

SRP Replication

IETF 117

Ted Lemon <elemo@apple.com>

Current Status

- Still at -00
- Expires in October
- A lot has changed, definitely needs an update

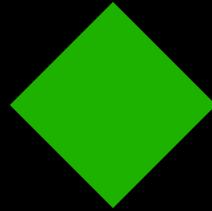
Brief review

- SRP Replication Goal
 - In the absence of stable infrastructure, use database replication to minimize (but not necessarily eliminate) database being lost due to user behavior
 - For IoT networks, e.g. Thread, having a nearby SRP server saves bandwidth/battery
- How it works
 - SRP client is the authority
 - SRP updates are signed
 - Most recent update is authoritative

Operational Experience

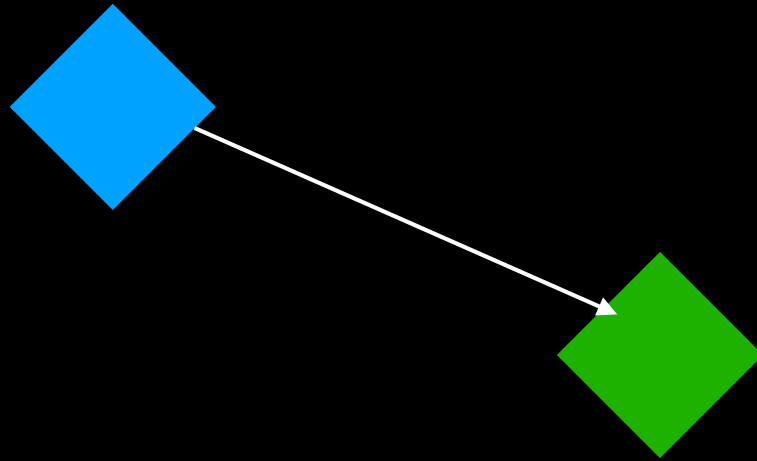
- Lots of conflicts, as mentioned previously
- Works pretty well at small scale
- Issues crop up as number of peers increases
- Issues crop up as number of registrations increases
- One very serious issue is that if no SRP peer enters normal operation, we have no SRP service
- A lot of work has gone into identifying and fixing corner cases that have this outcome
- It's not as simple as it seems

SRP Replication Startup



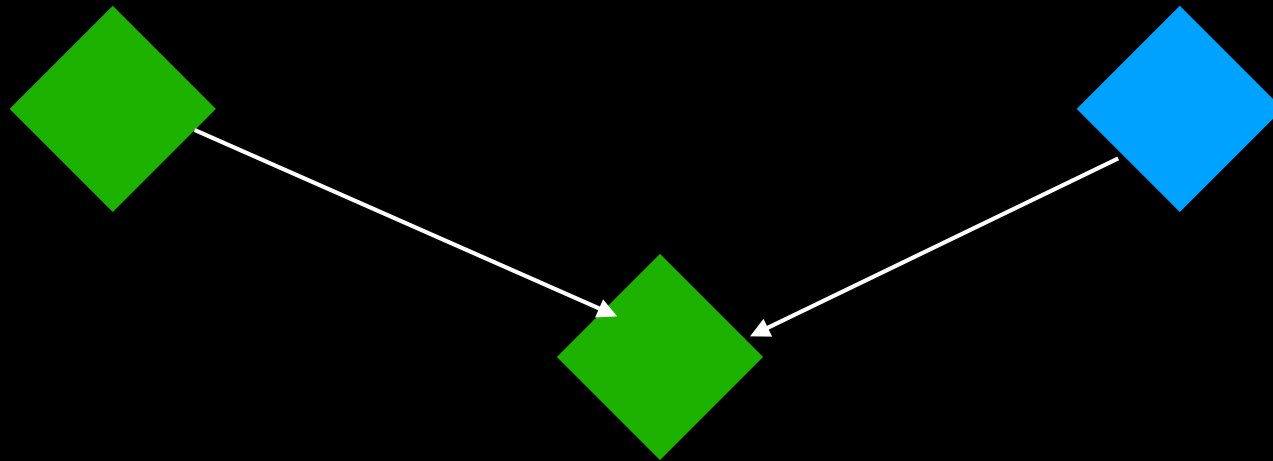
First Peer shows up, advertises via DNSSD

SRP Replication Startup



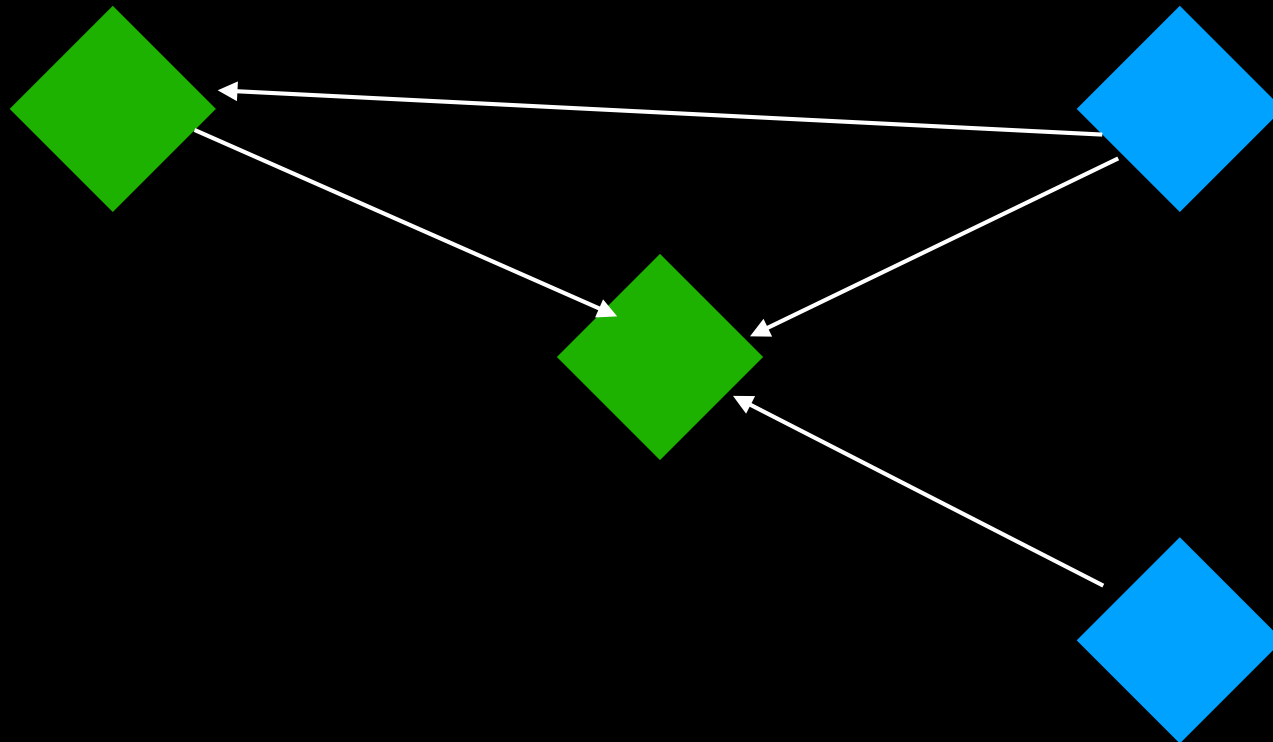
Second peer shows up, connects to first, begins to synchronize

SRP Replication topology



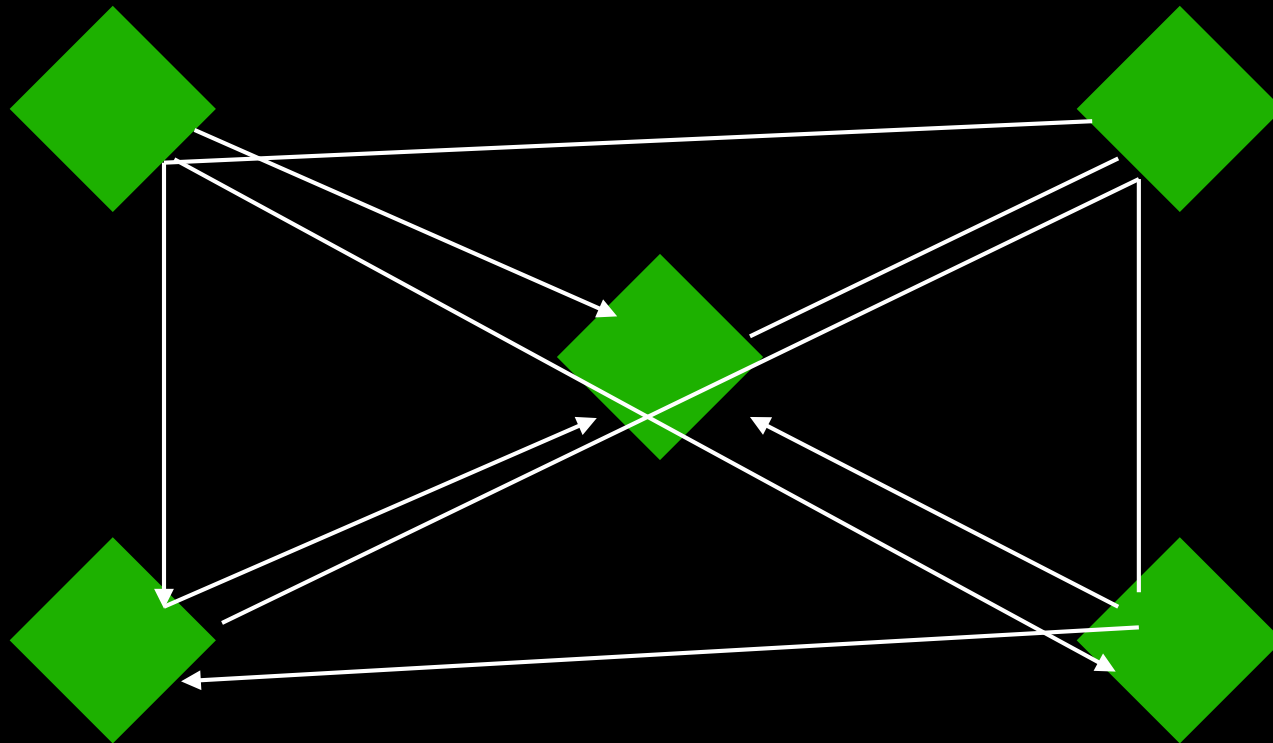
Second peer done synchronizing, meanwhile third shows up

SRP Replication process



Second peer done synchronizing with second
Third peer done synchronizing with first, starting with second
Fourth peer starts synchronizing with first

SRP Replication Topology

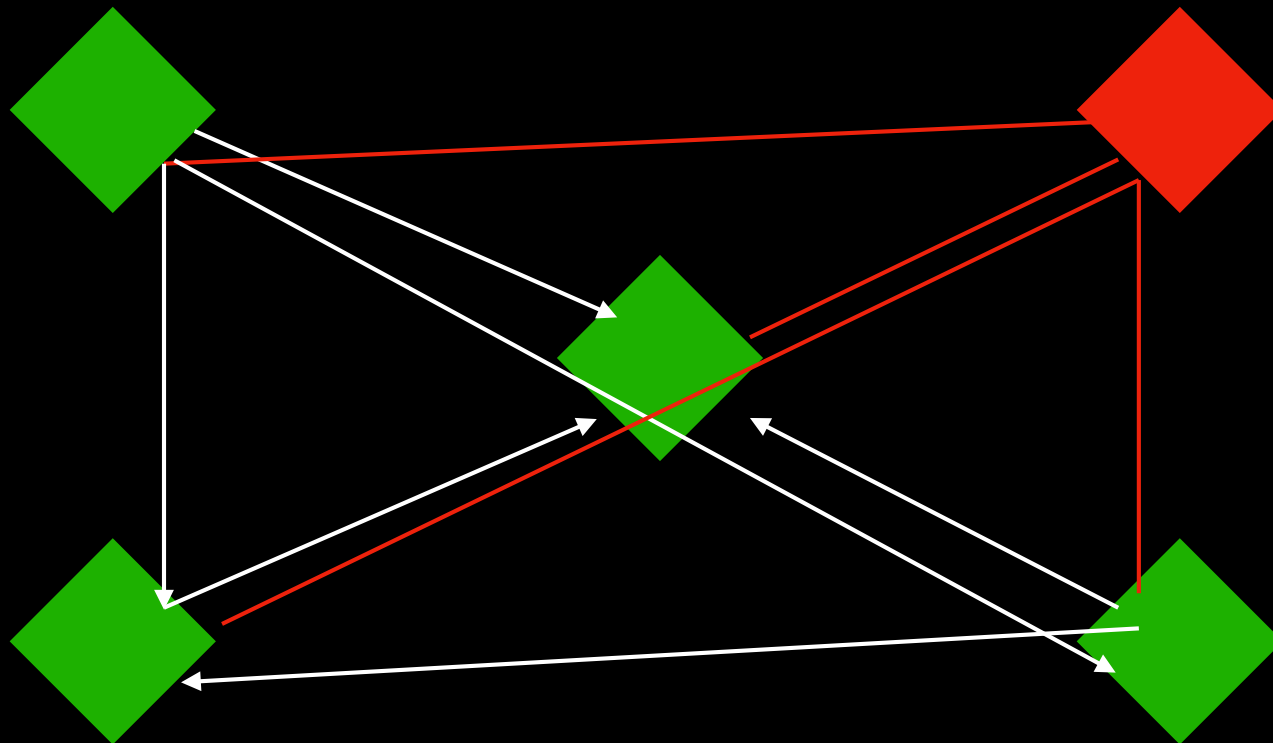


Ultimately all servers are in routine state
Total number of connections: $(n(n-1))/2$

Why so many connections?

- Authority strategy means that replication updates can never be forwarded
- This means that every peer has to have a connection to every other peer

SRP Replication process



**Red Server receives SRP update.
Updates each SRP replication peer
Updates are only on red connections**

Scaling

- Updates only happen when an SRP update happens
- Relatively low cost
- Peers must notice when peers have gone down
- This uses DNS Push Keepalive
- As number of servers grows, keepalive traffic increases
- e.g. 20 servers = 100 connections. One keepalive per 90 seconds means a little over one keepalive per second
- This is a lot of traffic for no work
- Plus, as mentioned earlier, each mDNS publisher per record increases mDNS traffic per record
- Thread currently limits SRP servers to five (10 connections)
- This is about seven keepalives per minute

How to limit

- Currently SRP does not limit connections
- Apple implementation
 - limits active peers to five
 - Standby peers are connected to active servers
- This isn't ideal
 - Still a lot of connections, but we scale keepalive traffic for standby peers
 - If an active peer goes down, we get a thundering herd: all the passive peers try to become active
 - (This is a bug)

Proposed approach to limiting SRP peer count

- Active peers publish in mDNS
- Numerically lowest partner ID is considered primary
- Peers that see mDNS advertisements register their SRP replication service with primary via SRP via DoT and then send SRP Replication Session DSO message
- If there are fewer than five peers and a peer registers, primary responds with Session Response, replication proceeds
- Otherwise, primary sends DSO Retry Delay message scaled to number of registered peers

Peer Priority

- Some peers are more equal than others. Consider:
 - Infrastructure server with stable storage (highest)
 - e.g. perhaps a CE router
 - Reasonably competent non-infrastructure stub router
 - e.g. a device like Apple TV with ethernet
 - Less competent non-infrastructure stub router
 - e.g. a device like HomePod Mini with wifi only
 - Last resort, only use if there's no alternative
 - Powered light bulb with WiFi and 802.15.4
- We should prioritize best devices that can act as peers when there is a choice
- This is somewhat Thread-specific, but I think it generalizes

Dealing with conflicts

- Current approach is adversarial rather than cooperative
 - mDNS assumes that it's the only communication channel
 - SRP replication is another communication channel
 - It doesn't make sense for SRP to agree, and then mDNS to disagree, about the same registration
 - Can we order SRP updates in such a way that we don't see conflicts?
 - Can we short-circuit conflict probing for the backup proxy use case without breaking other use cases?
 - In any case, we should decouple SRP and mDNS namespaces so that conflicts in mDNS don't affect the content of the SRP namespace (right?)

State Machines

- Per peer, for any SRP dataset, state machines for:
 - Monitoring each active peer
 - Monitoring operational state:
 - Routine (publishing)
 - Startup (synchronizing)
 - Idle (not participating)
 - Monitoring state of connection to each peer
 - Actual SRP protocol communication with each peer