

Store and Forward Module

The [Store and Forward](#) module lets a designated node buffer messages for clients that were offline when a message was sent. When the offline client reconnects to the mesh, it can request the message history and receive messages it missed.

Important caveat: Store and Forward is **best-effort, not a delivery guarantee**. The server only buffers traffic it personally received over RF; clients only get the history they explicitly request while in range of the server; and the buffer is volatile (held in RAM and lost on reboot or power loss). Do not rely on it to guarantee delivery of critical messages.

How Store and Forward Works

1. A node configured as a Store and Forward **server** (typically a fixed repeater with reliable power) listens to all channel traffic and stores messages in the device's PSRAM (volatile RAM, not flash). The buffer is lost on reboot or power loss.
2. The server periodically sends a **heartbeat** advertising itself to the mesh.
3. A client that was offline requests history from the Store and Forward server (on Android, by sending the text command `SF` to the server; on the Apple/connected app this happens automatically).
4. The server replays stored messages to the requesting client.

Enabling on a Server Node (Repeater)

Configure the node that will store messages. Note that only PSRAM-equipped ESP32 boards (for example T-Beam v1.0+, T3S3) can act as a server — nRF52840 devices cannot:

```
meshtastic --set store_forward.enabled true
meshtastic --set store_forward.is_server true
meshtastic --set store_forward.records 0
meshtastic --set store_forward.history_return_max 25
```

`records` sets the maximum number of messages to store (circular buffer; oldest messages are overwritten). The default is **0**, which auto-allocates roughly two-thirds of the device's PSRAM (about 11,000 records); set a specific number to cap it. Capacity scales with the device's PSRAM, not its flash — only ESP32 boards with onboard PSRAM (e.g. T-Beam, T3S3) can run an S&F server, and nRF52840 boards cannot.

Enabling on a Client Node

A client generally does not need any special Store and Forward configuration beyond being on the same channel as the server. The `is_server` flag already defaults to false:

```
meshtastic --set store_forward.enabled true
meshtastic --set store_forward.is_server false
```

When an app connects to the server, history can be retrieved automatically. Over LoRa, an Android client requests history by sending the text command `SF` to the server node.

Limitations and Considerations

- **Only one Store and Forward server per channel is recommended.** Multiple servers on the same channel will each replay messages independently, causing duplicate deliveries.
- **Server node must be always-on.** A server that goes offline defeats the purpose. Use a mains or solar-powered repeater with reliable uptime.
- **Only text messages are stored and replayed.** Position reports and telemetry are not stored.
- **Buffer is volatile.** Messages are held in PSRAM, not flash, so the entire history is lost on reboot or power loss. There is no flash-wear concern because nothing is written to flash for this purpose.
- **Message delivery is best-effort.** If the client is out of radio range of the server, history request packets cannot reach the server. History retrieval over LoRa is also unavailable on the default public channel.

Best Deployment Practice

Deploy Store and Forward on your highest-uptime fixed node - typically the primary community repeater with mains or solar power, and one that has onboard PSRAM. This is the node most likely to have been online and collecting messages during the period a user was offline.

Revision #5

Created 2026-05-03 05:41:26 UTC by Mesh America Admin

Updated 2026-06-09 12:13:37 UTC by Mesh America Admin