Co-funded by
the European Union

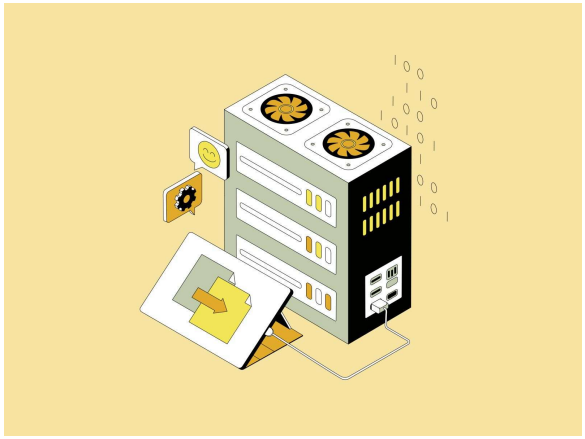
MERIT

Database Security

Secure Database Configuration

Secure Configuration: MySQL (my.cnf)

The my.cnf file is the main configuration file for the MySQL server.
It defines system-level parameters for behavior, performance, and security.
Proper configuration helps harden the server against attacks.
Security parameters should be explicitly defined to avoid vulnerabilities.

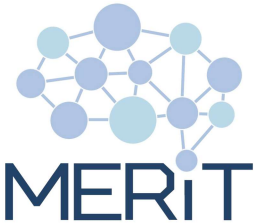


- bind-address = 127.0.0.1 → Limits access to localhost only
- skip-networking → Disables TCP/IP connections entirely
- local-infile = 0 → Prevents file injection via LOAD DATA
- secure-file-priv = /var/lib/mysql-files/ → Restricts import/export
- symbolic-links = 0 → Protects against symlink attacks
- default_authentication_plugin = caching_sha2_password → Modern auth
- validate_password.policy = STRONG → Enforces strong password policy
- log_error = /var/log/mysql/error.log → Centralized error logging
- log_warnings = 2 → Increases diagnostic verbosity

MySQL Secure Configuration – Script Example

```
#!/bin/bash
# Secure MySQL Configuration Script  # Author: Rubén López  Barrio
CONF_FILE="/etc/mysql/mysql.conf.d/mysqld.cnf“
BACKUP_FILE="/etc/mysql/mysql.conf.d/mysqld.cnf.bak“
echo "[INFO] Backing up current configuration...“
cp "$CONF_FILE" "$BACKUP_FILE“
echo "[INFO] Applying secure MySQL settings...“

# 1. Restrict access to localhost only
sed -i '/^bind-address/d' "$CONF_FILE“
echo "bind-address = 127.0.0.1" >> "$CONF_FILE“

# 2. Disable remote file loading (prevents file injection)
sed -i '/^local-infile/d' "$CONF_FILE“
echo "local-infile = 0" >> "$CONF_FILE“

# 3. Disable symbolic links (prevents symlink attacks)
sed -i '/^symbolic-links/d' "$CONF_FILE"echo "symbolic-links = 0" >> "$CONF_FILE“

# 4. Enforce secure password plugin (if available)
echo "plugin-load-add=validate_password.so" >> "$CONF_FILE“

# 5. Set secure password policy (requires validate_password plugin)
echo "validate-password-policy=STRONG" >> "$CONF_FILE"echo "validate-password-length=12" >> "$CONF_FILE“

# 6. Enable error logging to a dedicated file
echo "log-error=/var/log/mysql/error.log" >> "$CONF_FILE

"# 7. Set secure file operations directory
echo "secure-file-priv=/var/lib/mysql-files" >> "$CONF_FILE"
echo "[INFO] Restarting MySQL service...“
systemctl restart mysql
echo "[DONE] Secure MySQL configuration applied."
```

Secure Configuration: PostgreSQL (postgresql.conf)

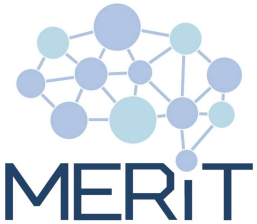


The postgresql.conf file is the primary configuration file for PostgreSQL.
It controls security, performance, connection settings, and logging.

Proper tuning of this file helps protect against unauthorized access and data breaches.

Should be used in combination with pg_hba.conf for access control.

Key Security Parameters in postgresql.conf



- listen_addresses = 'localhost' → Prevents remote connections
- ssl = on → Enables encrypted connections (requires certs)
- password_encryption = scram-sha-256 → Strong password storage
- logging_collector = on → Enables log file writing
- log_connections = on → Logs successful connections
- log_disconnections = on → Logs disconnections with duration
- log_line_prefix = '%m [%p] %u@%d ' → Adds detail to logs
- client_min_messages = warning → Suppresses verbose client output

PostgreSQL Secure Configuration – Script Example

```
#!/bin/bash
# Secure PostgreSQL Configuration Script  Author: Rubén López Barrio

CONF_FILE="/etc/postgresql/14/main/postgresql.conf“
HBA_FILE="/etc/postgresql/14/main/pg_hba.conf“
BACKUP_CONF="/etc/postgresql/14/main/postgresql.conf.bak“
BACKUP_HBA="/etc/postgresql/14/main/pg_hba.conf.bak“

echo "[INFO] Backing up configuration files...“
cp "$CONF_FILE" "$BACKUP_CONF“
cp "$HBA_FILE" "$BACKUP_HBA“
echo "[INFO] Applying secure PostgreSQL settings...“

# 1. Only listen on localhost
sed -i "s/^#*listen_addresses.*/listen_addresses = 'localhost'/" "$CONF_FILE“

# 2. Enable SSL connections
sed -i "s/^#*ssl.*/ssl = on/" "$CONF_FILE“

# 3. Enforce secure password encryption
sed -i "s/^#*password_encryption.*/password_encryption = scram-sha-256/" "$CONF_FILE“

# 4. Enable detailed logging
sed -i "s/^#*logging_collector.*/logging_collector = on/" "$CONF_FILE“
sed -i "s/^#*log_connections.*/log_connections = on/" "$CONF_FILE“
sed -i "s/^#*log_disconnections.*/log_disconnections = on/" "$CONF_FILE“
sed -i "s/^#*log_line_prefix.*/log_line_prefix = '%m [%p] %u@%d '/" "$CONF_FILE“
sed -i "s/^#*client_min_messages.*/client_min_messages = warning/" "$CONF_FILE“

# 5. Restrict pg_hba.conf to use strong auth
echo "host all all 127.0.0.1/32 scram-sha-256" > "$HBA_FILE“
echo "host all all ::1/128 scram-sha-256" >> "$HBA_FILE“
echo "local all all peer" >> "$HBA_FILE“
echo "[INFO] Restarting PostgreSQL service...“
systemctl restart postgresql
echo "[DONE] Secure PostgreSQL configuration applied."
```

Comparative Table: Secure MySQL vs PostgreSQL Configuration

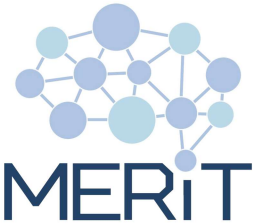
Security Setting	MySQL	PostgreSQL
Localhost-only access	bind-address = 127.0.0.1	listen_addresses = 'localhost'
Disable remote loading/injection	local-infile = 0	N/A
Disable symbolic links	symbolic-links = 0	N/A
Enable SSL/TLS	N/A (requires certs separately)	ssl = on
Secure password plugin/encryption	validate-password-policy = STRONG	password_encryption = scram-sha-256
Set password length	validate-password-length = 12	N/A
Restrict import/export operations	secure-file-priv = /var/lib/mysql-files	N/A
Enable logging	log-error, log-warnings	logging_collector, log_connections
Log line formatting	N/A	log_line_prefix = '%m [%p] %u@%d '
Client verbosity reduction	N/A	client_min_messages = warning
Authentication control via HBA	N/A	pg_hba.conf with scram-sha-256
Backup of configuration	mysqld.cnf.bak	postgresql.conf.bak, pg_hba.conf.bak
Restart service after changes	systemctl restart mysql	systemctl restart postgresql

Disabling Unused Features and Ports

Concept	MySQL	PostgreSQL
Disable network access	Add skip-networking to mysqld.cnf	Set listen_addresses = 'localhost' in postgresql.conf
Remove unused plugins	UNINSTALL PLUGIN plugin_name; via mysql CLI	Not applicable – PostgreSQL does not include unnecessary plugins by default
Restrict access to local machine	bind-address = 127.0.0.1	listen_addresses = 'localhost'
Close database ports	ufw deny 3306/tcp	ufw deny 5432/tcp
Remove test databases	mysql_secure_installation removes test db	No test DB is created by default in PostgreSQL
Remove anonymous users	mysql_secure_installation removes them	PostgreSQL uses system user mapping, anonymous access is not allowed
Use firewall to restrict connections	UFW, firewalld, or iptables	UFW, firewalld, or iptables
Restart after config changes	systemctl restart mysql	systemctl restart postgresql

PostgreSQL vs MySQL

Logging and Monitoring in MySQL and PostgreSQL



Feature	MySQL	PostgreSQL
Enable general logging	general_log = 1 (my.cnf)	Use log_statement = 'all' (postgresql.conf)
Log file path	log_error = /var/log/mysql/error.log	log_directory = 'pg_log', log_filename = 'postgresql-%Y-%m-%d.log'
Log connections	Not logged by default; use general_log	log_connections = on
Log disconnections	Not available	log_disconnections = on
Query log filtering	Use general_log and log_output = FILE	Use log_min_duration_statement
Log format customization	Not flexible	log_line_prefix = '%m [%p] %u@%d '
Audit plugin support	Supported via third-party plugin (e.g., McAfee Audit Plugin)	pgAudit extension available for detailed audits
Log rotation	Handled by system logrotate	PostgreSQL handles rotation by filename/date
Monitoring tools	Percona Toolkit, mysqladmin, slow query log	pg_stat_statements, pgBadger, Prometheus exporters
Enable slow query log	slow_query_log = 1, long_query_time = 2	Use pg_stat_statements and log_min_duration_statement

Sumary

- ✔ Secure configuration files: my.cnf (MySQL), postgresql.conf (PostgreSQL)
- ✔ Disable unused features: skip networking, remove test DBs and plugins
- ✔ Restrict access: bind to localhost, firewall rules (UFW)
- ✔ Enforce strong authentication: SCRAM, password policies
- ✔ Enable logging and monitoring: error logs, query logs, pgA