2026

SipLive SBC Manager

High Availability Setup & Configuration

2026 — Version 12.14

Contents

1 Overview

SipLive SBC Manager supports a two-node active-active High Availability cluster using MariaDB Galera replication. In this configuration:

- Both SBC nodes independently run Asterisk and serve live call traffic.
- Both nodes share a common database via Galera cluster synchronous replication.
- Configuration changes made on either node are immediately visible to the other.
- If one node fails, the other continues serving calls without manual intervention.
- All routing, trunk, DID, and policy data is kept in sync at the database level.

This guide covers the prerequisites, networking setup, configuration steps, operational procedures, and troubleshooting.

1.1 HA Model

- **Active-Active:** Both nodes accept and process calls concurrently.
- **Shared Database:** MariaDB Galera ensures all configuration changes are applied to both nodes synchronously.
- **Independent Asterisk:** Each node runs its own Asterisk process, decoupled from the other's uptime.
- **2-Node Cluster (No Arbitrator):** With only two nodes, a node that loses contact with its peer reports `non-Primary` until the peer is recovered or the node is manually bootstrapped as a standalone primary.

1.2 When to Use HA

Deploy HA when:

- Call availability is mission-critical and a single-node failure must not result in service loss.
- You need live maintenance capability: upgrade one node while the other continues serving calls.
- Your SIP carriers require multi-site redundancy.
- You want built-in database protection: database changes are replicated synchronously.

Single-node deployments are sufficient for testing, small deployments, or scenarios where occasional downtime is acceptable.

---

2 Architecture and Prerequisites

2.1 Network Requirements

An HA cluster requires at least **three separate networks**:

1. **Management Network:** Private/firewalled network for database replication and inter-node management. Example: `10.0.0.0/24`.
2. **SIP Network:** Public or carrier-facing network where Asterisk listens for SIP traffic. Example: `203.0.113.0/24`.
3. **Web Network (optional):** Private network for web GUI access and administrative connections. May be the same as Management.

Node	Management IP	SIP IP	Web URL
SBC Node 1	`10.0.0.1`	`203.0.113.1`	`https://sbc-node1.example.com`
SBC Node 2	`10.0.0.2`	`203.0.113.2`	`https://sbc-node2.example.com`
Database Server (optional)	`10.0.0.3`	—	—

For a shared MariaDB server ("Database Node" role), it resides on the same management network but has no SIP IP or web interface.

2.2 Hardware and OS

Each SBC node must be:

- A dedicated Debian 12 VM or physical server.
- Running the same SipLive SBC Manager version as its peer.
- Installed via the standard ISO or `install-debian12.sh` script.
- Have stable network connectivity to the management network (replication is latency-sensitive).

The shared database server (if used) requires MariaDB 10.11+, sufficient disk space for database growth, and at least 4 GB RAM.

2.3 DNS and Certificates

For a production cluster:

- Each node must have a stable hostname: `sbc-node1.example.com`, `sbc-node2.example.com`.

- Each node must have a valid TLS certificate for its hostname.
- The inter-node communication (database replication) uses unencrypted TCP on the management network, so network isolation is required.

For testing/lab, self-signed certificates and static IPs are acceptable.

---

3 Installation and Initial Setup

3.1 Step 1: Install Both Nodes

Install each SBC node independently using the standard ISO or `install-debian12.sh` script. Do not enable HA or Galera during initial installation—this is configured after both nodes are online.

After installation:

- Verify each node's web interface is accessible.
- Confirm Asterisk is running: `asterisk -rx "module show like pjsip"` should list modules.
- Document each node's **Node ID** (shown during installation or in **Settings → Network**).

3.2 Step 2: Verify Network Connectivity

From each node, test connectivity to the other node's management IP:

```
ping 10.0.0.2    # From Node 1 to Node 2's management IP
```

Bidirectional pings must succeed with consistent latency (under 50 ms typical for same-site clusters).

3.3 Step 3: Configure Replication Settings

On **both nodes**, navigate to **Settings → Replication Setup** and enter the peer's configuration:

3.3.1 On SBC Node 1, enter Node 2's information:

- **Peer Management IP:** `10.0.0.2`
- **Peer Hostname:** `sbc-node-2`
- **Peer Web URL:** `https://sbc-node2.example.com` (or the peer's IP/hostname)

- **Peer Token:** A shared secret, e.g. `my-secure-peer-token-12345`. Use a long, random string.

3.3.2 On SBC Node 2, enter Node 1's information:

- **Peer Management IP:** `10.0.0.1`
- **Peer Hostname:** `sbc-node-1`
- **Peer Web URL:** `https://sbc-node1.example.com`
- **Peer Token:** The **same token** as Node 1.

Click **Apply Settings** on each node to save.

3.4 Step 4: Register Peer Node

On **one node only** (typically the first node), navigate to **Operations → High Availability** and scroll to **Register Peer Node**. Fill in:

- **Node ID:** `sbc-node-2` (must match the peer's actual node ID)
- **Display Name:** `SBC Node 2`
- **Role:** `sbc`
- **Management IP:** `10.0.0.2`
- **SIP IP:** `203.0.113.2`
- **Web URL:** `https://sbc-node2.example.com`

Click **Register node**. The peer node should appear in the **Hot Hot SBC Nodes** table.

---

4 Galera Replication Setup

After both nodes are registered and configured, set up Galera database replication.

4.1 Step 1: Bootstrap the Primary Node

On the first node (which will become the cluster's primary), navigate to **Operations → HA → Galera Replication** and click **Bootstrap as Primary**. A confirmation dialog appears:

> *"Bootstrap this node as a standalone primary? Only do this if you have confirmed the peer node is actually down - if the peer is still serving traffic this will cause a split-brain."*

Click **OK** to proceed. The node becomes the cluster's primary, and the status table shows:

- **Cluster Status:** `Primary`
- **Node State:** `Synced`
- **Cluster Size:** `1`

4.2 Step 2: Direct the Peer to Sync From Primary

On the **primary node**, click **Sync to Peer** to direct the peer to download the database and join the cluster:

> *"Sync the peer node from this node? The peer's local database service will restart and pull current data from this node."*

Click **OK**. The operation:

1. Sends a command to the peer node to initiate a state transfer.
2. The peer's MariaDB service restarts and begins pulling data (IST or SST).
3. Once complete, the peer's status in the Galera table updates.

Monitor the peer's Galera status on the primary node's **Galera Replication** page. Once the peer shows:

- **Cluster Status:** `Primary`
- **Node State:** `Synced`
- **Cluster Size:** `2`

The cluster is fully formed.

4.3 Step 3: Verify Cluster Health

On **either node**, navigate to **Operations → High Availability**. The **Active Active Cluster Health** section should show:

- **Cluster Mode:** `Primary_Synced`
- **Active SBC Nodes:** `2`
- **Cluster Status:** Both nodes listed as healthy and online.

Both nodes now receive and apply database changes synchronously. Configuration changes made on one node are immediately visible on the other.

---

5 Managing Cluster State

5.1 Stop and Start Replication

Replication can be paused on a single node without affecting the other's cluster status or call handling.

To stop replication (on one node):

Navigate to **Operations → HA → Galera Replication** and click **Stop Replication**. The node remains operational but stops syncing with its peer. The status shows:

- **Cluster Status:** `non-Primary`
- Changes made to the database on this node will **not** replicate to the peer.

This is useful for:

- Maintenance operations that require the node to not sync changes (advanced scenarios).
- Isolating a node to test changes before applying them.

To resume replication:

Click **Start Replication**. The node re-joins the cluster and syncs any pending changes.

5.2 Sync Databases Between Nodes

If the databases fall out of sync (e.g. due to a replication error), use the **Server Maintenance** buttons to resync:

5.2.1 Sync this node FROM its peer:

Click **Sync from Peer**. The node's MariaDB service restarts and pulls the peer's current database state. Use this when the local node's database is behind.

5.2.2 Sync the peer FROM this node:

Click **Sync to Peer**. The peer's MariaDB service restarts and pulls this node's database state. Use this when the peer is behind.

5.2.3 Force a full resync:

Under **Server Maintenance**, expand **Force full resync (advanced, destructive)**. The options here:

- **Sync from Peer (force full resync):** Completely wipes this node's database and rebuilds it from the peer's snapshot (Snapshot State Transfer, SST). This is slow and renders the node temporarily unavailable.
- **Sync to Peer (force full resync):** Completely wipes the peer's database and rebuilds it from this node's snapshot. The peer becomes temporarily unavailable.

Use full resync only when incremental resync (IST) has failed or the databases are severely corrupted.

5.3 Bootstrap a Standalone Primary

If the peer node is permanently down and you need this node to continue operating independently:

Navigate to **Galera Replication** and click **Bootstrap as Primary** (red button). The dialog warns:

> *"Bootstrap this node as a standalone primary? Only do this if you have confirmed the peer node is actually down - if the peer is still serving traffic this will cause a split-brain."*

Click **OK**. The node becomes a 1-node cluster, continues serving calls, and can accept new database changes. When the peer comes back, it is detected as a foreign cluster member and must resync to rejoin.

Warning: Do not bootstrap as primary if both nodes are still up and serving traffic independently—this causes a split-brain where changes on each node are lost.

---

6 Per-Node Configuration Overrides

In an HA cluster, both nodes share the same configuration database. Sometimes, a single trunk or route needs different settings on each node (e.g. max channels, media codecs, or header manipulations).

Per-node overrides allow you to specify different values for a specific trunk or route on a specific node without creating separate trunk or route records.

6.1 Creating an Override

Navigate to **Operations → High Availability** and scroll to the **Node Overrides** section. Click the + **Add Override** button:

- **Node:** Select the target node.
- **Target:** Select `trunk` or `route` and then the specific trunk or route name.
- **Overrides:** Enter a JSON map of field names and override values.

Example override (for Teams Direct Routing on Node 2):

```
{
  "max_channels": 40,
  "passthrough_headers": ["X-MS-Teams"]
}
```

Click **Save**. The override is created, and generated config is marked as needing Apply.

6.2 Applying Overrides

After creating or modifying overrides:

1. Navigate to **Operations → Apply Status**.
2. Review the changes in **Config Preview**.
3. Click **Apply** to push the generated config to the affected node(s).

The override takes effect on the next Asterisk reload.

6.3 Viewing Override Assignments

The **Node Overrides** table shows all active overrides: which node, which trunk/route, and what values are overridden.

6.4 Bulk Import

Use **Operations → High Availability → Import Overrides** to import many overrides from a CSV file:

```
node_id,scope,target,overrides
sbc-node-1,trunk,Teams-Direct,"{""max_channels"": 25}"
sbc-node-2,trunk,Teams-Direct,"{""max_channels"": 50}"
```

7 Node Maintenance and Evacuation

7.1 Maintenance Mode

A node in maintenance mode is excluded from call routing and apply queue targets. Use this when upgrading or rebooting a node.

To enter maintenance mode:

Navigate to **Operations → High Availability** and locate the node in the **Hot Hot SBC Nodes** table. Click **Edit** and check **Maintenance Mode**, then click **Update**.

Active calls on the node are allowed to complete; new calls are routed to the other node. After pending calls drain, safely reboot or update the node.

To exit maintenance mode:

Click **Edit** again and uncheck **Maintenance Mode**, then click **Update**. The node re-joins the active pool.

7.2 Node Evacuation Workflow

The node evacuation workflow safely drains a node of active calls before removing it from the cluster or taking it offline for extended maintenance.

To start evacuation:

Click the three-dot menu for the node and select **Start Evacuation**. The node is placed in maintenance mode, and active calls are allowed to complete (up to a timeout). The UI shows:

- A countdown timer until the evacuation completes.
- The number of remaining active calls.

To complete evacuation:

Once all calls have drained (or the timeout expires), click **Complete Evacuation**. The node is fully isolated from call routing and ready for shutdown or maintenance.

To cancel evacuation:

Click **Cancel Evacuation** to immediately exit maintenance mode and resume accepting calls.

---

8 Monitoring and Health Checks

8.1 Cluster Health Dashboard

The **High Availability** page shows real-time cluster status:

- **Cluster Mode:** `Primary_Synced` (healthy), `non-Primary` (node lost its peer), or `disconnected` (replication error).
- **Active SBC Nodes:** Count of healthy, reachable nodes.
- **Drift Status:** Whether nodes' local Asterisk config files match the database-generated state.

Recommendations appear below the metrics if issues are detected.

8.2 Heartbeat Monitoring

The SBC Manager uses heartbeats to detect node failures. Each node sends a heartbeat to the database periodically (default every 30 seconds). A node is considered **stale** if its heartbeat is older than the configured threshold (typically 90 seconds).

A stale node is excluded from:

- Active call routing.
- Apply queue operations.
- Cluster recommendations.

If a stale node comes back online, it automatically resyncs and resumes participation.

8.3 Config Drift Detection

Each node's actual Asterisk include files are compared against the database-generated state. If drift is detected (e.g. a file was manually edited or a apply failed), a recommendation appears:

> *"Node <name> has local config drift. Review changes and reapply if intentional."*

Click the node's drift indicator to view details.

8.4 Operations Alerts

If alert delivery is configured, the system sends notifications for:

- Cluster mode changes (e.g. entering `non-Primary`).
- Node becoming stale or offline.
- Significant config drift.

- Replication errors.

Configure alert delivery in **Settings → Operations Alerts**.

---

9 Troubleshooting

9.1 Node Reports Non-Primary Cluster Status

Symptom: A node shows `non-Primary` in Galera status.

Cause: The node lost contact with its peer and cannot confirm it is part of a healthy cluster.

Resolution:

1. Check the peer node's **Galera Replication** page to confirm it is online and in `Primary` state.
2. If the peer is down, bootstrap the `non-Primary` node as primary: **Galera Replication → Bootstrap as Primary**.
3. If both nodes are showing `non-Primary`, the cluster has split. Bootstrap the node you want to keep as primary, then direct the other to sync from it.

9.2 Nodes Report Different Cluster State

Symptom: Node 1 shows `Primary_Synced`, Node 2 shows `non-Primary`.

Cause: Network partition between nodes, or Node 2 lost replication.

Resolution:

1. SSH to Node 2 and check network connectivity to Node 1's management IP: `ping 10.0.0.1`.
2. If network is OK, on Node 2 click **Galera Replication → Sync from Peer**.
3. Monitor Node 2's Galera status; it should transition to `Primary_Synced` once sync completes.

9.3 Cannot Reach Peer Web Interface

Symptom: Node 1 shows the peer as unreachable; the Galera Replication page says "Peer unreachable."

Cause: Network issue, peer is down, or peer token is wrong.

Resolution:

1. Check physical/firewall connectivity: SSH to Node 1 and ping the peer's management IP.
2. Verify the peer's web service is running: Navigate to the peer's web URL from a browser.
3. Confirm the peer token is identical on both nodes (**Settings → Replication Setup**).
4. Check the peer's firewall allows TCP on the inter-node communication ports (default 4567 for Galera).

9.4 Database Replication Slow or Lagging

Symptom: Changes made on Node 1 take several seconds to appear on Node 2.

Cause: High latency on management network, or heavy database load during large apply operations.

Mitigation:

1. Check network latency: `ping` between nodes should show consistent < 50 ms round-trip.
2. Verify management network is not congested; consider dedicated network links if possible.
3. Large apply operations (importing 10,000 DIDs) temporarily increase replication lag. This is normal; allow the operation to complete.

9.5 One Node's Asterisk Config Out of Sync After Apply

Symptom: Apply succeeded on Node 1 but config is not live on Node 2's Asterisk process.

Cause: Network issue during apply, or apply operation incomplete on Node 2.

Resolution:

1. Navigate to **Operations → Apply Status** and click **View Details** for the apply operation.
2. Check the per-node apply history to see if Node 2 shows success or failure.
3. If failed, the error reason is displayed. Fix the issue and re-apply.
4. If no apply history on Node 2, manually queue an apply: **Operations → Apply Status → Queue Apply (all nodes)**.

9.6 Cluster Size Shows 1 Even Though Both Nodes Are Up

Symptom: The Galera status table shows `Cluster Size: 1` for both nodes.

Cause: Cluster formation incomplete, or replication not yet bootstrapped.

Resolution:

1. On the primary node, click **Sync to Peer** to trigger the peer to join.
2. Monitor the peer's Galera status; once it shows `Cluster Size: 2`, the cluster is formed.
3. If it remains at size 1 after 30 seconds, check the peer's node logs: Navigate to peer's **Operations → Diagnostics → Recent Logs** and search for Galera errors.

---

10 Best Practices

10.1 Backup Both Nodes

Configure S3 remote backup storage with different prefixes for each node (e.g. `sbc-node1/` and `sbc-node2/`). This ensures both nodes' backups are available offsite even if one node is down.

10.2 Test Failover Regularly

Periodically put one node in maintenance mode and verify the other node handles the traffic. This validates that:

- Calls route correctly to the active node.
- No manual intervention is needed.
- Your alerting detects the down node promptly.

10.3 Use Different Hostnames for Each Node

Hostnames should be unique: `sbc-node-1.example.com` and `sbc-node-2.example.com`. This prevents confusion when reviewing logs and alerts.

10.4 Avoid Network Asymmetry

Ensure both nodes have:

- Similar network latency to carriers and endpoints (within 10 ms of each other).
- Sufficient bandwidth for concurrent call traffic (each node should independently handle peak load).
- Isolated management networks to prevent replication interference.

10.5 Monitor Replication Lag

Check **Operations → High Availability** frequently to confirm both nodes are in `Primary_Synced` state. If a node enters `non-Primary`, investigate immediately.

10.6 Coordinate Maintenance

When updating or rebooting a node:

1. Place the node in **Maintenance Mode**.
2. Allow calls to drain (watch the active-calls count on the **HA** page).
3. Perform maintenance (upgrade, reboot, etc.).
4. Exit **Maintenance Mode** once the node is back online.
5. Verify cluster health before resuming full traffic.

---

11 Permissions Reference

Permission	Allows
`apply.run`	Manage HA nodes, start/stop replication, create overrides, evacuate nodes, queue applies
`dashboard.view`	View HA page and cluster health

Users without `apply.run` can view the HA dashboard but cannot perform any management actions. Administrators have both permissions by default.

---

12 Related Sections

- **Backup & Restore Guide:** Backup and restore procedures in an HA cluster.
- **Administrator Guide:** General system administration tasks.
- **API Reference:** REST/SOAP operations for HA automation.