MariaDB Roadshow 2016

MariaDB

Secure Data Management - Threats and Best Practices

Common Threats to Your Data

1. The Internet
2. Applications
3. Excessive Trust
4. Outdated MariaDB packages
5. Lack of Visibility

The Internet

- Database driven Webpages are commodity now
- Started as a small hosted Web application, now business critical
- ...

Defense

- Do not allow TCP connections to MariaDB from the Internet at large.
- Configure MariaDB to listen on a network interface that is only accessible from the host where your application runs.
- Design your physical network to connect the app to MariaDB
- Use bind-address to bind to a specific network interface
- Use your OS's firewall

Best Practice: Encrypt sensitive data

- Encrypt some data in the application
  - Non-key data
  - Credit card numbers
- Encrypt data in transit using SSL
  - From clients to MaxScale
  - From clients to MariaDB
  - Between MariaDB replication nodes
- Encrypt data at rest using advanced tablespace encryption functionality in MariaDB 10.1
  - InnoDB tablespace encryption
  - InnoDB redo log encryption
  - Binary log encryption

Threats from Applications

- Denial of Service Attacks created by overloading application
- SQL query injection attacks
- …

Defense

- Do not run your application on your MariaDB Server.
- Do not install unnecessary packages on your MariaDB Server.
- An overloaded application can use so much memory that MariaDB could slow or even be killed by the OS. This is an effective DDoS attack vector.
- A compromised application or service can have many serious side effects
  - Discovery of MariaDB credentials
  - Direct access to data
  - Privilege escalation

Best Practice: Use a Gateway

- Create a Database Firewall
- Restrict the operations that clients (applications) are allowed to perform
- Identify and flag potentially dangerous queries
- Customize rules about what's allowed and what's not
- Implement connection pooling capabilities can protect against DDoS attacks

Excessive Trust

- Disgruntled employees
- Mistakes and human error

- Do not use the MariaDB “root” user for application access.
- Grant only the privileges required by your application.
- Minimize the privileges granted to the MariaDB user accounts used by your applications
  - Don’t grant CREATE or DROP privileges.
  - Don’t grant the FILE privilege.
  - Don’t grant the SUPER privilege.
  - Don’t grant access to the mysql database
- Limit users who have:
  - SSH access to your MariaDB server.
  - Sudo privileges on your MariaDB server.
- Set the secure_file_priv option to ensure that users with the FILE privilege cannot write or read MariaDB data or important system files.

Best Practice: Manage MariaDB user accounts carefully

- Use OS permissions to restrict access to MariaDB data and backups.
- Allow root access to MariaDB only from local clients—no administrative access over the network.
- Use the unix_socket authentication plugin so that only the OS root user can connect as the MariaDB root user.
- Use strong passwords.
  - Enable the cracklib_password_check plugin.
- Use a separate MariaDB user account for each of your applications.
- Allow access from a minimal set of IP addresses.

Outdated MariaDB Packages

- Linux vendors often distribute outdated versions of MariaDB which lack the most up-to-date security fixes and features:
  - MariaDB 10.0 in Debian 8 (Jessie)
  - MariaDB 5.5 in RHEL 7
- Use MariaDB Enterprise packages:
  - Updated with the most-recent security fixes and features
  - Critical security features enabled by default

Best Practice: Update MariaDB and other packages

- Stay on top of the most recent security fixes by keeping your MariaDB packages updated
- Apply security updates distributed by your OS vendor, as highlighted by recent problems in glibc and openssl.

Lack of Visibility

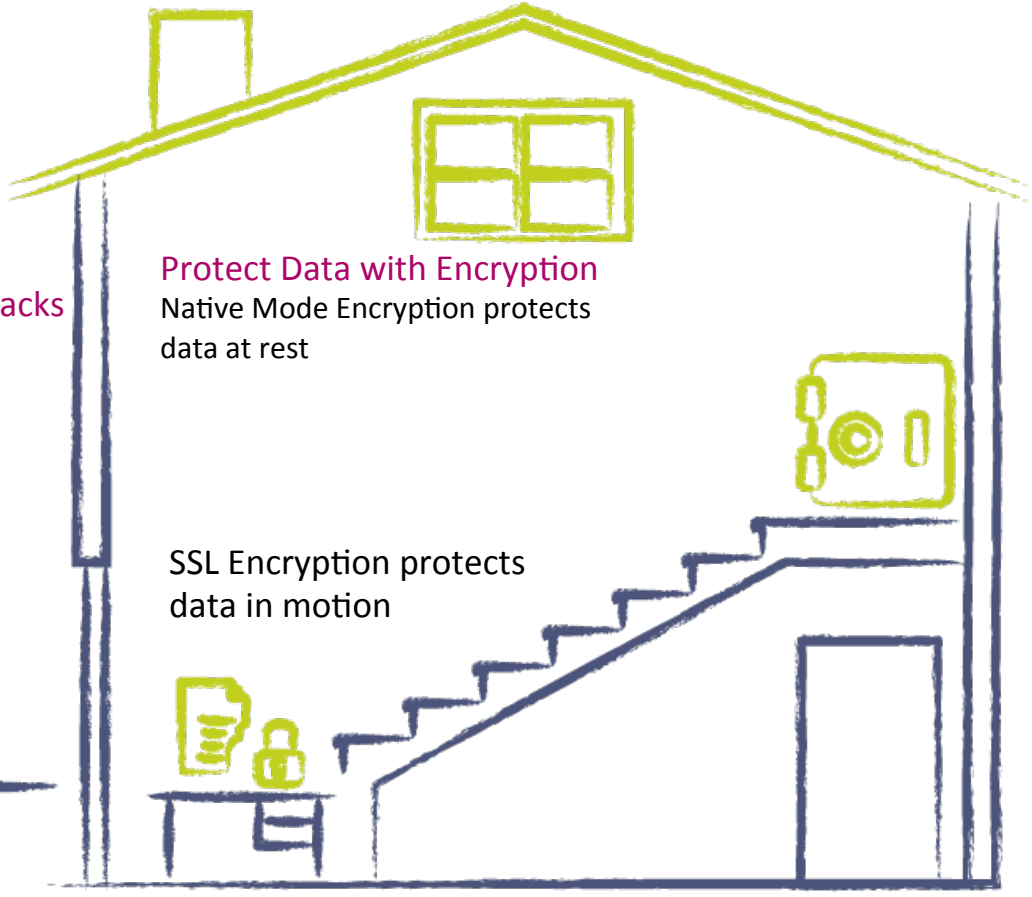
- Applications share the same user for database connections
  - No visibility at the database backend about which application is accessing the data
- Scripts do not use specific users or even use the root user
  - No chance to evaluate which tool is causing issues
- Direct access to database without using named DBA users
  - No way to track which DBA / DevOpp did access data
- Requirement to audit access to data is increasing
  - We need to know who is accessing what and when

Best Practice: Named Users and Audit

- Use named users whenever possible
  - Distinguish between technical users (applications) and direct access (tools, DBA, ..)
- Ensure regulatory compliance with robust logging
- Record connections, query executions, and tables accessed
- Activate auditing using the MariaDB Audit Plugin
  - Use logs for forensic analysis after an incident
  - Enable ediscovery
  - Log either to a file or to syslog

Detect, Protect, Audit, Improve

MariaDB Enterprise Security



Protect Data with Encryption

Native Mode Encryption protects data at rest

SSL Encryption protects data in motion

Detect and Prevent Attacks

- Unauthorized Access
- Denial of Service
- SQL Injections

MariaDB 10.1
InnoDB / XtraDB
Aria

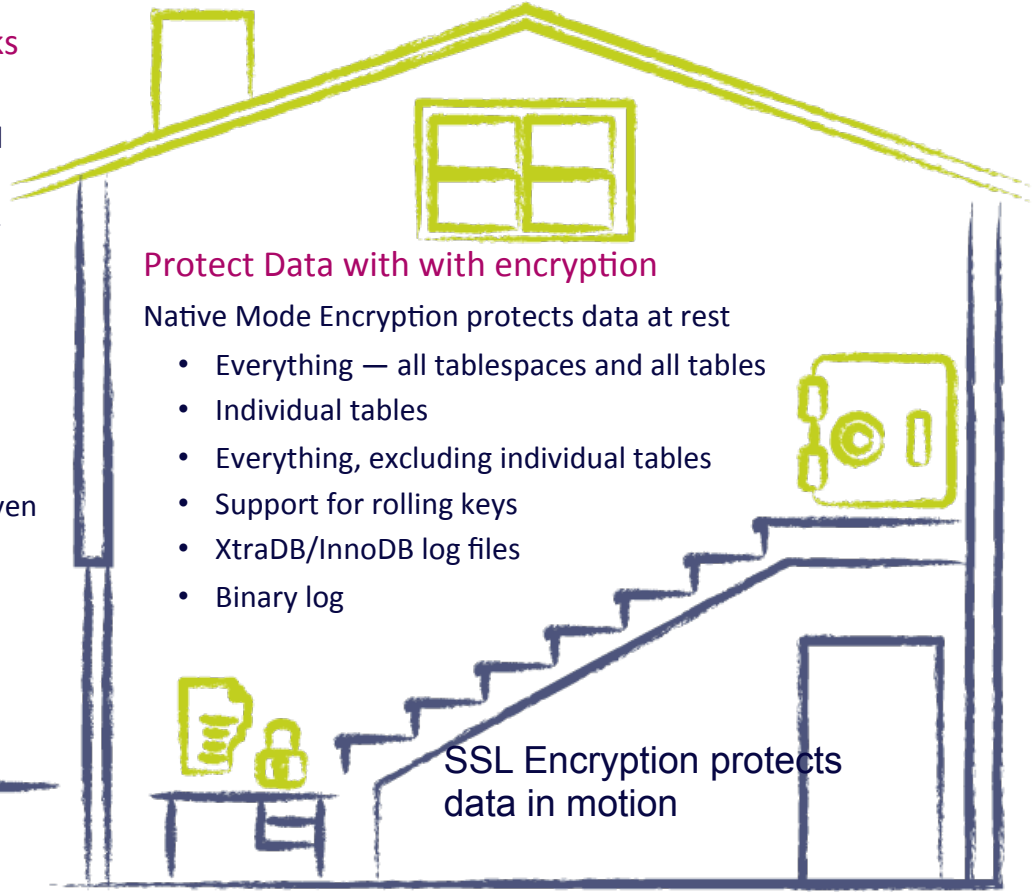
Benefit from Community Protection

Audit for Forensics and Compliance

MariaDB Enterprise Security

Detect and Prevent Attacks

- Role Based Access Control
- Password management and validation plugin
- Key Management Services - AWS or Eperi KMS
- Blacklist firewall filtering in MaxScale
- Authentication plug in
  1. LDAP
  2. ssh passphrases
  3. One-time passwords (even with SMS confirmation)
  4. System authentication
  5. Combinations of authentication modules



Protect Data with with encryption

Native Mode Encryption protects data at rest

- Everything — all tablespaces and all tables
- Individual tables
- Everything, excluding individual tables
- Support for rolling keys
- XtraDB/InnoDB log files
- Binary log

SSL Encryption protects data in motion

Benefit from Community Protection

- Faster detection of vulnerabilities
- Better threat response
- Security features

Audit for Forensics and Compliance

- Log database connection, queries and table access

Authentication

Password Validation

- **Simple_password_check plugin**
  Enforce a minimum password length and type/number of characters to be used
- **Cracklib_password_check plugin**
  - Stop users from choosing easy to guess passwords.
  - Prohibit weak passwords based on username or dictionary word

External Authentication

Single Sign On is getting mandatory in most Enterprises.

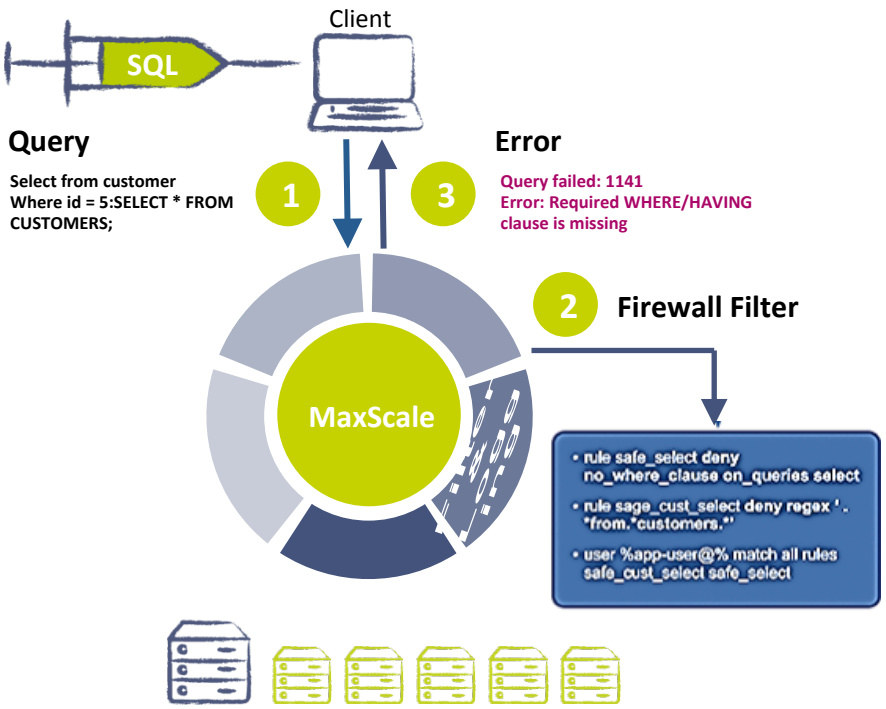
- **PAM-Authentication Plugin** allows using /etc/shadow and any PAM based authentication like LDAP
- **Kerberos-Authentication** as a standardized network authentication protocol is provided GSSAPI based on UNIX and SSPI based on Windows

Threat Protection with the Database Firewall

Protect against SQL injection
Prevent unauthorized data access
Prevent data damage

How it Works

- Block or Allow queries that
  - match a set of rules
  - matching rules for specified users
  - match certain patterns
- Multiple ordered rules
- Match on
  - date/time
  - a WHERE clause
  - Query type
  - Column match
  - a wildcard or regular expression



Denial of Service attack protection

- MariaDB MaxScale Persistent Connections
- Connection pooling protects against connection surges
- Cache the connections from MaxScale to the database server
- Rate limitation
- Client multiplexing

Encryption for Data in Motion

Secured Connections

- SSL Connections based on the TLSv1.2 Protocol
- Between MariaDB Connectors and Server
- Between MariaDB Connectors and MaxScale
- SSL can also be enabled for the replication channel

Encryption Functions

- Selective Data-In-Use Encryption
- Application control of data encryption
- Based on the AES (Advanced Encryption Standard) or DES (Data Encryption Standard) algorithm

Encryption for Data at Rest

Data-at-Rest Encryption

- Table or tables spaces
- Log files
- Independent of encryption capabilities of applications
- Based on encryption keys, key ids, key rotation and key versioning

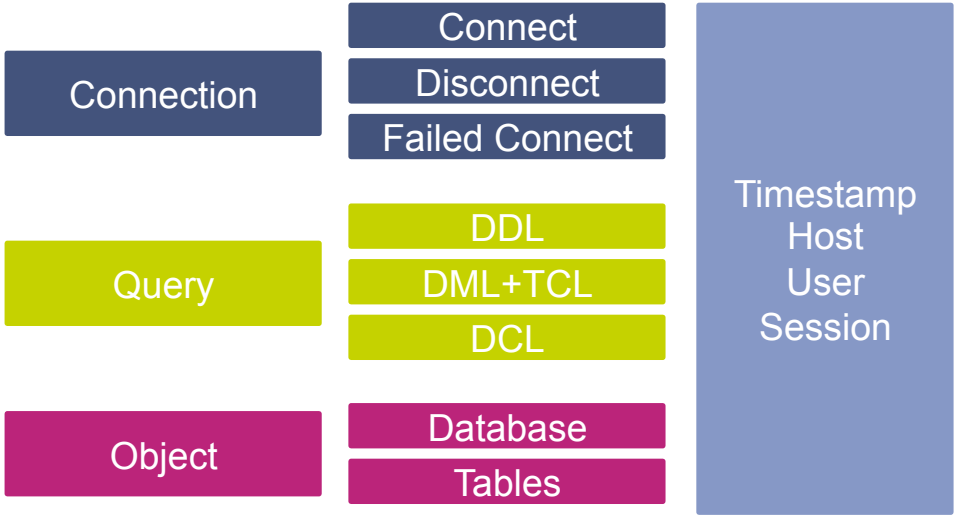
Key Management Services

- Encryption plugin API offers choice
  - Plugin to implement the data encryption
  - Manage encryption Keys
- MariaDB Enterprise options
  - Simle Key Management included
  - Amazon AWS KMS Plugin included
  - Eperi KMS for on premise key management - optional

Auditing for Security and Compliance

MariaDB Audit Plugin

- Logs server activity
  - Who connected to the server
  - Source of connection
  - Queries executed
  - Tables touched
- File based or syslog based logging



MariaDB Security Gets Stronger All the Time

MariaDB User Community

- Quickly identifies new threats
- Reports vulnerabilities
- Creates solutions
- Contributes features

MariaDB Products Solutions Customers Partners Resources News & Events About

Home » Resources » Knowledge Base » MariaDB » Development » Security Vulnerabilities Fixed in MariaDB

Security Vulnerabilities Fixed in MariaDB

Home
Open Questions
MariaDB
MariaDB Enterprise
MaxScale
All Topics

History
Subscriptions
Edit
Source
Flag as Spam/Inappropriate
Translate

Created
1 year, 4 months ago
Modified
7 months, 3 weeks ago
Type
article
Status
active
License

This page is about security vulnerabilities fixed in MariaDB. If you are looking for information on securing your MariaDB installation, see Securing MariaDB.
Sensitive security issues can be sent directly to the persons responsible for MariaDB security: security [AT] mariadb (dot) org.

Contents

1. About CVEs
2. Full List of CVEs fixed in Maria
   1. CVEs without specific versi
   2. CVE's affecting Oracle MyS

About CVEs

CVE® stands for *"Common Vulnerabilities and Exposures"*. It is a publicly available and free to use databas software vulnerabilities maintained at https://cve.mitre.org/

On this page is the master list of CVEs fixed across all versions of MariaDB. Follow the links to more inform particular CVE or specific version of MariaDB.

Separate lists of CVEs fixed in specific MariaDB series are maintained on their individual "What is MariaDB

- What is MariaDB 10.1?
- What is MariaDB 10.0?
- What is MariaDB 5.5?
- What is MariaDB 5.3?
- What is MariaDB 5.2?
- What is MariaDB 5.1?

Full List of CVEs fixed in MariaDB

- CVE-2016-2047: MariaDB 5.5.47, MariaDB 10.1.10, MariaDB 10.0.23
- CVE-2016-0616: MariaDB 5.5.47, MariaDB 10.1.10, MariaDB 10.0.23
- CVE-2016-0610: MariaDB 10.1.9, MariaDB 10.0.22
- CVE-2016-0609: MariaDB 5.5.47, MariaDB 10.1.10, MariaDB 10.0.23
- CVE-2016-0608: MariaDB 5.5.47, MariaDB 10.1.10, MariaDB 10.0.23
- CVE-2016-0606: MariaDB 5.5.47, MariaDB 10.1.10, MariaDB 10.0.23
- CVE-2016-0600: MariaDB 5.5.47, MariaDB 10.1.10, MariaDB 10.0.23
- CVE-2016-0598: MariaDB 5.5.47, MariaDB 10.1.10, MariaDB 10.0.23
- CVE-2016-0597: MariaDB 5.5.47, MariaDB 10.1.10, MariaDB 10.0.23
- CVE-2016-0596: MariaDB 5.5.47, MariaDB 10.1.10, MariaDB 10.0.23

GET STARTED: MariaDB Security Audit

Evaluate and address database security policies, technologies, and practices

- Review of your database security needs and requirements
- Access control assessment
- Automated attack protection review
- Encryption tools and practices
- Forensic capabilities review
- Ongoing compliance and security planning

Fully leverage MariaDB's security capabilities

Reduce legal, financial, and brand reputation risk

Q&A

Thank You