

The Okta logo is displayed in a bold, white, lowercase sans-serif font against a blue background.

MySQL Data Security Risk Assessment

Ronald Bradford

Principal Database Reliability Engineer

June 2018

Database+Operations Conference

Barcelona, Spain

21-22 June 2018

<https://dataops.barcelona/>

Table of contents

MySQL Data Security Risk Assessment

- 1 Why?
- 2 How?
- 3 Recommendations
- 4 Implementation Challenge
- 5 Ongoing Risk



The background is a solid blue color with two large, overlapping circles on the right side. The circles are in different shades of blue, creating a layered effect. The text is positioned on the left side of the image.

About Okta/Speaker

About speaker – Ronald Bradford

- Principal Database Reliability Engineer (DBRE) at Okta
- 29 years of RDBMS experience
- 19 years of MySQL experience

- Speaker

<http://ronaldbradford.com/presentations/>

- Author

<http://effectivemysql.com/>



About Okta

- Leading provider of identity for the enterprise
- Connects and protects employees of many of the world's largest enterprises
- Securely connects enterprises to their partners, suppliers and customers
- Okta helps customers fulfill their missions faster by making it safe and easy to use the technologies they need to do their most significant work



Why?



Why data security risk assessment is important?

- Humans seek convenience over complexity
- Humans prey on other humans
- Humans are better at recognition than programmed solutions



Reference example (3 years)

- Are any of your password 3 years old?
- Have any employees left in the past 3 years?
- Were any password stored in clear-text?



What is happening now on your database?

- Hard failures
 - Invalid access to any data-store (e.g. Invalid password)
 - How frequent/often?
- Soft attacks
 - `SELECT email from customers;`



Open source failures

- Generational bad habits (e.g. defaults)
 - No default administrator password
 - No password strength
 - Open ports
 - Poor ACLs examples
- Continued bad habits
 - NoSQL products
 - Docker



IRL comparison

- Physical Security
 - Badge+Photo+Scan+Security Guard
 - Pinpad+Timed Access
 - Metal Detectors
 - Secondary Scan
- Monitoring
 - Security Cameras+Recording+Image Recognition
- Human Intelligence
 - Random Security Guard Checks
 - Peers



How?



Password-less authentication

- First Access
 - Computer Login (Password)
 - VPN (Password+Token/MFA)
 - Company Systems (Password+MFA/Token)
 - Other (Firewall, bastion, ssh)
- Then
 - `$ ssh <any-db-server>`
 - `$ mysql -e "ANY COMMAND I LIKE"`



No MySQL password necessary (sudo)

- OS 'root' access
 - `$ ssh dba@server`
 - `$ sudo su -`
- Compromises
 - `$ service mysql restart --skip-grant-tables`
 - `$ strings /var/lib/mysql/mysql/user.MYD`

 - `mysql> create user demo@localhost identified by 'SomeLongP155wd#';`
 - `strings /ebs/var/lib/mysql/mysql/user.MYD | grep demo`
 - `demo*294B43D3206B0B0A1670A2E606F1D5B9655906B7`



Password use

- Lack of strength
- Lack of rotation
- Clear-text
 - my.cnf
 - master.info
 - Third party tools
 - /etc/percona-toolkit/percona-toolkit.conf
 - Process space
 - Command line
- Weaker encryption methods (e.g. SHA1 v SHA256+SALT)



MySQL privileges

- The GRANT ALL problem (i.e. SUPER, ALTER and everything else)
- The *.* problem (i.e. not schema.table)
- The % or 10.% problem (i.e. not host but network)
- The DEFINER / INVOKER stored function problem
- The mysql.user problem
- The read-only problem



NoSQL and no security

- MongoDB
- Cassandra
- Redis
- Elasticsearch
- <insert other products here>

<https://www.slideshare.net/wurbanski/nosql-no-security>

<https://speakerdeck.com/xeraa/nosql-means-no-security>



Recommendations



Practical policies and actions

1. Purpose driven credentials (*)
2. Least privileged model (*)
3. Segregation of responsibility
4. Environment boundaries (*)
5. No clear-text passwords (*)
6. Longer & stronger passwords
7. Password rotation
8. Sha256 password with salt (*)
9. Remove snowflakes
10. Timeouts
11. Timed access
12. Logging (*)
13. Auditing (*)
14. Human Factor Authentication (HFA) (*)
15. Release cadence (*)



Accounts with a purpose (1)

- Individually Named Accounts
 - By name
 - johnsmith
 - dba_jsmith
 - By Purpose
 - zabbix
 - splunk
 - pt
 - collectd
 - xtrabackup



Know your privileges (2)

- If an account requires SUPER, why?
 - Evaluate and reevaluate regularly (e.g. each quarter)
- e.g. Percona Toolkit
 - `GRANT ALL PRIVILEGES ON *.* to percona@localhost;`
 - You can alter a table with?
- pt-heartbeat requires
 - `GRANT REPLICATION CLIENT ON *.* TO percona@localhost`
 - `GRANT INSERT, DELETE ON heartbeat.heartbeat TO percona@localhost`
- pt-slave-delay requires
 - `GRANT SUPER ON *.* TO percona@localhost;`
 - Replaceable with native MySQL 5.6 delayed replication



Always separate environments (4)

- Is an password shared
 - Across test/stage/prod
- Do you have tools to validate passwords across environments?
- It's just a test environment is not an excuse



Removing clear-text passwords (5)

- `.my.cnf`
 - Clear-text
 - Can have any OS permissions
 - Can reside in any directory
 - Any MySQL version
- `.mylogin.cnf`
 - Not clear-text
 - Restricted file privileges
 - Locked to a specific OS user
 - MySQL 5.6+



A stronger password plugin (8)

- mysql_native_password
 - SHA1(SHA1()) (20 bytes)
- sha256_password plugin (5.6)
 - sha256 (32 bytes)
 - + salt
- caching_sha2_password (5.7)

https://mysqlserverteam.com/protecting-mysql-passwords-with-the-sha256_password-plugin/
<https://dev.mysql.com/doc/refman/5.7/en/sha256-pluggable-authentication.html>



Logging (12) / Auditing (13)

- Limiting accounts to exact SQL (i.e. Whitelisting)
 - Allowed
 - SHOW PROCESSLIST
 - SHOW SLAVE STATUS
 - SHOW MASTER STATUS
 - SHOW ENGINE INNODB STATUS
 - Allowed via SUPER
 - KILL
 - Not Allowed via SUPER
 - SET GLOBAL



Human Factor Authentication (HFA) (14)

- Requiring a human (or second human)
 - Very destructive operations
 - CHANGE MASTER TO
 - ALTER TABLE DROP PARTITION



Software releases (15)

- New releases provide new functionality
- Who is running MySQL 5.0?
- Who is running MySQL 5.5?
 - `sha256_password` (5.6)
 - `mysql_config_editor` (5.6)
 - SUPER granularity (8.0)



New available functionality (15)

- Stronger encryption plugins
- mysql_config_editor
- Password Expiry
- Password strength check
- Root default password
- Mysql client logging removed
- Start Slave password
- Default SSL connections
- Active/Inactive user accounts
- Roles
- Super granularity
- Password history



Implementation Challenges



Convergence is really hard

- CREATE USER
 - user @ host
- DROP USER

- GRANT privilege
- REVOKE privilege

- Individual accounts
- Environment accounts
- Organization accounts

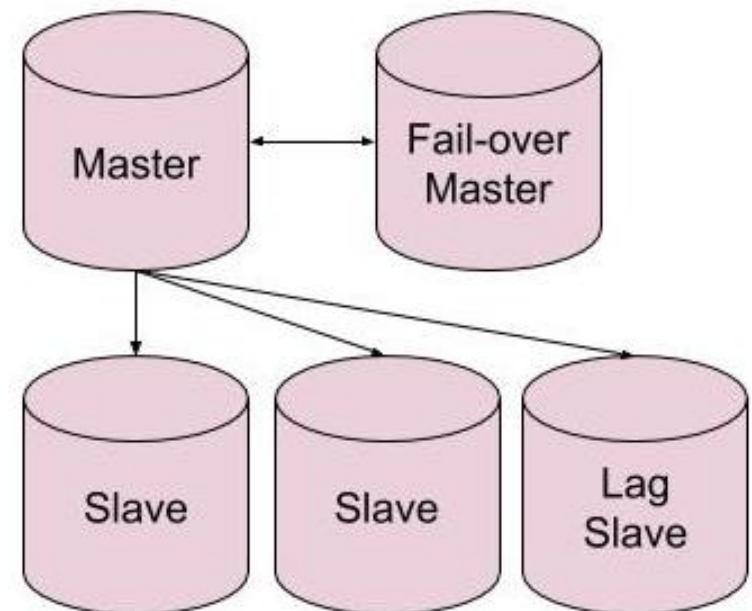
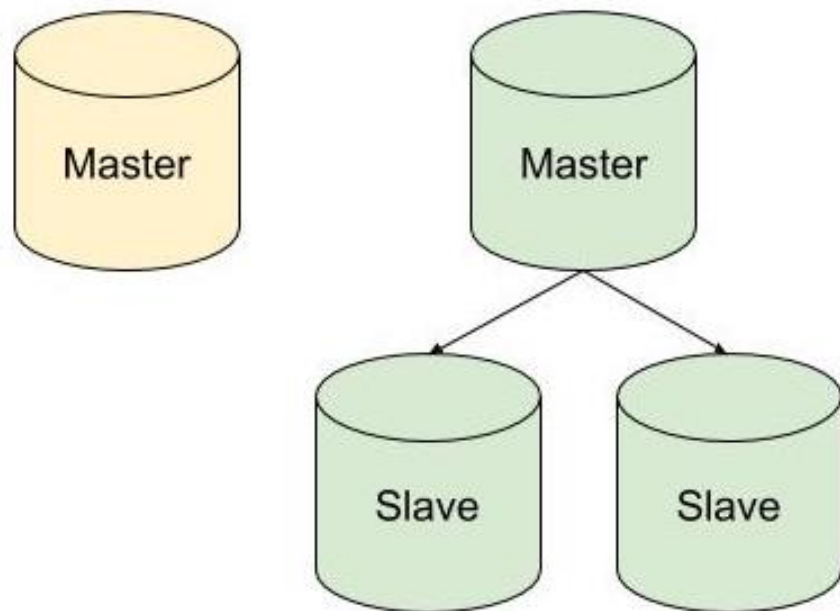


Changing passwords

- Single server – Simple
- Complex topology – Hard
- To replicate or not to replicate
 - Configuration management v replication
 - Are your replicas in read only mode?
 - Disabled configuration management
 - Lag slaves



Example Topologies



User convergence

- User account with multiple @hosts
- Different grants per @host
- User only on some servers
 - `DROP USER [IF EXISTS]` - MySQL 5.7



GRANT convergence example

- Monitor user (runs something every second)
- Has
 - `GRANT SELECT, PROCESS, SHOW DATABASES, SUPER, REPLICATION CLIENT ON *.*`
 - `GRANT SELECT, INSERT, UPDATE ON schema1.*`
 - `GRANT CREATE, INSERT ON mysql.*`
- Should have
 - `GRANT PROCESS, REPLICATION CLIENT ON *.*`



Revoking privileges

- `REVOKE ALL` works on a subset, but only per schema
 - `GRANT SELECT ON *.*`
 - `REVOKE ALL ON *.*`
- There is no `REVOKE [IF EXISTS]`
 - `REVOKE ALL ON *.*` does not fail when re-executed
 - `REVOKE ALL ON schema.*` does
- A user always has the `USAGE` privilege (can never have no schemas)
- `REVOKE`, `GRANT` are atomic statements
 - i.e. the time in-between
 - All or nothing does not apply (i.e. both work or both fail)



Tools

- External CMDB for users/grants
 - Yet another language or metadata
- Is `pt-show-grants` a CMDB option?
 - Password hash's not clear-text
 - But unknown
 - `GRANT not CREATE USER`



Guidelines

- Center for Information Security

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

- National Vulnerability Database
 - Common Vulnerabilities and Exposures (CVE)
- FedRAMP
- PCI
- Other compliance bodies



A stronger model example

- AWS RDS (not allowing SUPER)
 - `mysql> CALL mysql.rds_skip_repl_error;`
 - `mysql> CALL mysql.rds_kill(thread-id);`

<https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/Appendix.MySQL.CommonDBATasks.html>



MySQL wish list

- A user should be able to have a comment
 - Similar to CREATE TABLE
- ~~Be able to active/inactive an account – MySQL 5.7~~
- ~~Be able to expire a password – MySQL 5.7~~
- ~~SUPER granularity – MySQL 8.0~~
- SQL whitelist
- SQL blacklist
- REVOKE [ANY] PRIVILEGE



Conclusions



Data security not discussed

- Many other issues to consider in security scope
 - Encryption
 - Secure communication, e.g. SSL/ipsec
 - Backups
 - Log Files
 - Data integrity (read_only, sql_mode)



What can you do?

- Data security is not convenient
- Data security is not easy
- Data security is not a one off task

- **Be an advocate at your company**



The slide features a blue background with a radial light effect emanating from the center. Scattered across the background are several semi-transparent circular icons, each consisting of a dark blue outer ring and a lighter blue inner circle. The word "okta" is centered in a white, lowercase, sans-serif font.

okta

Thank You

