Co-funded by
the European Union

MERIT

Database Security

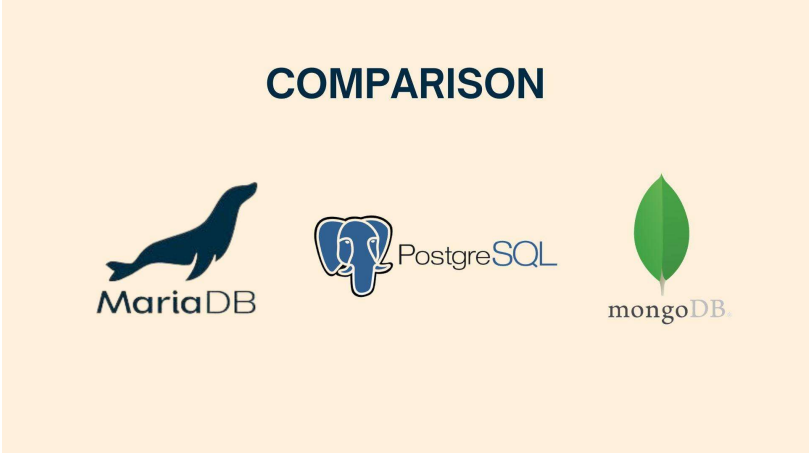
Modern Open Source Databases and Security Features

Modern Open Source Databases and Security Features

MERiT

In today's data-driven world, securing **open-source databases** like **MariaDB, PostgreSQL and MongoDB** is paramount. This presentation dives into the security features of these databases, emphasizing practical applications and compliance requirements. We will examine authentication methods, access control mechanisms, and encryption techniques critical for robust data protection.

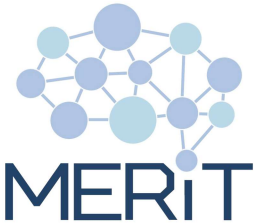
Our focus will be on enabling database administrators and security engineers to bolster their defenses against evolving threats. By understanding and implementing these security measures, organizations can maintain data integrity, ensure regulatory compliance, and fortify their overall security posture.



Security Features – PostgreSQL

PostgreSQL offers a range of security features designed to protect sensitive data and ensure secure database operations. These features are crucial for maintaining compliance and mitigating potential threats. Here are some key security aspects of PostgreSQL:

Security Features – PostgreSQL



🔒 **1. Data Masking / Obfuscation**
This technique involves hiding or altering sensitive data (such as personal identifiers) in a way that it remains usable for testing or analytics but is no longer traceable to real individuals. It helps protect privacy and is useful in non-production environments.

🔐 **2. Transparent Data Encryption (TDE)**
TDE encrypts data at rest automatically, meaning that files stored on disk (like tables and indexes) are encrypted without requiring application changes. It enhances data security in case of disk theft or unauthorized physical access.

🗝️ **3. Stored Procedure Encryption**
This feature allows the encryption of business logic written in stored procedures or functions. It helps prevent reverse engineering, protects intellectual property, and adds another layer of control over sensitive operations.

🛡️ **4. Security Policies**
Security policies define access controls, permissions, and compliance rules at the database level. These policies ensure that users and roles only have the access necessary to perform their tasks, enforcing the principle of least privilege.

🔒 **5. Client / Server Encryption**
This ensures that all communication between PostgreSQL clients and the database server is encrypted using SSL/TLS. It prevents eavesdropping, man-in-the-middle attacks, and data interception during transmission.

👤 **6. Single Sign-On (SSO)**
SSO allows users to authenticate once and gain access to multiple systems, including PostgreSQL, using integrated identity providers (e.g., Kerberos, LDAP, Active Directory). It improves security and user experience by centralizing authentication.

Security Features – MariaDB

MariaDB incorporates robust security features to protect data at rest and in transit. These capabilities are essential for meeting stringent security requirements and ensuring data integrity.

The following security features are available in MariaDB:

Pluggable Authentication Modules

PAM, LDAP, and Kerberos integration for flexible authentication options.

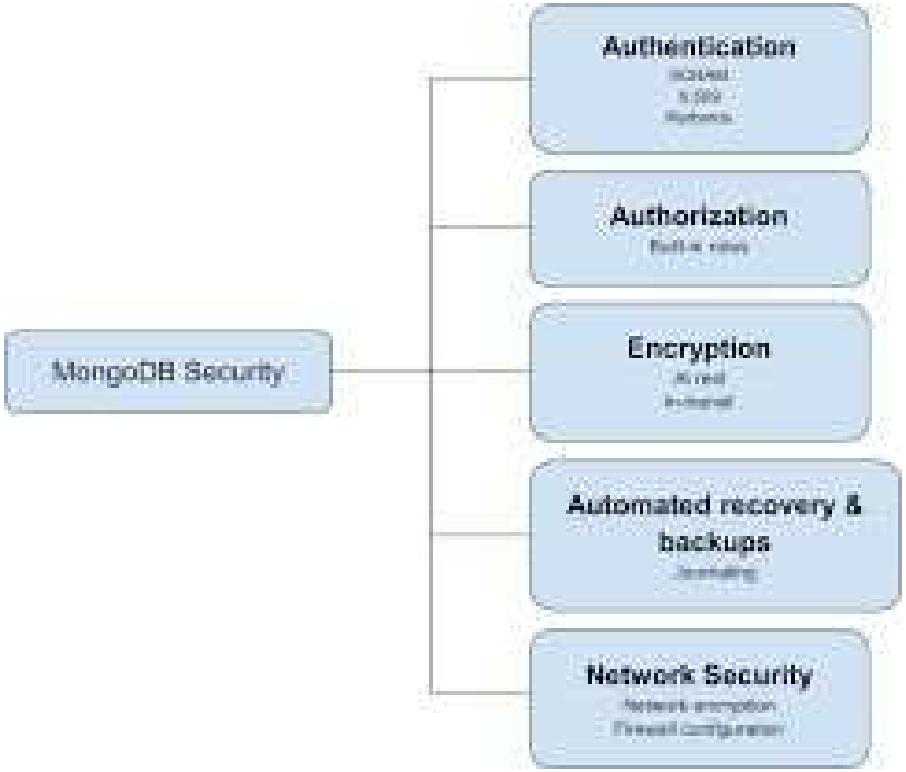
Data-at-Rest Encryption

Encryption of data files with key rotation capabilities.

User Roles and Dynamic Privilege Management

Streamlined management of user privileges.

Security Features – MongoDB



MongoDB provides a comprehensive suite of security features aimed at protecting data across various layers. From authentication to network encryption, MongoDB offers the tools necessary to build a secure database environment.

Authentication

Supports SCRAM, LDAP, and X.509 authentication methods.

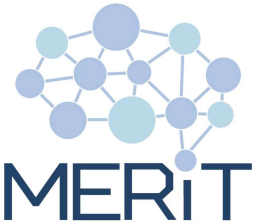
Role-Based Access Control (RBAC)

Granular control over data access based on user roles.

Encryption

TLS for transport and encrypted storage engine for data-at-rest.

Authentication Methods: SCRAM, LDAP, X.509



Authentication is the cornerstone of database security, ensuring that only authorized users gain access.

Open source databases offer various methods to achieve this, each with its own strengths and use cases.

SCRAM (Salted Challenge Response Authentication Mechanism)

The default authentication mechanism in PostgreSQL and MongoDB, providing robust password protection.

LDAP (Lightweight Directory Access Protocol)

Enables centralized credential management, simplifying user administration across multiple systems.

X.509 Certificates

Utilized for TLS-based client authentication, ensuring secure connections via mutual authentication.

Authentication Methods: SCRAM, LDAP, X.509

Feature	SCRAM	LDAP	X.509
Type	Challenge-Response	Directory-based Authentication	Certificate-based Authentication
Used in	Databases, SASL, PostgreSQL, MongoDB	Active Directory, OpenLDAP, enterprise apps	HTTPS, TLS, VPN, SSO
Authentication	Password with salt and hash	Username and password	X.509 digital certificates
Security	High (passwords not exposed)	Medium (depends on secure setup)	Very high (uses PKI)

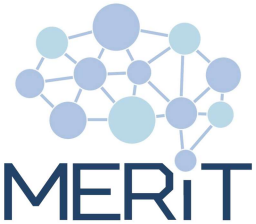
Authentication Methods: SCRAM, LDAP, X.509

Feature	SCRAM	LDAP	X.509
Hashing	Yes (SHA-1, SHA-256, etc.)	Optional, depends on implementation	Not directly applicable
Storage	Salt + hash	Passwords or LDAP objects	Digital certificates
Requires PKI	No	No	Yes
Mutual Authentication	No (unless via external protocols)	Limited	Yes (client and server)

Authentication Methods: SCRAM, LDAP, X.509

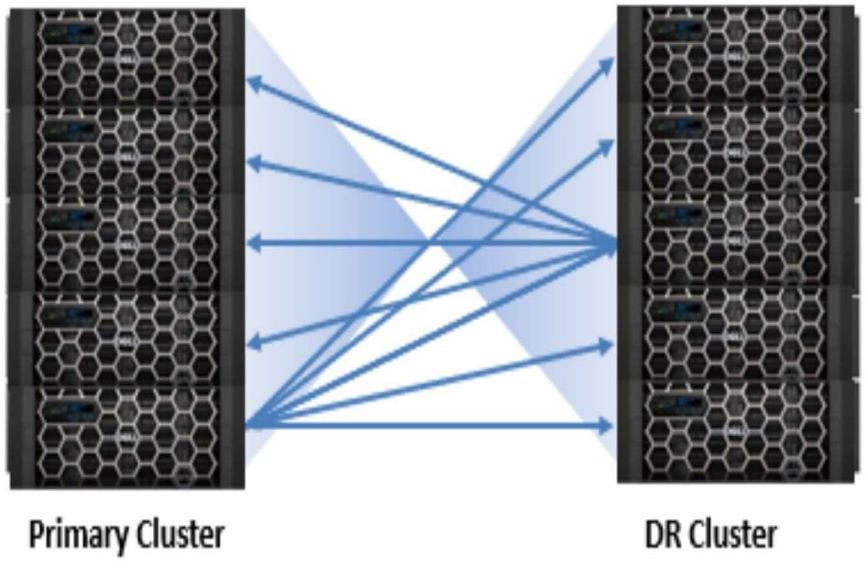
Method	Advantages	Disadvantages
SCRAM	Secure, modern, avoids password exposure	Does not directly support multifactor authentication
LDAP	Scalable, centralized, easy to integrate	Vulnerable if not using LDAPS or strong password policies
X.509	Very secure, ideal for high-trust environments	Complex to implement and manage (PKI, revocation, etc.)

Authentication Methods: SCRAM, LDAP, X.509



Use Case	SCRAM	LDAP	X.509
Database access	✅	✅	❌
Enterprise applications	❌	✅	✅ (with SSO)
SSO environments	❌	✅	✅
TLS/HTTPS authentication	❌	❌	✅

Replication Security and Cluster Hardening



Securing replication channels and hardening cluster configurations are vital for maintaining data integrity and availability in open-source databases. These measures protect against unauthorized access and potential data breaches.

Encrypted Replication Channels

Use SSL/TLS for PostgreSQL, MariaDB, and MongoDB to encrypt data during replication.

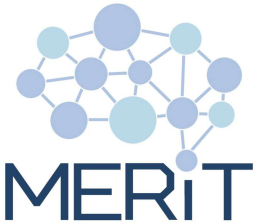
Authentication for Replication Users

Require authentication for all replication users to prevent unauthorized access.

Access Controls on Replication Ports

Enable strict access controls on replication ports to limit exposure.

Replication Security

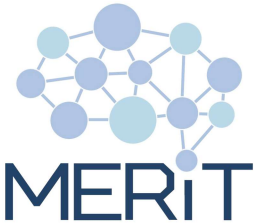


Replication security involves **protecting data as it is duplicated and synchronized across multiple database** instances or nodes. The goal is to ensure confidentiality, integrity, and authenticity during replication processes. Key practices include:

- ✔ **Encrypting replication traffic using TLS/SSL to prevent eavesdropping or tampering.**
- ✔ **Authenticating replication nodes to ensure only trusted servers participate in the replication.**
- ✔ **Restricting access to replication ports and services through firewall rules or IP whitelisting.**
- ✔ **Monitoring and logging replication activity to detect anomalies or unauthorized attempts.**

Why it matters:
Without proper security, replicated data can be intercepted, altered, or even lead to privilege escalation within the cluster.

Cluster Hardening



Cluster hardening is the process of **securing all nodes in a database cluster against potential threats** by reducing the attack surface and enforcing security best practices. This typically includes:

- **Disabling unused services and ports** on each node.
- **Applying the principle of least privilege** to users and services.
- **Regularly patching and updating** all nodes to fix known vulnerabilities.
- **Enabling strict access controls**, such as RBAC (Role-Based Access Control).
- **Auditing and monitoring** for suspicious behavior across the cluster.
- **Securing internal communication** between nodes with encryption and authentication.

Why it matters:
Clusters are attractive targets for attackers due to their distributed nature. Hardening reduces the risk of internal and external compromise, improving overall database resilience.

SUMMARY

Modern open-source databases, such as PostgreSQL, MySQL, and MongoDB, offer robust and evolving security features to meet enterprise demands:

- Authentication Options: Support for SCRAM, LDAP, X.509, and Single Sign-On (SSO).
- Encryption: Transparent data encryption, client/server SSL/TLS, and encrypted backups.
- Access Control: Role-based access, fine-grained permissions, and policy-based security.
- Audit & Monitoring: Native logging, extensions for audit trails, and integration with SIEM tools.
- Data Protection: Techniques like data masking, obfuscation, and row-level security.
- Cluster & Replication Security: Secure replication, node authentication, and hardened configurations.

These features make open-source databases viable for secure deployment in production environments, even for critical and regulated workloads.