User & Security Management in MySQL

Comprehensive Overview of Authentication, Privileges, and Best Practices

Purpose of User & Security Management
Defines and enforces how users connect, authenticate, and interact securely with the MySQL database.

Core Focus Areas
Covers user creation, privilege assignment, SSL configuration, password policies, and administrative best practices.

Objective
Ensure data confidentiality, integrity, and availability through robust user and security configurations.

MySQL User Accounts and Authentication

Managing Access and Secure Logins

User Account Structure
Each account is defined by 'username'@'host'—the username for login and the client host allowed to connect. Example: 'john'@'localhost'.

User Creation
Accounts can be created for specific hosts, IP ranges, or globally using CREATE USER commands with appropriate IDENTIFIED BY clauses.

Authentication Methods
MySQL supports native password (mysql_native_password), SHA2 password (caching_sha2_password), or no-password options (not recommended).

Security Considerations
Restrict host access, enforce strong passwords, and avoid wildcard '%' for administrative users to prevent unauthorized logins.

MySQL Privilege System Overview

Understanding Access Control Hierarchy

Privilege Levels
MySQL privileges are structured hierarchically: Global (*.*), Database (db.*), Table (db.table), and Column (db.table.column). Higher levels inherit lower ones.

Privilege Types
Includes administrative privileges (SUPER, PROCESS), database-level privileges (CREATE, DROP, ALTER), and data-level privileges (SELECT, INSERT, UPDATE).

Granting Privileges
Use GRANT statements to assign access. Example: GRANT ALL PRIVILEGES ON database_name.* TO 'user'@'host';

Revoking Privileges
Access can be removed via REVOKE statements to ensure proper control when user roles change.

Granting and Revoking Privileges

Controlling User Permissions in MySQL

Granting Access
Privileges are assigned using GRANT statements, specifying database scope and user. Example: GRANT SELECT, INSERT ON db_name.* TO 'user'@'host';

Grant Option
WITH GRANT OPTION allows a user to pass privileges to others — useful for delegated administration but risky if misused.

Revoking Privileges
Use REVOKE statements to remove specific or all privileges, ensuring least-privilege enforcement. Example: REVOKE ALL PRIVILEGES ON db_name.* FROM 'user'@'host';

Audit and Verification
Administrators can inspect privileges with SHOW GRANTS FOR 'user'@'host'; to verify configurations and compliance.

User Management Operations

Creating, Modifying, and Removing MySQL Users

Renaming and Modifying Accounts
Users can be renamed or their host changed using RENAME USER. Passwords are updated with ALTER USER or SET PASSWORD commands.

Dropping Users
DROP USER removes one or more accounts from the server. Use DROP USER IF EXISTS to prevent errors in automated scripts.

Resource Limits
Define limits on queries, updates, and connections per hour using CREATE USER ... WITH clauses to prevent abuse or overuse.

Monitoring Connections
Use SHOW PROCESSLIST or INFORMATION_SCHEMA.PROCESSLIST to monitor active connections and terminate problematic sessions with KILL.

Resource Limits and Monitoring

Controlling User Resource Consumption

- **Per-User Resource Controls:** MySQL allows setting limits such as MAX_QUERIES_PER_HOUR, MAX_CONNECTIONS_PER_HOUR, and MAX_USER_CONNECTIONS to prevent resource abuse.
- **Modifying Limits:** ALTER USER ... WITH clauses can adjust limits dynamically without recreating the user. Removing limits is done via WITH UNLIMITED.
- **Monitoring Usage:** SHOW STATUS LIKE 'Queries'; and 'Max_used_connections'; provide metrics on user activity and system load.
- **Connection Management:** Active connections can be inspected via SHOW PROCESSLIST and terminated using KILL CONNECTION or KILL QUERY for troubleshooting.

SSL-Based Connections

Securing MySQL Client-Server Communication

- **Verifying SSL Support:** Use SHOW VARIABLES LIKE 'have_ssl'; to ensure SSL is enabled. Check '%ssl%' settings for configuration details.
- **Setting Up SSL:** Generate certificates via mysql_ssl_rsa_setup or OpenSSL. Configure ssl_ca, ssl_cert, and ssl_key in my.cnf for secure connections.
- **Requiring SSL for Users:** CREATE USER ... REQUIRE SSL mandates encrypted sessions. Can specify cipher suites or X.509 certificates for stronger identity verification.
- **Verifying SSL Sessions:** Use SHOW STATUS LIKE 'ssl_version'; to confirm SSL handshake and cipher in use. The MySQL client command \s also displays SSL details.

Password Policies and Validation

Enforcing Strong Authentication Standards

- **Validate Password Plugin:** Install and verify using INSTALL PLUGIN validate_password SONAME 'validate_password.so'; and SHOW PLUGINS LIKE 'validate_password';
- **Password Policy Levels:** Three policies available: LOW (length ≥8), MEDIUM (adds mixed case, digits, special chars), STRONG (includes dictionary checks).
- **Custom Policy Settings:** Administrators can adjust parameters such as validate-password.length, mixed_case_count, and special_char_count for stricter control.
- **Enforcement and Feedback:** Weak passwords trigger errors during CREATE USER or ALTER USER operations, ensuring compliance with configured standards.

Security Best Practices

Enhancing MySQL Database Protection

- **Strong Password Enforcement:** Always use complex passwords following validated policy requirements and periodically rotate credentials.
- **Principle of Least Privilege:** Grant users only the minimum privileges needed for their roles to reduce the attack surface and potential misuse.
- **Host Restrictions and SSL:** Restrict access to specific IPs or hosts and enforce SSL connections for all remote database users.
- **Auditing and Monitoring:** Regularly review user privileges, check for accounts without passwords, and monitor connections for anomalies.
- **Remove Default or Unused Accounts:** Delete default root@% and anonymous users, and ensure application accounts are distinct from administrative ones.

Key Takeaways & Summary

Consolidating MySQL Security Essentials

- **User Authentication:** MySQL accounts use the 'username'@'host' format with authentication via native or SHA2 password methods.
- **Privilege Management:** Access control is organized hierarchically across global, database, table, and column levels.
- **Resource and SSL Control:** Per-user resource limits and SSL-based connections ensure both performance fairness and encrypted communication.
- **Password and Policy Enforcement:** Validate password plugin enforces complexity and rotation standards, aligning database credentials with enterprise security.
- **Security Mindset:** Adopt least privilege, remove defaults, and audit regularly to maintain a hardened MySQL environment.