Authorization




# 대부분 작업을 SQL방식으로 수행……

1. DBA는 mysql 실행 중 OS에 SSH/로그인 할 필요가 없습니다.
   - This is common.
2. OS 관리자들은 MySQL을 건드릴 필요가 없습니다.
   - OS 감사는 초기 설치 후 부터 포함되어야 합니다.
   - 명령문은 노출되지 않아야 합니다.
3. 모든 DBA 작업은 감사되어야 합니다.
   - MySQL 감사는 SQL을 통해 dba 가 수행하는 모든  명령문을 캡처할 수 있습니다.
4. DevOps Friendly – Service oriented.
5. 반복적으로 평가하고 및 자동 수정에 유리합니다.

# First thing

Secure the Root password

root 계정의 관리:

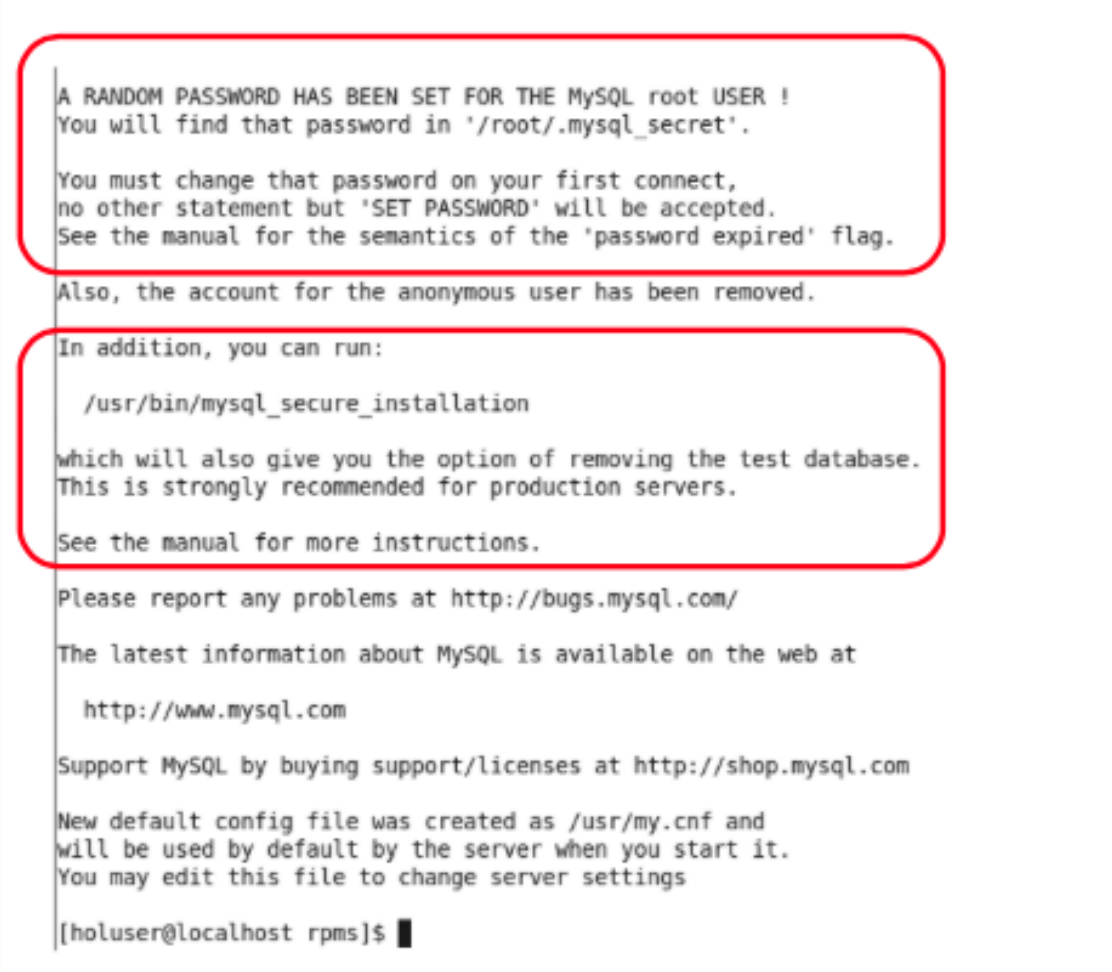
- 초기 비밀 번호 변경:

$ mysql_secure_installation --host=::1 --port=3307 –p

SQL> SET PASSWORD = PASSWORD('secret');

- 필요 없는 글로벌 root계정 확인 후 삭제

SQL> SELECT user,host FROM mysql.user WHERE user='root' and host<>'localhost';



# MySQL Authentication

Built in 인증

- 유저 테이블에 유저 명, 호스트 명 및 암호 해시 값 저장

MySQL SHA 256

- SHA-256 해싱 및 유저 기반 salting
- 더 빠른 캐시 활용 – can handle very high connect and disconnect counts

X.509

- 서버에서 인증서 기반으로 client에 대해 인증

외부 인증 - **MySQL Enterprise Authentication**

- Microsoft Active Directory
- Native LDAP – User/Password, SASL, GSSAPI/Kerberos via LDAP SASL, Kerberos
- Native Kerberos
- Linux PAMs (Pluggable Authentication Modules)

# MySQL 사용자

누구, 어떤 유형, 어디서/어떻게 인증?

- 내부 사용자
- X.509를 사용한 내부 사용자
- 외부 인증 사용자
- Proxy 사용자




# 내부 사용자

Authenticated internally

```
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')
and (plugin not like 'auth%'
and plugin <>
'mysql_no_login') and
length(authentication_string)
> 0 order by plugin;
```

Result Grid | Filter Rows: Search | Export:

| host | user | plugin | HASHTYPE |
|---|---|---|---|
| % | mysql.yse | caching_sha2_password | STRONG SHA2 |
| % | newuserq | caching_sha2_password | STRONG SHA2 |
| % | newuserw | caching_sha2_password | STRONG SHA2 |
| localhost | dev1 | caching_sha2_password | STRONG SHA2 |
| localhost | mysql.sys | caching_sha2_password | STRONG SHA2 |
| localhost | read_user1 | caching_sha2_password | STRONG SHA2 |
| localhost | read_user2 | caching_sha2_password | STRONG SHA2 |
| localhost | root | caching_sha2_password | STRONG SHA2 |
| localhost | rw_norole_user | caching_sha2_password | STRONG SHA2 |
| localhost | rw_user1 | caching_sha2_password | STRONG SHA2 |
| % | newuser | mysql_native_password | WEAK SHA1 |
| % | newuserr | mysql_native_password | WEAK SHA1 |
| % | root2 | mysql_native_password | WEAK SHA1 |
| % | root3 | mysql_native_password | WEAK SHA1 |

확인 사항:

- 모르는 계정을 잠그고, 확인후 삭제
- 특정 호스트를 지정한 계정 생성
- MySQL 5.7 에서 생성된 계정을 8.0 으로 업그레이드

https://mysqlserverteam.com/mysql-8-0-4-new-default-authentication-plugin-caching_sha2_password/

# 내부 사용자

## Proxy 및 X509/FIPS 인증서 인증 사용자

```
SELECT `user`.`Host`, `user`.`User`, `user`.`ssl_type`,
CAST(`user`.`x509_issuer` as CHAR) as Issuer,
CAST(`user`.`x509_subject` as CHAR) as Subject
FROM `mysql`.`user` where (user not like 'mysql.%') AND ssl_type='X509';

SELECT VARIABLE_NAME, VARIABLE_VALUE, FIPS Mode' as Note,
IF(VARIABLE_VALUE = 'ON' OR VARIABLE_VALUE = 'STRICT', 'Yes', 'No')
FROM performance_schema.global_variables
where variable_name = 'ssl_fips_mode';

SELECT * FROM mysql.proxies_priv where grantor<>'root@';
```

# 외부 인증 사용자

Globally manage – map to Enterprise, Use stronger Options

LDAP, Windows AD SSPI,  Kerberos, FIDO2 – Many Options

```
SELECT `user`.`Host`, `user`.`User`,`user`.`plugin`,
`user`.`authentication_string` from mysql.user where plugin like
'auth%';
```

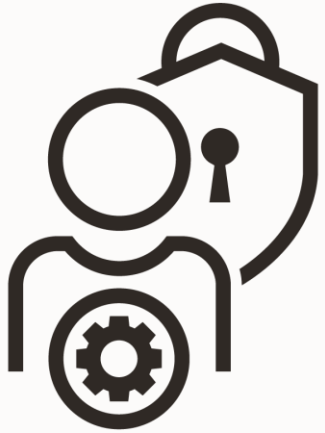
| Host | User | plugin | authentication_string |
|---|---|---|---|
| localhost | betsy | authentication_ldap_simple | uid=betsy_ldap,ou=People,dc=example,dc=com |
| NULL | NULL | NULL | NULL |

많은 회사에서 외부 인증 사용, 특히 내부 사용자인 DBAs 혹은  개발자

LDAP 에서 관리, 실제 사용자에 대한 감사 수행

관련 사용자 혹은 해당 부서에서 MySQL접근이 필요

# MySQL 암호 정책 컴퍼넌트



모든 계정에 비밀 번호를 설정
암호 인증 플러그인 설치

```
$ mysql_secure_installation --host=::1 --port=3307 -p
INSTALL COMPONENT 'file://component_validate_password';
```

암호 유효 기간 및 Rotation 수행

- 암호 재 설정 요구

```
Set global default_password_lifetime = 180;
```

| VARIABLE_NAME | VARIABLE_VALUE |
|---|---|
| default_password_lifetime | 0 |
| validate_password.check_user_name | ON |
| validate_password.dictionary_file | <FILENAME OF DICTIONARY FILE |
| validate_password.length | 8 |
| validate_password.mixed_case_count | 1 |
| validate_password.number_count | 2 |
| validate_password.policy | LOW |
| validate_password.special_char_count | 0 |
| NULL | NULL |

계정에 대한 잠금 조치 룰(in v. 5.7)
암호 시도 룰 (in v. 5.7.16+)

- 인증 실패 회수에 따른 잠금

```
CREATE USER 'u1'@'localhost' IDENTIFIED BY 'password' FAILED_LOGIN_ATTEMPTS 3 PASSWORD_LOCK_TIME 3;
```

# Multi-Factor 인증 및 듀얼 패스워드

## 멀티 인증:

CREATE USER 'alice'@'localhost' IDENTIFIED WITH caching_sha2_password BY '*sha2_password*' AND IDENTIFIED WITH authentication_ldap_sasl AS 'uid=u1_ldap,ou=People,dc=example,dc=com’;

ALTER USER 'alice'@'localhost' ADD 3 FACTOR IDENTIFIED WITH authentication_fido;

## 듀얼 패스워드

ALTER USER ….IDENTIFIED BY '*auth_string*' [REPLACE '*current_auth_string*'] [**RETAIN CURRENT PASSWORD**]

ALTER USER ... **DISCARD OLD PASSWORD**;

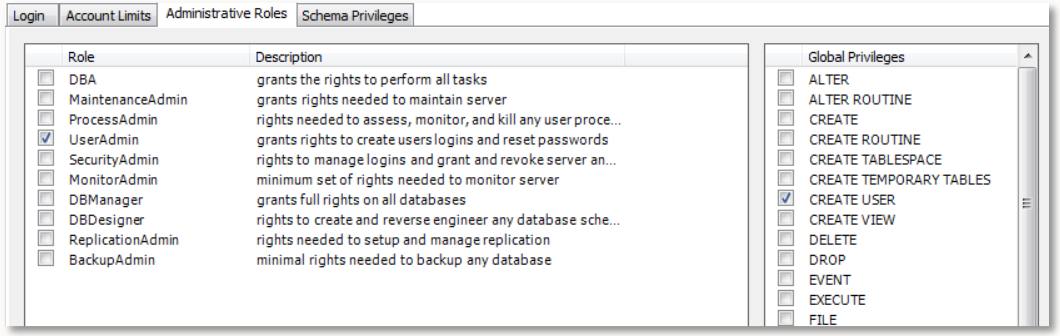
## 계정 코멘트

CREATE/ALTER USER . . . COMMENT ‘Something about the user/account’;

CREATE/ALTER USER …. ATTRIBUTE ‘json object’;

# MySQL 권한 관리

1. 관리자 권한
2. 데이터베이스 권한
3. 세션 제한 및 오브젝트 권한
4. 사용자 권한에 대한 세분화된 제어
   - Create, Alter , Delete databases
   - Create, Alter , Delete tables
   - INSERT, SELECT, UPDATE, DELETE queries
   - Create, Execute, Delete stored procedures and with what rights
   - Create, Delete indexes




*Security Privilege Management in MySQL Workbench*

# MySQL 권한 관리 테이블

## user

- User Accounts
- Global Privileges
- Auth Type(s)

## db

- Database Level Privileges
- Database, Tables, Objects
- User and host

## tables_priv

- Table level privileges
- Table and columns

## columns_priv

- Specific columns

## procs_priv

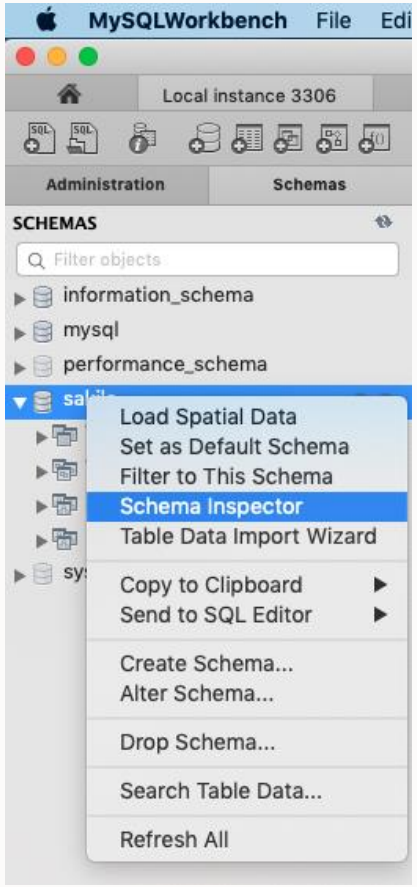
- Stored Procedures
- Functions
- Single function privilege

## proxies_priv

- Proxy Users
- Proxy Privileges



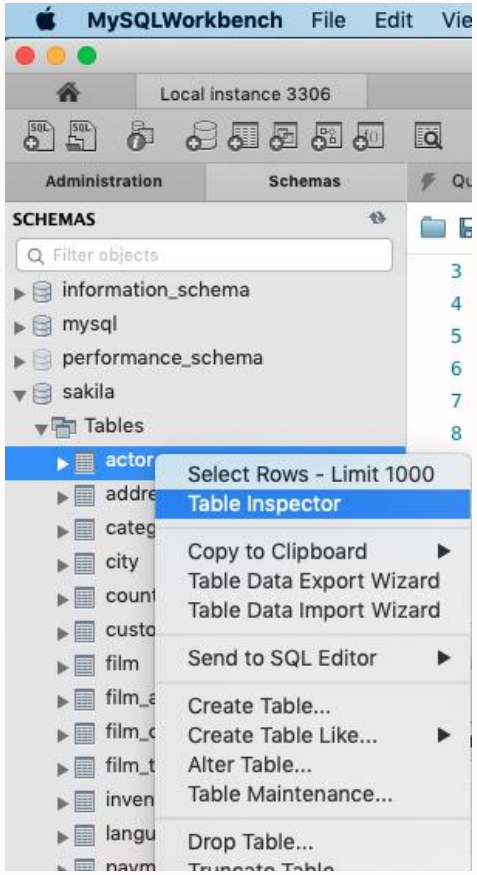

# MySQL Workbench Schema Inspector - Grants




Info | Tables | Columns | Indexes | Triggers | Views | Stored Procedures | Functions | Grants | Events

| Host | User | Scope | Select | Insert | Update | Delete | Create | Drop | Grant | Refere... | Index | Alter | Create... | Lock Ta... | Create... | Create... | Alter R... | Execut |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| % | root2 | <global> | Y | Y | Y | Y | Y | Y | Y | Y | Y | Y | Y | Y | Y | Y | Y | Y |
| localhost | mysql.infosch... | <global> | Y | N | N | N | N | N | N | N | N | N | N | N | N | N | N | N |
| localhost | root | <global> | Y | Y | Y | Y | Y | Y | Y | Y | Y | Y | Y | Y | Y | Y | Y | Y |
| % | app_read | sakila | Y | N | N | N | N | N | N | N | N | N | N | N | N | N | N | N |
| % | app_write | sakila | N | Y | Y | Y | N | N | N | N | N | N | N | N | N | N | N | N |
| localhost | rw_norole_user | sakila | Y | Y | Y | Y | N | N | N | N | N | N | N | N | N | N | N | N |

# MySQL Workbench Table Inspector - Grants




Info | Columns | Indexes | Triggers | Foreign keys | Partitions | Grants | DDL

**Table privileges**

| User | Host | Scope | Select | Insert | Update | Delete | Create | Drop | Grant | Refere... | Index | Alter | Create... | Show v... | Trigger |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| root2 | % | <global> | Y | Y | Y | Y | Y | Y | Y | Y | Y | Y | Y | Y | Y |
| mysql.infosch... | localhost | <global> | Y | N | N | N | N | N | N | N | N | N | N | N | N |
| root | localhost | <global> | Y | Y | Y | Y | Y | Y | Y | Y | Y | Y | Y | Y | Y |
| app_read | % | sakila | Y | N | N | N | N | N | N | N | N | N | N | N | N |
| app_write | % | sakila | N | Y | Y | Y | N | N | N | N | N | N | N | N | N |
| rw_norole_user | localhost | sakila | Y | Y | Y | Y | N | N | N | N | N | N | N | N | N |

# 권한 조회– Direct Grants

```
WITH
tableprivs AS (SELECT user, host, 'mysql.tables_priv' as PRIV_SOURCE , DB as _db, Table_Name as _obj
, ' ' as _col
FROM mysql.tables_priv where Table_name like '%' ),
colprivs AS (SELECT User, Host, 'mysql.columns_priv' as PRIV_SOURCE , DB as _db, table_name as _obj
, column_name as _col
FROM mysql.columns_priv WHERE Table_name like '%' )
SELECT user,host, PRIV_SOURCE , _db as _db, _obj, _col FROM
( SELECT user,host, PRIV_SOURCE, _db, _obj, _col FROM colprivs UNION
SELECT user,host, PRIV_SOURCE, _db, _obj, _col FROM tableprivs) as tt group by user, host,
PRIV_SOURCE, _db, _obj, _col;
```

| user | host | PRIV_SOURCE | _db | _obj | _col |
|---|---|---|---|---|---|
| newuserw | % | mysql.columns_priv | mysql | plugin | name |
| mysql.session | localhost | mysql.tables_priv | mysql | user | |
| newuserq | % | mysql.tables_priv | mysql | plugin | |
| newuserw | % | mysql.tables_priv | mysql | plugin | |
| betsy | localhost | mysql.tables_priv | sakila | actor | |
| mysql.sys | localhost | mysql.tables_priv | sys | sys_config | |

# 어떤 users / roles 들이 actor 에 대한 접근 권한을 가지고 있는지?

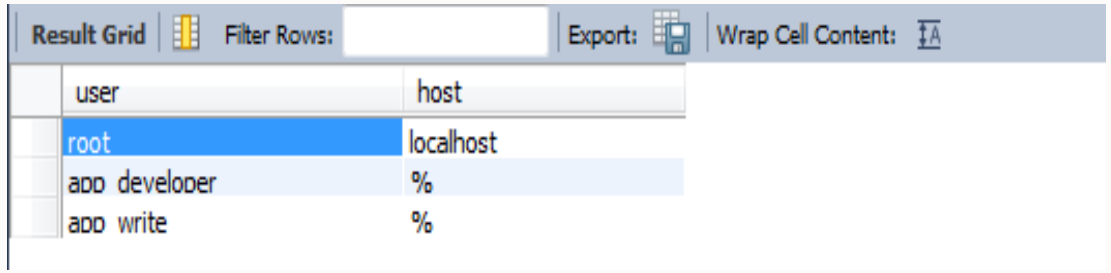
```
use mysql;
WITH
globalprivs AS (SELECT user,host FROM mysql.user WHERE 'Y' IN
    (Select_priv, Insert_priv, Update_priv, Delete_priv, Create_priv,
    Drop_priv, Reload_priv, Shutdown_priv, Process_priv, File_priv,
    Grant_priv, References_priv, Index_priv, Alter_priv, Show_db_priv,
    Super_priv, Create_tmp_table_priv, Lock_tables_priv, Execute_priv,
    Repl_slave_priv, Repl_client_priv, Create_view_priv, Show_view_priv,
    Create_routine_priv, Alter_routine_priv, Create_user_priv,
    Event_priv, Trigger_priv, Create_tablespace_priv, Create_role_priv,
    Drop_role_priv)
  ),
  dbprivs AS (SELECT user,host FROM mysql.db WHERE 'Y' IN
   (Select_priv, Insert_priv, Update_priv, Delete_priv, Create_priv, Drop_priv,
    Grant_priv, References_priv, Index_priv, Alter_priv, Create_tmp_table_priv,
    Lock_tables_priv, Create_view_priv, Show_view_priv, Create_routine_priv,
    Alter_routine_priv, Execute_priv, Event_priv, Trigger_priv)
  ),
  tableprivs AS (SELECT user, host FROM tables_priv WHERE Table_name='actor' ),
  colprivs AS (SELECT User, Host FROM mysql.columns_priv WHERE Table_name='actor' )
SELECT user,host FROM (SELECT user,host FROM globalprivs UNION
SELECT user,host FROM dbprivs UNION
SELECT user,host FROM colprivs UNION
SELECT user,host FROM tableprivs) as tt group by user, host;
```

| user | host |
|---|---|
| root2 | % |
| mysql.infoschema | localhost |
| mysql.session | localhost |
| root | localhost |
| app_read | % |
| app_write | % |
| ▶ rw_norole_user | localhost |
| mysql.sys | localhost |
| betsy | localhost |

Note:
There are various mysql.* users
used by internal components
mysql.informationschema
mysql.session, mysql.sys

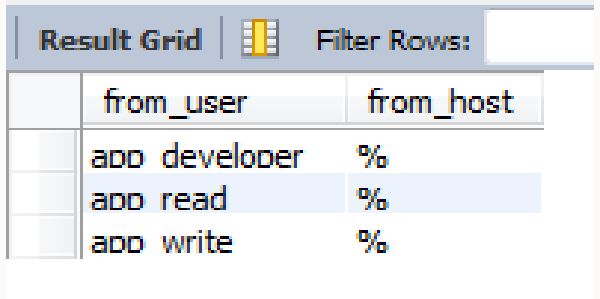
# actor 테이블에 대한 select 권한이 있는 사용자

```
WITH
  globalprivs AS (SELECT user,host FROM mysql.user WHERE
    Select_priv = 'Y'),
  dbprivs AS (SELECT user,host FROM mysql.db WHERE
    Select_priv = 'Y'),
  colprivs AS (SELECT user, host FROM mysql.columns_priv WHERE
Table_name='actor' AND FIND_IN_SET('Select',Column_priv)),
  tableprivs AS (SELECT User, Host FROM mysql.tables_priv WHERE
Table_name='actor' AND FIND_IN_SET('Select',Table_priv))
SELECT user,host FROM (SELECT user,host FROM globalprivs UNION
SELECT user,host FROM dbprivs UNION
SELECT user,host FROM colprivs UNION
SELECT user,host FROM tableprivs) as tt group by user, host;
```



# actor 테이블에 대해서 데이터를 변경할 수 있는 roles 조회

```
WITH
  globalprivs AS (SELECT user,host FROM mysql.user WHERE 'Y' IN
    (Insert_priv, Update_priv, Delete_priv, Drop_priv, Alter_priv)),
  dbprivs AS (SELECT user,host FROM mysql.db WHERE 'Y' IN
   (Insert_priv, Update_priv, Delete_priv, Drop_priv, Alter_priv)),
  tableprivs AS (SELECT user, host FROM tables_priv WHERE
table_name='actor'),
  colprivs AS (SELECT User, Host FROM mysql.columns_priv WHERE
table_name='actor')
SELECT from_user,from_host FROM (SELECT user,host FROM globalprivs UNION
SELECT user,host FROM dbprivs UNION
SELECT user,host FROM colprivs UNION
SELECT user,host FROM tableprivs) as tt
RIGHT JOIN
mysql.role_edges as tr
ON tr.to_user=tt.user AND tr.to_host= tt.host GROUP BY from_user, from_host;
```



# administrative/global 권한을 가진 사용자

```
SELECT user,host, 'Global Priv', Select_priv, Insert_priv, Update_priv, Delete_priv, Create_priv,
Drop_priv, Reload_priv, Shutdown_priv, Process_priv, File_priv,
Grant_priv, References_priv, Index_priv, Alter_priv, Show_db_priv,
Super_priv, Create_tmp_table_priv, Lock_tables_priv, Execute_priv,
Repl_slave_priv, Repl_client_priv, Create_view_priv, Show_view_priv,
Create_routine_priv, Alter_routine_priv, Create_user_priv,
Event_priv, Trigger_priv, Create_tablespace_priv, Create_role_priv,
Drop_role_priv FROM mysql.user
WHERE ( 'Y' IN
(Select_priv, Insert_priv, Update_priv, Delete_priv, Create_priv,
Drop_priv, Reload_priv, Shutdown_priv, Process_priv, File_priv,
Grant_priv, References_priv, Index_priv, Alter_priv, Show_db_priv,
Super_priv, Create_tmp_table_priv, Lock_tables_priv, Execute_priv,
Repl_slave_priv, Repl_client_priv, Create_view_priv, Show_view_priv,
Create_routine_priv, Alter_routine_priv, Create_user_priv,
Event_priv, Trigger_priv, Create_tablespace_priv, Create_role_priv,
Drop_role_priv)) and (user.user not like 'mysql.%');
```

| user | host | Global Priv | Select_priv | Insert_priv | Update_priv | Delete_priv | Create_priv | Drop_priv | Reload_priv | Shutdown_priv |
|---|---|---|---|---|---|---|---|---|---|---|
| root2 | % | Global Priv | Y | Y | Y | Y | Y | Y | Y | Y |
| root | localhost | Global Priv | Y | Y | Y | Y | Y | Y | Y | Y |

# MySQL 연결 제어

Dealing with Failed Login Attempts related to Brute Force Attacks

## 설치 방법:

```
INSTALL PLUGIN CONNECTION_CONTROL SONAME 'connection_control.so';
INSTALL PLUGIN CONNECTION_CONTROL_FAILED_LOGIN_ATTEMPTS SONAME
'connection_control.so’;
```

## 설정 확인:

```
select @@connection_control_failed_connections_threshold,
@@connection_control_min_connection_delay,
@@connection_control_max_connection_delay;


SET PERSIST connection_control_failed_connections_threshold = 4;
SET PERSIST connection_control_min_connection_delay = 1500;
```




# 계정 별 연결 제어

Configure Account Resource Limits

```
mysql> CREATE USER 'mmarx'@'::1' IDENTIFIED BY 'P@SS'
    ->     WITH MAX_QUERIES_PER_HOUR 20
    ->          MAX_UPDATES_PER_HOUR 10
    ->          MAX_CONNECTIONS_PER_HOUR 5
    ->          MAX_USER_CONNECTIONS 2;
```

https://dev.mysql.com/doc/mysql-security-excerpt/8.0/en/connection-control-installation.html

https://mysqlserverteam.com/the-connection_control-plugin-keeping-brute-force-attack-in-check/




# Side Bar –

Use SET PERSIST

- MySQL 8.0에서 DBA는 시스템 변수를 SQL 문으로 변경 가능
- 변경된 값은 mysqld-auto.cnf 기록
- SET PERSIST ONLY – mysqld-auto.cnf 만 기록하고 runtime 값은 변경 하지 않음
  - Read-only 변수 변경할 경우, 다음 재시작후 적용

- 일부 변수는 지원하지 않음
  - https://dev.mysql.com/doc/refman/8.0/en/nonpersistible-system-variables.html


```
{
  "Version": 1,
  "mysql_server": {
    "require_secure_transport": {
      "Value": "ON",
      "Metadata": {
        "Timestamp": 1564600430679850,
        "User": "root",
        "Host": "localhost"
      }
    },
    "validate_password.dictionary_file": {
      "Value": "<FILENAME OF DICTIONARY FILE",
      "Metadata": {
        "Timestamp": 1564598898444506,
        "User": "root",
        "Host": "localhost"
      }
    },
    "authentication_ldap_sasl_server_host": {
      "Value": "127.0.0.1",
      "Metadata": {
        "Timestamp": 1564695043687370,
        "User": "root",
        "Host": "localhost"
      }
    },
    "authentication_ldap_sasl_bind_base_dn": {
```

# Encryption

Network & data & files




# MySQL Encryption

SSL/TLS 암호화

- MySQL clients &  Server 사이
- Replication Source & Replica 사이
- 최신 OpenSSL 3.0 및 FIPS Object Model 3.0
- X509 표준 기반 인증
- CA인증서 지원
- SET PERSIST require_secure_transport=ON;

데이터 암호화

- AES Encrypt/Decrypt

**MySQL Enterprise TDE**

- Transparent Data Encryption
- 로그 암호화
- 키 관리

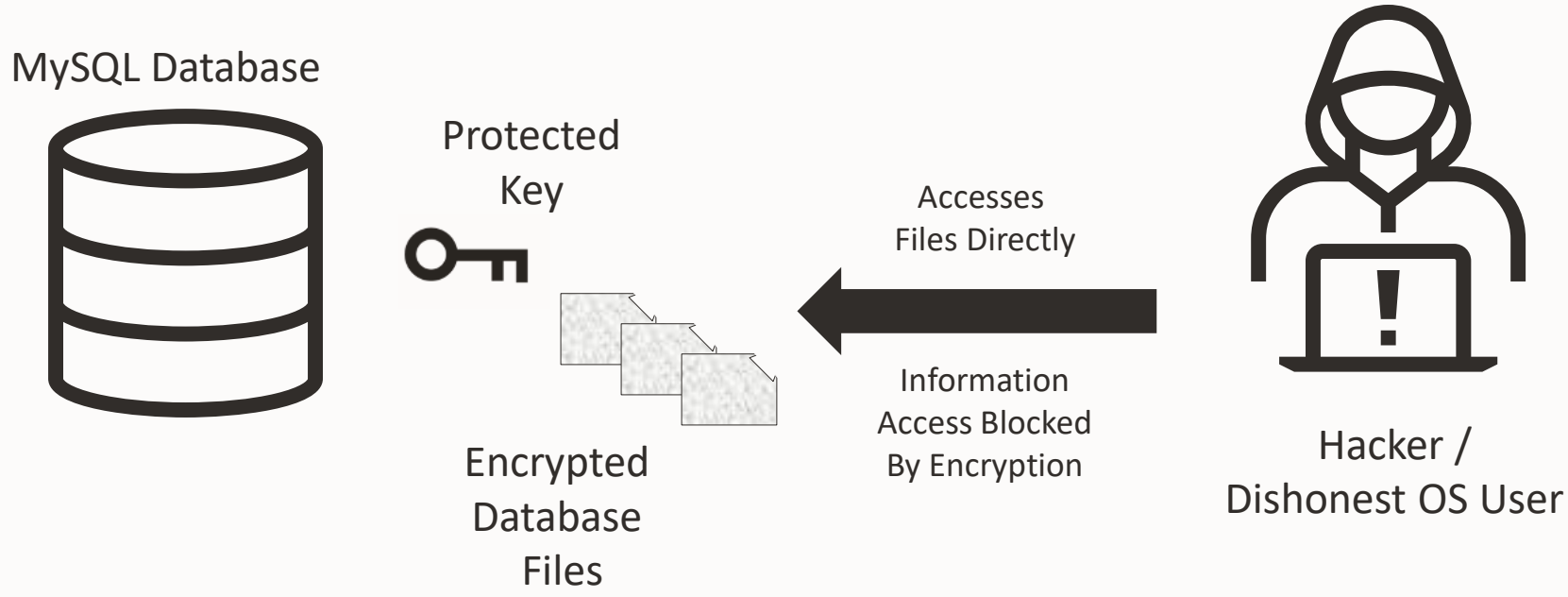
**MySQL Enterprise Encryption**

- 비대칭 Encrypt/Decrypt
- Public Key & Private Keys 생성
- Session Keys
- Digital Signatures

**MySQL Enterprise Backup**

- AES Encrypt/Decrypt
- Key Backup and Recovery

# Attack on Files



# Transparent Data Encryption

CREATE TABLE 옵션

`ENCRYPTION="Y"`

시스템 변수

`default_table_encryption`

`table_encryption_privilege_check`

Keyring plugin

- `keyring_file|okv|aws|hashcorp|oci`

```
[mysqld]
early-plugin-load=keyring_file.so
```

Component(8.0.24~)

- `component_keyring_file`
- `component_keyring_encrypted_file`
- `component_keyring_oci`
- `plugin_dir` 및 `manifest` 으로 설정

InnoDB에서만 지원

- 암호화 테이블 지원
- IMPORT/EXPORT 지원
- master key rotation 기능 지원

```
ALTER INSTANCE ROTATE INNODB
MASTER KEY
```

# Key Managers

OASIS KMIP protocol 기반 구현:

- Oracle Key Vault
- Gemalto KeySecure
- Thales Vormetric Key Management Server
- Fornetix Key Orchestration
- Townsend Alliance Key Manager
- Entrust KeyControl

HTTPS 기반 APIs 지원:

- Oracle Cloud Infrastructure Vault
- Hashicorp Vault
- AWS KMS

# key는 안전할 가요? keyring 설치 되어 있나요? Key 관리?

```
SELECT `PLUGIN_NAME`, `PLUGIN_STATUS`, `PLUGIN_TYPE`, `PLUGIN_LIBRARY`,
`PLUGIN_DESCRIPTION`, `LOAD_OPTION`
FROM `information_schema`.`PLUGINS` where PLUGIN_NAME LIKE  'keyring_file' and
plugin_status='ACTIVE';
```

NOTE: keyring_file – is not for production. (Dev/QA only – its in a Plain text file)

KMIP, Encrypted Keyring, OCI Vault, Hashicorp, AWS KMS, etc. should be used in production.

| PLUGIN_NAME | PLUGIN_... | PLUGIN_TYPE | PLUGIN_LIBRARY | PLUGIN_DESCRIPTION | LOAD_OPTION |
|---|---|---|---|---|---|
| ▶ keyring_file | ACTIVE | KEYRING | keyring_file.so | store/fetch authentication data to/from a flat file | ON |

NOTE: Keyring installation is key manager specific.  See
https://dev.mysql.com/doc/refman/8.0/en/keyring.html

# InnoDB REDO, UNDO, Binary, Audit 로그 암호화 되었나요?

```
SELECT VARIABLE_NAME, VARIABLE_VALUE, 'innodb_redo_log AT REST ENCRYPTION' as Note,
IF(VARIABLE_VALUE = 'ON', 'PASS', 'FAIL')  as CHECK_VAL
FROM performance_schema.global_variables where variable_name =
'innodb_redo_log_encrypt';
-
SELECT VARIABLE_NAME, VARIABLE_VALUE, 'innodb_undo_log AT REST ENCRYPTION' as Note,
IF(VARIABLE_VALUE = 'ON', 'PASS', 'FAIL')  as CHECK_VAL
FROM performance_schema.global_variables where variable_name =
'innodb_undo_log_encrypt';
-
SELECT VARIABLE_NAME, VARIABLE_VALUE, 'BINLOG - AT REST ENCRYPTION' as Note,
IF(VARIABLE_VALUE = 'ON', 'PASS', 'FAIL')  as CHECK_VAL
FROM performance_schema.global_variables where variable_name = 'binlog_encryption';
-
SELECT VARIABLE_NAME, VARIABLE_VALUE, 'AUDIT LOG - AT REST ENCRYPTION' as Note,
IF(VARIABLE_VALUE = 'AES', 'PASS', 'FAIL')
FROM performance_schema.global_variables where variable_name = 'audit_log_encryption';
```




# MySQL Enterprise Encryption

MySQL 암호화 함수

- 대칭 암호화 AES256
- Public-key / asymmetric cryptography – RSA

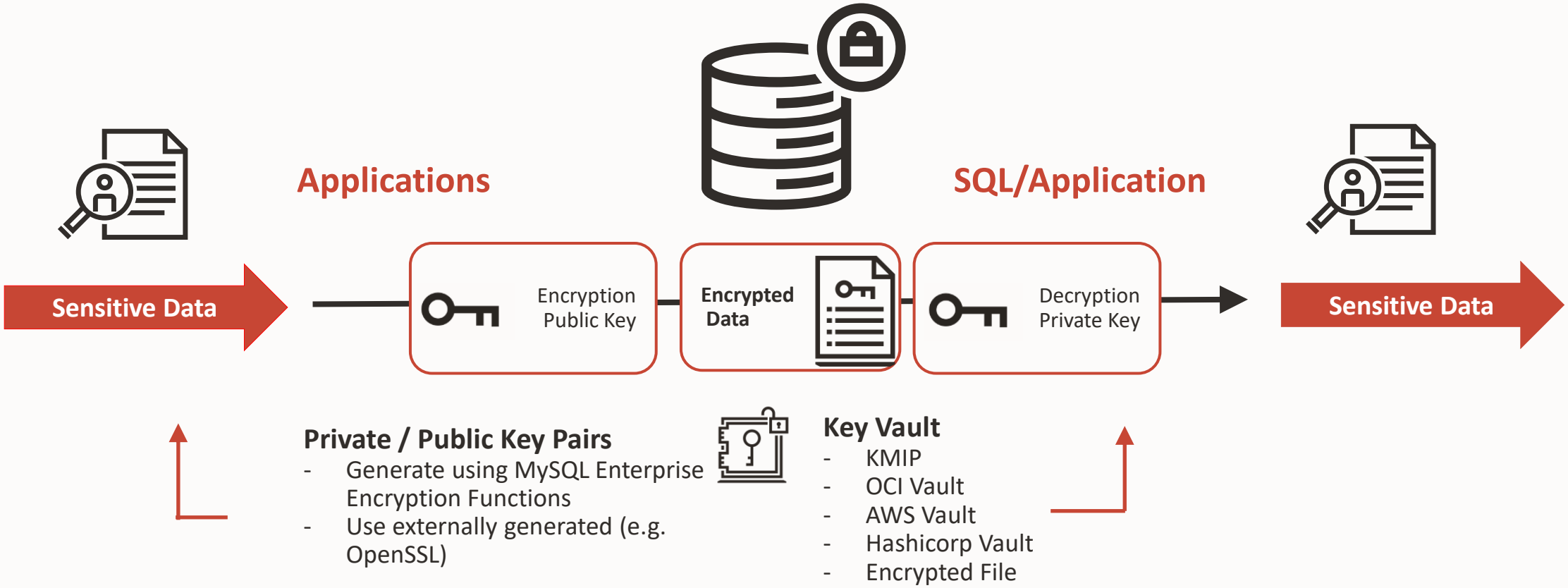
Key 관리 함수

- public 및 private keys 생성
- Key 변환 함수: DH

사인 및 데이터 인증 함수

- Cryptographic hashing for digital signing, verification, & validation – RSA,DSA

# MySQL Enterprise Encryption
## Encryption/Decryption within MySQL



# Audit, Firewall, Masking




# MySQL Enterprise Audit: Complete Audit Data

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>
```

# 세분화된 감사 정책 및 필터링

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

# 감사를 사용하고 있나요?

확인 방법:

```
SELECT `PLUGIN_NAME`, `PLUGIN_STATUS`, `PLUGIN_TYPE`, `PLUGIN_LIBRARY`,
`PLUGIN_DESCRIPTION`, `LOAD_OPTION`  FROM `information_schema`.`PLUGINS` where
PLUGIN_NAME LIKE 'audit_log' and plugin_status='ACTIVE';
```

설치 방법: share디렉토리

```
# shell> mysql -u root -p < audit_log_filter_linux_install.sql;
```

```
# Edit the mysql config file my.cnf (or my.ini on windows)
audit-log = ON|FORCE|FORCE_PLUS_PERMANENT
```

디폴트는 로깅하지 않음

```
SELECT audit_log_filter_set_filter('log_all', '{ "filter": { "log": true } }');
SELECT audit_log_filter_set_user('%', 'log_all');
```



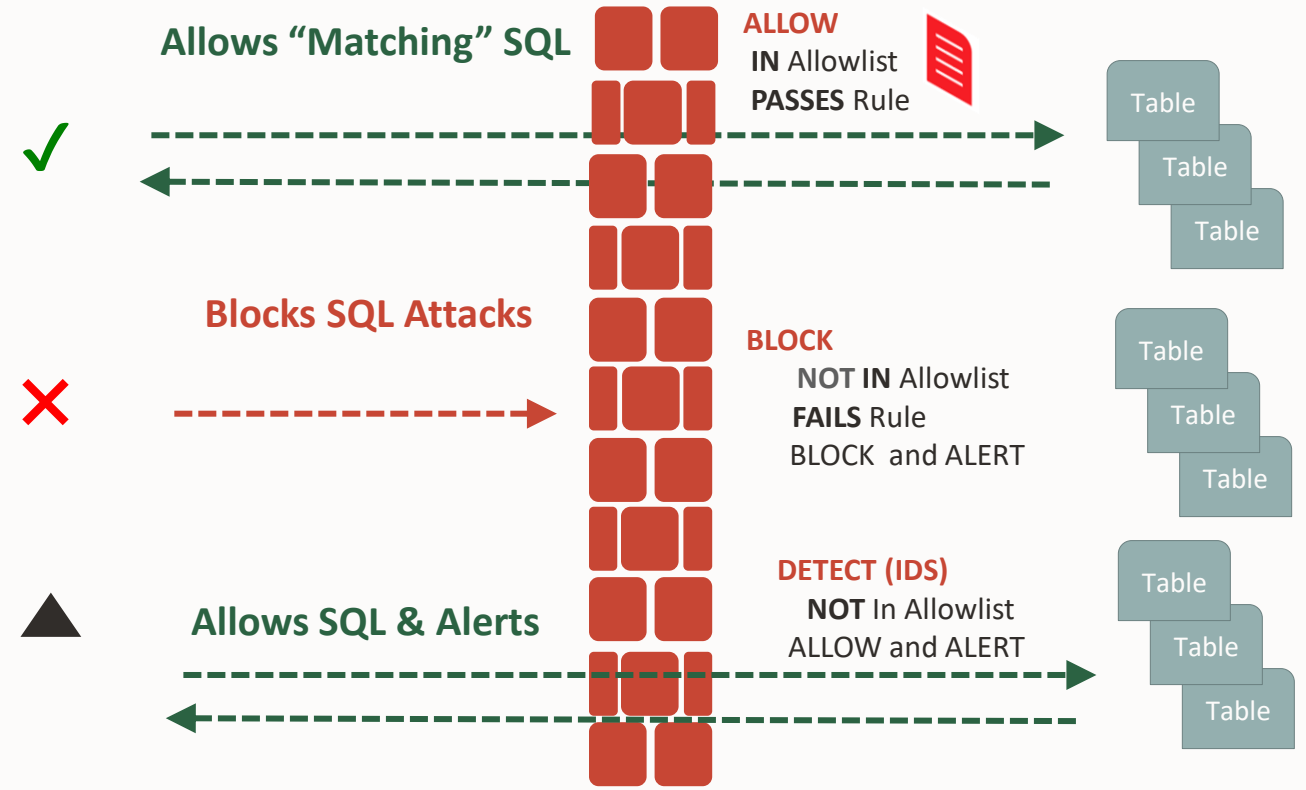

# MySQL Enterprise Firewall: Operating Modes

## 허용 리스트 자동 생성 – RECORDING

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



# MySQL Enterprise Firewall Details

유저 레벨에서 사용, 유저 별 상태 지정

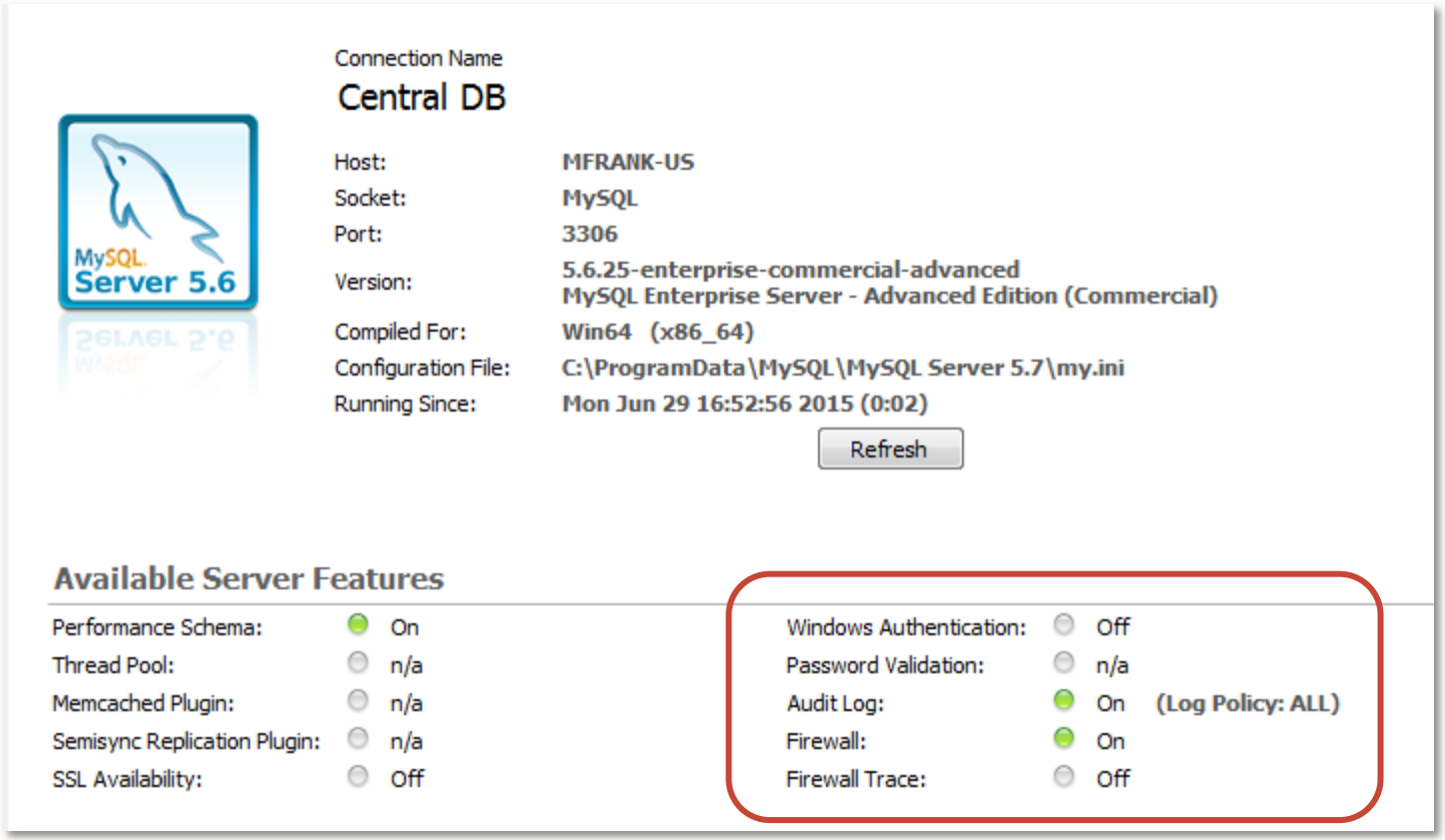
- RECORDING
- PROTECTING
- DETECTING
- OFF

```
call mysql.set_firewall_mode ('fwuser@localhost', 'RECORDING');

call mysql.set_firewall_mode ('fwuser@localhost', 'PROTECTING');

call mysql.set_firewall_mode ('fwuser@localhost', 'DETECTING');

call mysql.set_firewall_mode ('fwuser@localhost', 'OFF');
```

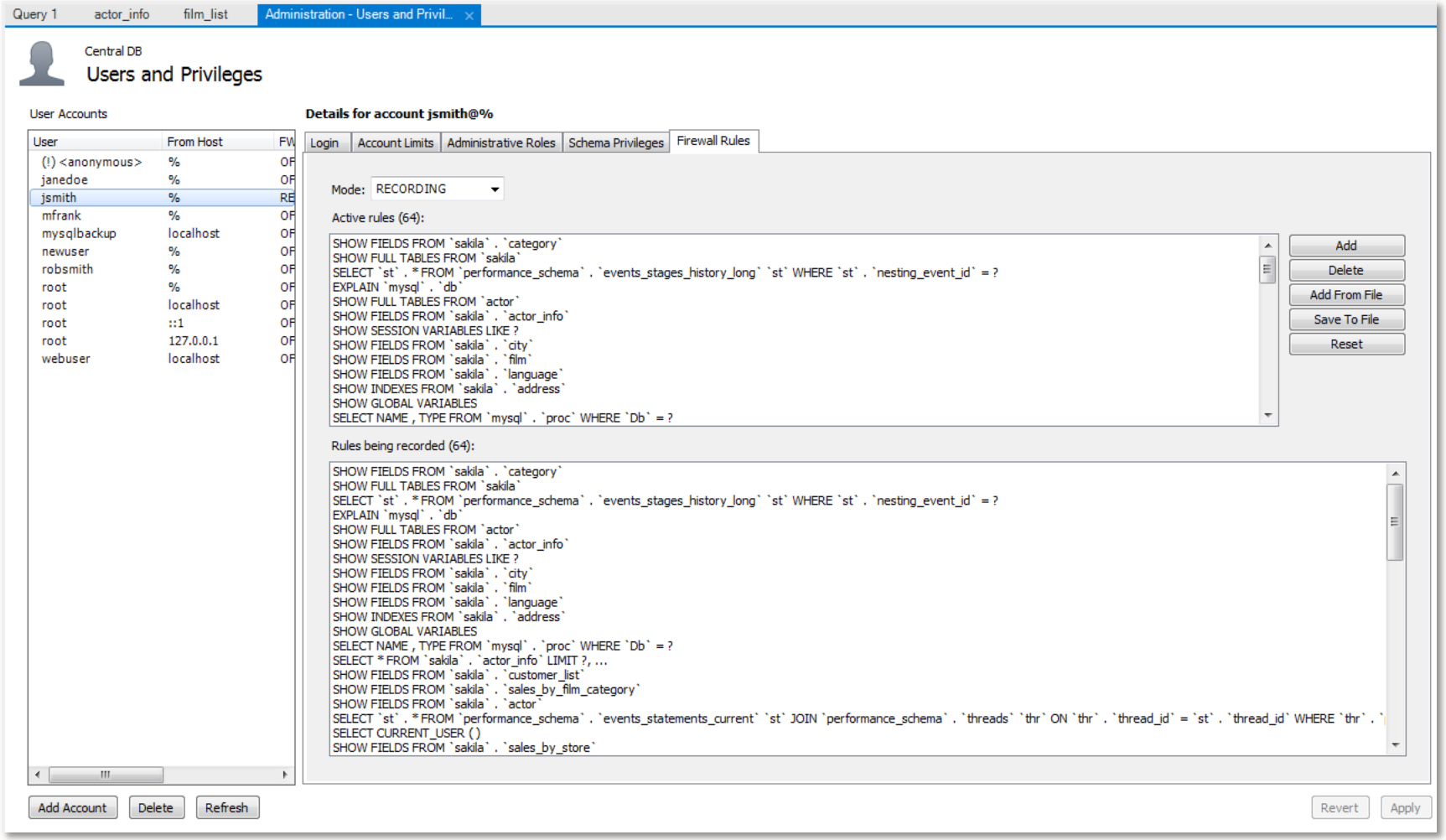
유저 그룹 별로 적용

- `CALL mysql.sp_set_firewall_group_mode_and_user('drupal_group', 'RECORDING', 'drupal_user@localhost');`
- `CALL mysql.sp_firewall_group_enlist('drupal_group', 'drupal_user@localhost');`
- `CALL mysql.sp_set_firewall_group_mode('drupal_group', 'DETECTING');`

# MySQL Workbench: Firewall Status



# MySQL Enterprise Firewall: Per User Allowlists



# MySQL Enterprise Masking and De-Identification

De-identify, Anonymize Sensitive Data

" 데이터 마스킹 (Data Masking)은 실제 값을 대체 값으로 바꾸어 민감한 정보를 숨기는 방법이다."

**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 Firewall Details

설치 및 사용 방법:

```
INSTALL PLUGIN data_masking SONAME 'data_masking.so’;
CREATE FUNCTION gen_range RETURNS INTEGER SONAME 'data_masking.so';
CREATE FUNCTION gen_rnd_email RETURNS STRING SONAME 'data_masking.so';
CREATE FUNCTION gen_rnd_us_phone RETURNS STRING SONAME 'data_masking.so';
CREATE FUNCTION mask_inner RETURNS STRING SONAME 'data_masking.so';
CREATE FUNCTION mask_outer RETURNS STRING SONAME 'data_masking.so’;
SELECT mask_inner(NAME, 1,1) FROM world.city limit 10;
SELECT mask_outer(NAME, 1,1) FROM world.city limit 10;
SELECT gen_range(1, 200);
SELECT gen_rnd_email();
```




# MySQL Enterprise Edition

MySQL Enterprise Authentication

- External Authentication Modules
  - LDAP, Native Kerberos, Microsoft AD

MySQL Enterprise Encryption

- Public/Private Key Cryptography
- Asymmetric Encryption
- Digital Signatures, Data Validation

MySQL Enterprise Firewall

- Block SQL Injection Attacks
- Intrusion Detection

MySQL Enterprise Audit

- User Activity Auditing, Regulatory Compliance

MySQL Enterprise TDE

- AES 256 encryption
- Key Management

MySQL Enterprise Masking

- Data Masking and Obfuscation
- Formatted Data Randomization
- Pseudonymization, Data blocklists

MySQL Enterprise Monitor

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

MySQL Enterprise Backup

- Securing Backups, AES 256 encryption

# MySQL Database Hardening

## Installation

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

## Configuration

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

## User Management

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

## Passwords

- Strong Password Policy
- Hashing, Expiration
- Password Validation Plugin

## Encryption

- SSL/TLS for Secure Connections
- Data Encryption (AES, RSA)
- TDE

## Backups

- Monitor Backups
- Encrypt Backups

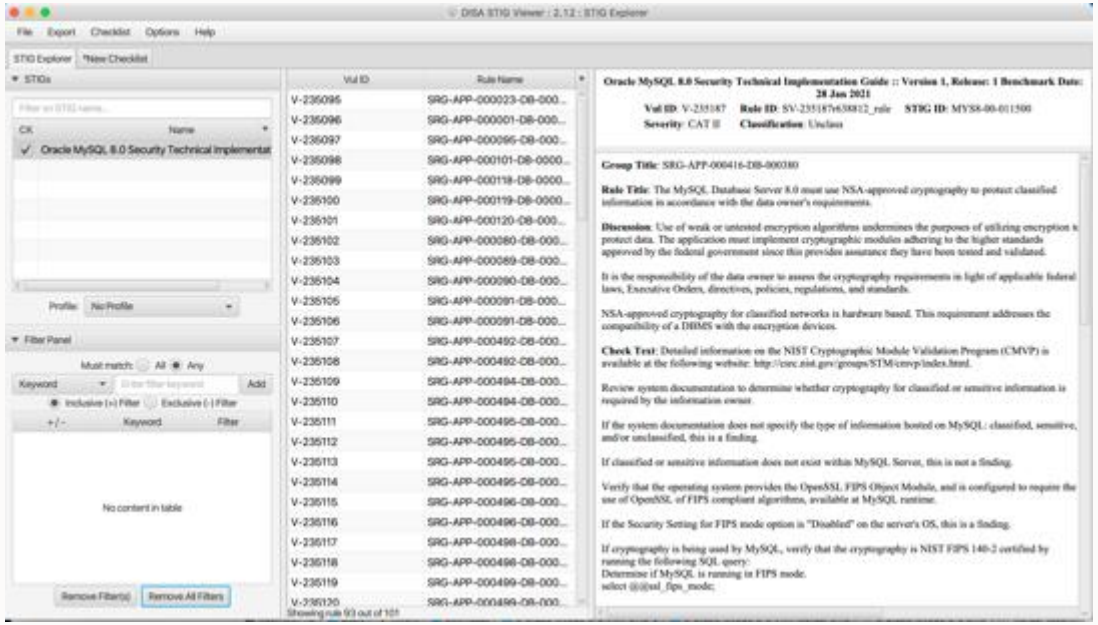
# Security Guidelines

SECURITY TECHNICAL IMPLEMENTATION GUIDES (STIGS)

- DISA STIG for MySQL 8.0 EE

https://www.mysql.com/products/enterprise/stig.html

https://public.cyber.mil/stigs/



# Security Guidelines

CIS Benchmark for MySQL 8.0 EE

- https://www.cisecurity.org/benchmark/oracle_mysql/

- CIS 권고안은 DoD 클라우드 컴퓨팅 보안 권고 가이드 (SRG), 지불 카드 산업 데이터 보안 표준 (PCI DSS), 건강 보험 이동성 및 책임법 (HIPAA), 연방 정보 보안 관리법 (FISMA), 연방 위험 및 인가 관리 프로그램 (FedRAMP) 및 미국 국립 표준 기술 연구소 (NIST)에 의해 보안 구성 표준으로 인정됩니다.

감사합니다!