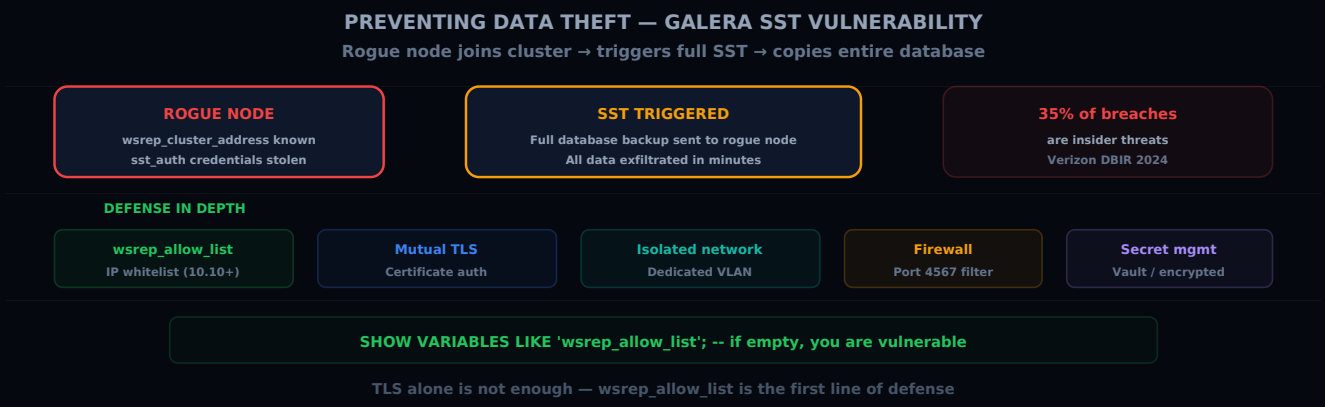
Preventing Data Theft: Galera Edition

Sylvain ARBAUDIE · March 12, 2025

GALERA MARIADB SECURITY SST



The Nightmare Scenario

Imagine: an attacker configures a MariaDB server with the right wsrep parameters, knows the Galera cluster address and SST password. They join the cluster. Galera detects a new node without data and triggers a **State Snapshot Transfer (SST)** — a complete transfer of all cluster data to the attacker's node.

Within minutes (or hours depending on database size), the attacker owns a full copy of your database. No SQL injection, no application vulnerability exploitation. Just a cluster JOIN with the right credentials.

This is not science fiction. According to the 2024 Verizon Data Breach Investigations Report, **35% of data breaches involve insider threats** — employees, contractors, or people with legitimate infrastructure access.

How SST Works

The State Snapshot Transfer is the mechanism by which Galera initializes a new node. When a node joins the cluster without data (or with data too old for an incremental IST), the cluster triggers an SST:

1. A donor node (existing cluster member) is selected

2. The donor performs a full backup (via mariabackup, rsync, or mysqldump)
3. The backup is sent to the joining node over the network
4. The joining node restores the backup and joins the cluster

The problem: **by default, any node with the right cluster information can trigger an SST**. There is no whitelist, no identity verification of the joining node.

Minimal Configuration for an Attack

What an attacker needs:

```
[mysqld]
wsrep_cluster_address = gcomm://10.0.1.10,10.0.1.11,10.0.1.12
wsrep_sst_method = mariabackup
wsrep_sst_auth = sst_user:sst_password
```

Three pieces of information: the cluster address, SST method, and SST credentials. In many organizations, this information is stored in unencrypted configuration files, plaintext Ansible playbooks, or private Git repositories.

Why TLS Is Not Enough

"But we use TLS for Galera traffic!" — this is a common objection. And it is insufficient.

TLS encrypts traffic between nodes, but it does not necessarily verify the identity of the joining node. Even with TLS, if the attacker possesses a certificate signed by the same CA (often the case in internal deployments with an enterprise PKI), they can join the cluster.

Moreover, many Galera deployments do not use mutual certificate verification (mutual TLS). They enable TLS for encryption but not for authentication.

The Solution: wsrep_allow_list

Since MariaDB 10.10, the `wsrep_allow_list` variable offers an IP whitelist mechanism for nodes allowed to join the cluster:

```
[mysqld]
wsrep_allow_list = 10.0.1.10,10.0.1.11,10.0.1.12
```

Only nodes whose IP address is in the list can join. A node with an unlisted IP will be rejected, even with valid SST credentials and TLS certificates.

It is simple, effective, and the first line of defense every Galera cluster should have.

Defense in Depth

Galera cluster security does not rely on a single mechanism. Here is a defense-in-depth approach:

1. wsrep_allow_list — Network Filtering

Restrict IPs allowed to join the cluster.

2. Mutual TLS — Node Authentication

Each node must present a certificate signed by the cluster's CA.

3. Isolated Network — Segmentation

Galera traffic (ports 4567, 4568, 4444) should flow on a dedicated network, isolated from application and management networks.

4. Firewall — Port Filtering

```
# iptables: only allow cluster IPs on Galera ports
iptables -A INPUT -p tcp -s 10.0.1.10 --dport 4567 -j ACCEPT
iptables -A INPUT -p tcp -s 10.0.1.11 --dport 4567 -j ACCEPT
iptables -A INPUT -p tcp -s 10.0.1.12 --dport 4567 -j ACCEPT
iptables -A INPUT -p tcp --dport 4567 -j DROP
```

5. Encrypted SST Credentials

Never store SST passwords in plaintext. Use secret managers (Vault, AWS Secrets Manager) or at minimum configuration file encryption.

Audit Your Cluster

Check your Galera cluster's security posture right now:

```
SHOW VARIABLES LIKE 'wsrep_allow_list';
SHOW VARIABLES LIKE 'wsrep_provider_options';
SELECT * FROM information_schema.WSREP_MEMBERSHIP;
```

If `wsrep_allow_list` is empty, your cluster is vulnerable. Configure it immediately.

Conclusion

The Galera SST vulnerability is an underestimated attack vector. An unauthorized node can obtain a complete copy of your database simply by joining the cluster. The fix is simple: `wsrep_allow_list` + mutual TLS + isolated network + firewall.

35% of data breaches are insider threats. Is your Galera cluster protected?

This article was originally published on Medium.