A Decentralized Mapping System

Draft-farinacci-lisp-decent-01

IETF Montreal

July 2018

Dino Farinacci & Colin Cantrell

Problem Statement

- What if LISP xTR’s didn’t rely on a preconfigured map server?
- What if map servers could auto allocate to their own shards when they come online?
- What if abstraction layers on top of the OSI could improve key management and authorization in map servers?
- What if we could mitigate map server DDOS attacks through partial pre-image collisions of zero bits?

Shard the Map-Server?

- What if each xTR was a Map-Server allocated to a DHT shard
- What if each xTR could Map-Register to each xTR based on a deterministic modulus?
- What if map servers could provide redundancy for each other and remain distributed?

DDOS Protection

- We always have the needle in a haystack problem with DDOS attacks, what do we do if a central map server goes under DDOS attack, does this cripple the network?

- What if server load determines allocation in the DHT shards for map servers, and also acts to require requesting party to complete a proof of work by iterating a Nonce to find a value that meets a number of 0 bits (000111011011), making it require work for the requester if the server happens to be under heavy load.

- This could be a sign of a DDOS attack, especially if the map server shards are assigned to allocate more shards on > 30% resource utilization over moving average window

How to Shard the Mapping System

- A set of DNS A records can resolve initial Map Server(s) which can seed other known map servers

- The xTRs that are part of a mapping system resolve the first DNS records to obtain initial DNS seed, which then resolves to return it a list of known map servers

- Map-Registers are sent to the correct shard, allocated by deterministic assignment of modulus

An example of Shards

N mod 1

N mod 2

N mod 3

New allocation:
N mod 4

A new allocation N mod4

This would begin handling map Registers on mod 4 to begin cache Transition

DDoS Request Throttling

- What if map server requires an map register or map lookup per EID to throttle more computing resources asymmetrically?
- The requester would be required to compute thousands of hashes to find a given number of zero bits, while the map server would only be required to compute one hash
- If the hash cash was seen as invalid, the cost then reduces to calculating one hash and dropping the packet saving map server from CPU exhaustion attacks through ECDSA_verify or Memory Overflow attacks from filling up the Map Cache
- This would require shards to function, if the DDoS protection kicked in at a threshold of CPU usage over period of time for the map server.

Benefits

- xTRs only depend on each other - they do so already if they want to talk to each other
- No third-party trust or dependency exists
- Map-Request lookup has low latency
- Map Servers have redundancy and scale ability

Use-Cases

- Distributed Ledger Networks over LISP
- EID based indexing for distributed databases

Why Decentralized?

- Peer to Peer networks have proven high levels of robustness

- Always have fallbacks if a map server gets put under DDOS attack

- Distributed Ledger Technologies relies on the robust nature of peer to peer, to run LISP reliably on such a network would require a distributed topology

Questions/Reactions/Tomatoes?