MariaDB security features and best practices

Robert Bindar
Software Developer @MariaDB Foundation

Percona Live
Austin, 28-30 May 2019
Motivation - Users

- Potential public shaming through data breaches
- Massive loss of business
- Angry Clients
- Expensive lawsuits and fines
- And it’s getting worse as more people start realizing the impact of data security
- Bonus points for being compliant with data protection regulations
Motivation - MariaDB

- One of the most popular db servers
- Part of critical infrastructure worldwide
- Its main purpose is to manage data
- Very important for our users
Potential threats and defense mechanisms
Direct DB threats

**Threat**

- Man in the middle attacks
- Spoofing
- Memory corruption exploits

**Prevention**

- Limit/block outside TCP connections to MariaDB
- Secure your DNS infrastructure
- MariaDB should accept connections only from the application host
- Use bind_address
- Use TLS/SSL
- Keep your OS updated
- Keep your MariaDB Server updated
Application threats

<table>
<thead>
<tr>
<th>Threat</th>
<th>Prevention</th>
</tr>
</thead>
<tbody>
<tr>
<td>DOS attacks</td>
<td>Your MariaDB server should ideally run on a dedicated machine</td>
</tr>
<tr>
<td>Data leaks/corruption</td>
<td>Avoid running the application on the DB machine</td>
</tr>
<tr>
<td>SQL injection</td>
<td>Keep the DB machine as clean as possible</td>
</tr>
<tr>
<td></td>
<td>Have a strong permissions system</td>
</tr>
<tr>
<td></td>
<td>Application code security practices</td>
</tr>
</tbody>
</table>
Limiting Human Errors

Threat

• Genuine human mistakes
• Bad intentions

Prevention

• Limit sudo access on the MariaDB server machine
• Limit ssh access
• Avoid running mysqld as root
• Use specific hostnames instead of wildcards
• Use secure_file_priv
• Robust defaults

MariaDB Security Features and Best Practices
https://mariadb.org
Secure Installations with mysql_secure_installation
It won’t provide bullet-proof security for your deployment. This script just presents a basic set of recommended settings to get started.
mysql_secure_installation

- Set/Change root accounts passwords
- Delete root accounts accessible from outside
- Remove anonymous user accounts
- Remove test database
- FLUSH PRIVILEGES on the house!
Data Encryption
**In-Transit Data:**

Data transmitted between clients and the MariaDB server, between server instances in replication or data transmitted within the Galera cluster. Defaults to unencrypted!

**At-Rest Data:**

Some of the data residing in persistent storage: tables, tablespaces, binary logs. Supported with InnoDB and XtraDB, partially with Aria.
Encryption Libraries in MariaDB

- MariaDB uses TLS
- static linking with yaSSL - server + client
- dynamic linking with OpenSSL - server + client
- dynamic linking with GnuTLS or Schannel - client
- `have_ssl` will tell you if TLS is supported/enabled

MariaDB [(none)]> SHOW GLOBAL VARIABLES LIKE 'version_ssl_library';
| Variable_name | Value                      |
|----------------+----------------------------+
| version_ssl_library | OpenSSL 1.1.0g  2 Nov 2017 |
Server <-> Clients data encryption with TLS

```
[mariadb]
ssl_cert = /etc/my.cnf.d/certificates/server-cert.pem
ssl_key = /etc/my.cnf.d/certificates/server-key.pem
ssl_ca = /etc/my.cnf.d/certificates/ca.pem
```

- Defaults to not encrypted
- `have_ssl == YES` means TLS is enabled
- FLUSH SSL reloads TLS context from 10.4
- Two-way TLS is required if `REQUIRE X509`, `REQUIRE SUBJECT`, `REQUIRE ISSUER` are used for an account
- TLS can be required for specific accounts from untrusted hosts
Secure Connections in Replication

- Defaults to not encrypted
- Enable TLS on all server instances
- Stop slaves and execute CHANGE MASTER
- Two-way TLS can also be enabled with CHANGE MASTER

```
MariaDB [(none)]> CHANGE MASTER TO
  MASTER_SSL_CA = '/path/to/ca/ca.pem',
  MASTER_SSL_VERIFY_SERVER_CERT=1;
```
Encryption for Galera Cluster

- Defaults to not encrypted
- Enable TLS on each server node
- Add wsrep ssl options to cnf along the server ssl options
- Traffic is encrypted within the cluster and with external client connections
- Backup utilities also support encryption

```bash
[mariadb]
... cert,key,ca
wsrep_provider_options="socket.ssl_cert=/path/server-cert.pem;
socket.ssl_key=/path/server-key.pem;
socket.ssl_ca=/path/ca.pem"
```
At-Rest Data Encryption

- Encrypting some of the data on disk
- Overhead is estimated at 3-5%
- Supported fully with InnoDB and XtraDB SEs
- Aria support for ROW_FORMAT=PAGE tables
- You need to install an encryption management plugin
- Only helpful if the attacker is not an authorized MariaDB user
Password Validation Plugins

Simple Password Check

• .so shipped with MariaDB - easy install
• Minimum length
• Mixed case
• Alphanumeric checks
• Special chars
• Can be used with PAM as of 10.4

Cracklib Password Check

• Not shipped by default with MariaDB
• Checks password against a dictionary
• Uses the CrackLib db
• Can be used with PAM as of 10.4

MariaDB Security Features and Best Practices
https://mariadb.org
Unix Socket Authentication

MariaDB [(none)]> CREATE USER username@hostname IDENTIFIED VIA unix_socket;
Query OK, 0 rows affected (0.00 sec)

- Use OS credentials when connecting to MariaDB
- Enabled by default in 10.4.3
Account Locking

- Mark an account as *locked* and deny any subsequent connection requests for that account
- Minimum privilege package = no client connection at all
- Integrated solution for refusing client connections
Account Locking

- Creates a user account that is locked

MariaDB [(none)]> CREATE USER user@localhost ACCOUNT LOCK;
Query OK, 0 rows affected (0.00 sec)
Account Locking

SHOW CREATE USER user@localhost;

```
| CREATE USER for user@localhost |
| CREATE USER 'user'@'localhost' ACCOUNT LOCK |
```

1 row in set (0.000 sec)

- **SHOW CREATE USER** displays the locking status of an account
Account Locking

- Altering an existing account to lock/unlock

MariaDB [(none)]> ALTER USER user@localhost ACCOUNT UNLOCK;
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> ALTER USER user@localhost ACCOUNT LOCK;
Query OK, 0 rows affected (0.00 sec)
Account Locking

Attempting a connection using a locked account returns

ER_ACCOUNT_HAS_BEEN_LOCKED
Account Locking

Whether an account is locked or not is checked during the authentication phase (including COM_CHANGE_USER).

Locking an account does **not** affect existing connections.
Expiration of User Passwords

- A new connection with an expired password is either denied or only allowed to execute `SET PASSWORD`
- Supports expiring passwords with immediate effect, per-account automatic expiration as well as global policies for automatic expiration
- Compliance with latest security standards
- Fully compatible with MySQL 5.7 datadirs
Password Expiration

MariaDB [(none)]> CREATE USER user@localhost PASSWORD EXPIRE;
Query OK, 0 rows affected (0.00 sec)

- Creates a new account and expire the password with immediate effect
Password Expiration

**Disconnect Mode:**
In this mode, any new connections for accounts with expired passwords are refused.

**Sandbox Mode:**
A new connection for an account with the password expired is only allowed to execute SET PASSWORD to change the account password, attempts to execute any other statements are rejected.
Password Expiration

- `disconnect_on_expired_password` system var (default OFF) controls how clients unaware of the sandbox mode are treated.

- But `--connect-expired-password` arg passed to the client takes precedence and the server knows to put the connection in sandbox mode.

- Also `interactive` client connections are always put in sandbox mode.

- In the MariaDB C Connector, `MYSQL_OPT_CAN_HANDLE_EXPIRED_PASSWORDS` can be passed to `mysql_options` to achieve a similar behavior.
The client is still able to connect to the server, but only the SET PASSWORD statement is allowed for changing the account password.

Executing any other statement returns ER_MUST_CHANGE_PASSWORD.
Password Expiration

Trying to connect using an expired password account returns

```
$ mysql -u user
ERROR 1862 (HY000): Your password has expired. To log in you must change
it using a client that supports expired passwords
```

● *Trying to connect using an expired password account returns*

```
ER_MUST_CHANGE_PASSWORD_LOGIN
```
Password Expiration

- `default_password_lifetime` (default 0) controls the global automatic password expiration policy.

- Can be set at runtime using `SET GLOBAL`, specified in the config file or as server arg (`--default-password-lifetime=90`).

- `default_password_lifetime`=0 means passwords never expire.

- `default_password_lifetime`=90 means passwords expire every 90 days.

- But per-account expiration policies override the global policies.
The password of this account will never expire regardless of what global policies say.
Password Expiration

MariaDB [(none)]> ALTER USER user@localhost PASSWORD EXPIRE INTERVAL 30 DAY;
Query OK, 0 rows affected (0.00 sec)
MariaDB [(none)]> SHOW CREATE USER user@localhost;

<table>
<thead>
<tr>
<th>CREATE USER for user@localhost</th>
</tr>
</thead>
<tbody>
<tr>
<td>CREATE USER 'user'@'localhost' PASSWORD EXPIRE INTERVAL 30 DAY</td>
</tr>
</tbody>
</table>

1 row in set (0.00 sec)

- default_password_lifetime is overriden and for this account the password will expire every 30 days
By specifying DEFAULT as per-account policy, the value in the default_password_lifetime sys var will be used.
--max-password-errors

MariaDB [(none)]> SET GLOBAL max_password_errors=3;
Query OK, 0 rows affected (0.00 sec)

- Or pass --max-password-errors=N to mysqld
- Denies further connections if a password was wrong the previous N times
- FLUSH PRIVILEGES resets the counter
Role-based Access Control

- Separation of privileges per group of users
- It’s possible to set a default role per user
- Only one active role at a time
- A role can be assigned to another role
- Managing privileges for groups becomes easier
- It’s easy to inspect roles info through I_S
Monitoring Server Activity
MariaDB Audit Plugin

- Log server’s activity for each client session
- Username Host for each connection
- Executed queries
- Accessed tables
- Updates to server variables
- Shipped with MariaDB
- Compliance with audit regulations
Thank You!

Contact details:
robert@mariadb.org

About:
mariadb.org/about/staff/robert-bindar/