Secure Resolution of End-Host Identifiers for Mobile Clients

Samu Varjonen

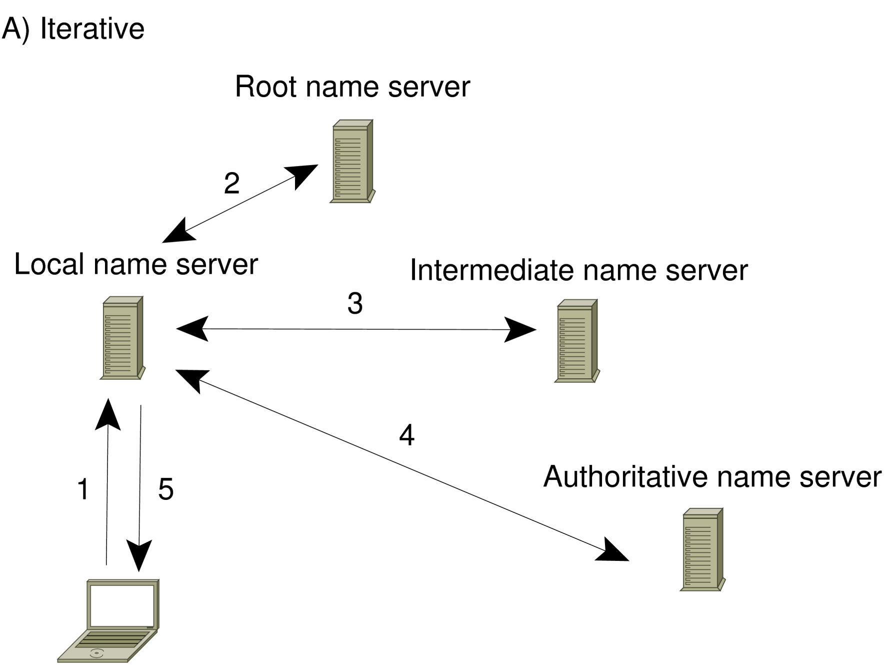
Agenda

- DNS
- DNS over DHT
- Motivation for new namespace
- Requirements
- System design
- Evaluation
- Conclusions

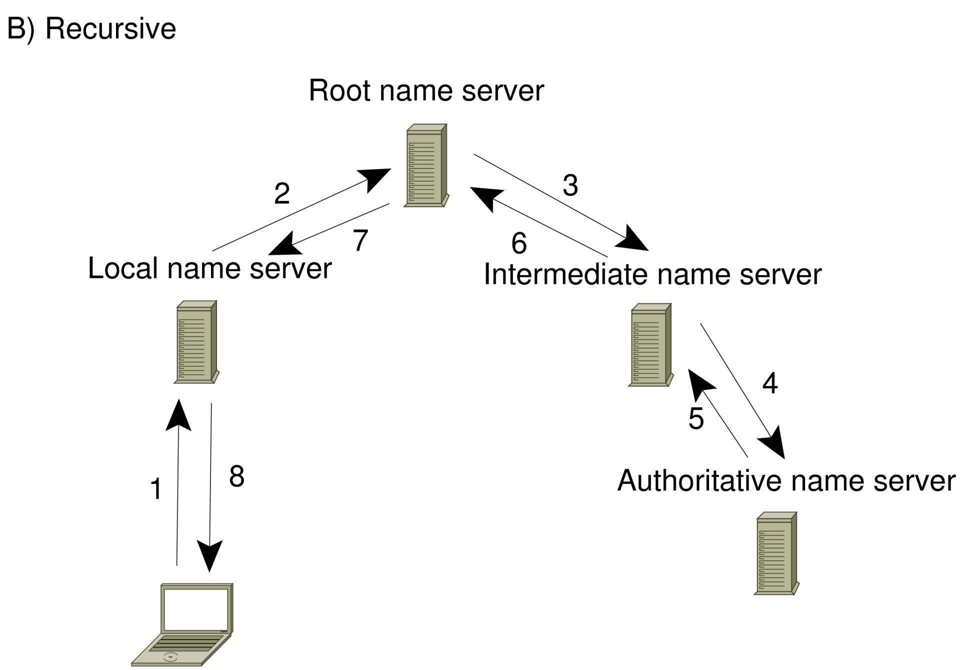
FQDN → IP

FQDN → EID → IP

DNS



A) Iterative
Root name server
2
Local name server
Intermediate name server
3
4
Authoritative name server
1
5



B) Recursive
Root name server
2
3
7
6
Local name server
Intermediate name server
4
5
Authoritative name server
1
8

DNS problems

- Administrative needs
  - Albitz & Liu state that most of the failed queries are caused by improperly configured servers
- Misc:
  - Missing and incorrect mappings
  - Typos in queries
  - Malformed queries
- Attack resilience
  - Spoofing the IP of a legit name server
  - Cache poisoning (no checks on the origin of RR)

DNSsec

- Public key cryptography and digital signatures
- Provides
    - Authentication for the origin
    - Integrity protection
- Zone enumeration
    - Queries to non-existent names result in NSEC records of the next existing name
    - Fix: 3NSEC record, hashed value of NSEC

DNS performance

- Uneven growth
  - Most of the new names go to popular domains
  - Popular domains have higher loads
  - But usually more available
    - Constant maintenance, better hardware, redundant network connections, only fraction of available servers are used
- Caches improves the performance but:
  - Uneven growth flattens the namespace
  - Use of short timeouts
  - Cache coherency

DNS over DHT

- Requirements:
  - Availability
  - Attack resilience
  - Lookup latency
  - Failure rate
  - Load distribution

Availability

- Malfunctioning or down server
  - Can separate clients from the network
  - Connections could be made but name resolution is not possible
  - Caches may help but may also cause problems
- DHTs:
  - Self-healing
  - Support replication

Attack resilience

- DoS attacks:
  - Flood the target with bogus work
  - Result, server cannot serve valid users
  - Can be thought as sub-category of availability
- Securing the stored data
  - More on later slides

The rest

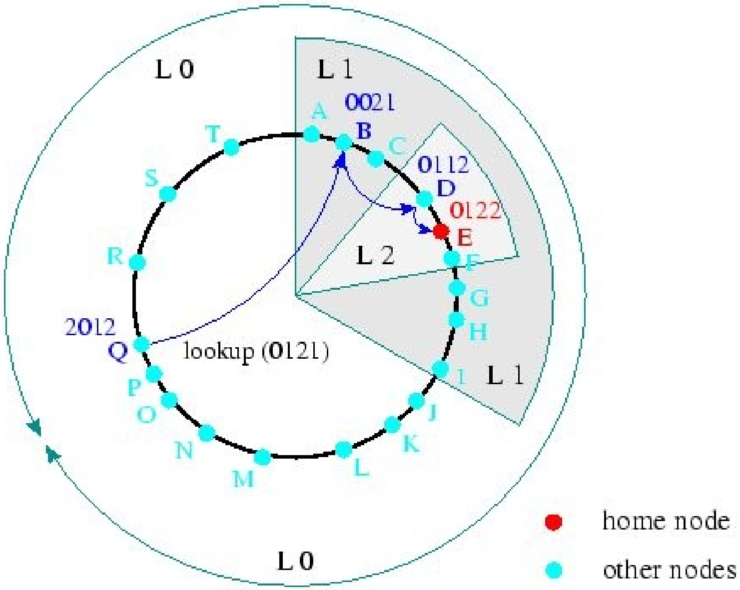
- Lookup latency
  - Also update latency
- Failure rate
  - People still make typos
  - But configuration can be minimized
- Load distribution
  - Better in DHTs

Two examples

- CoDoNS
- DDNS

CoDoNS

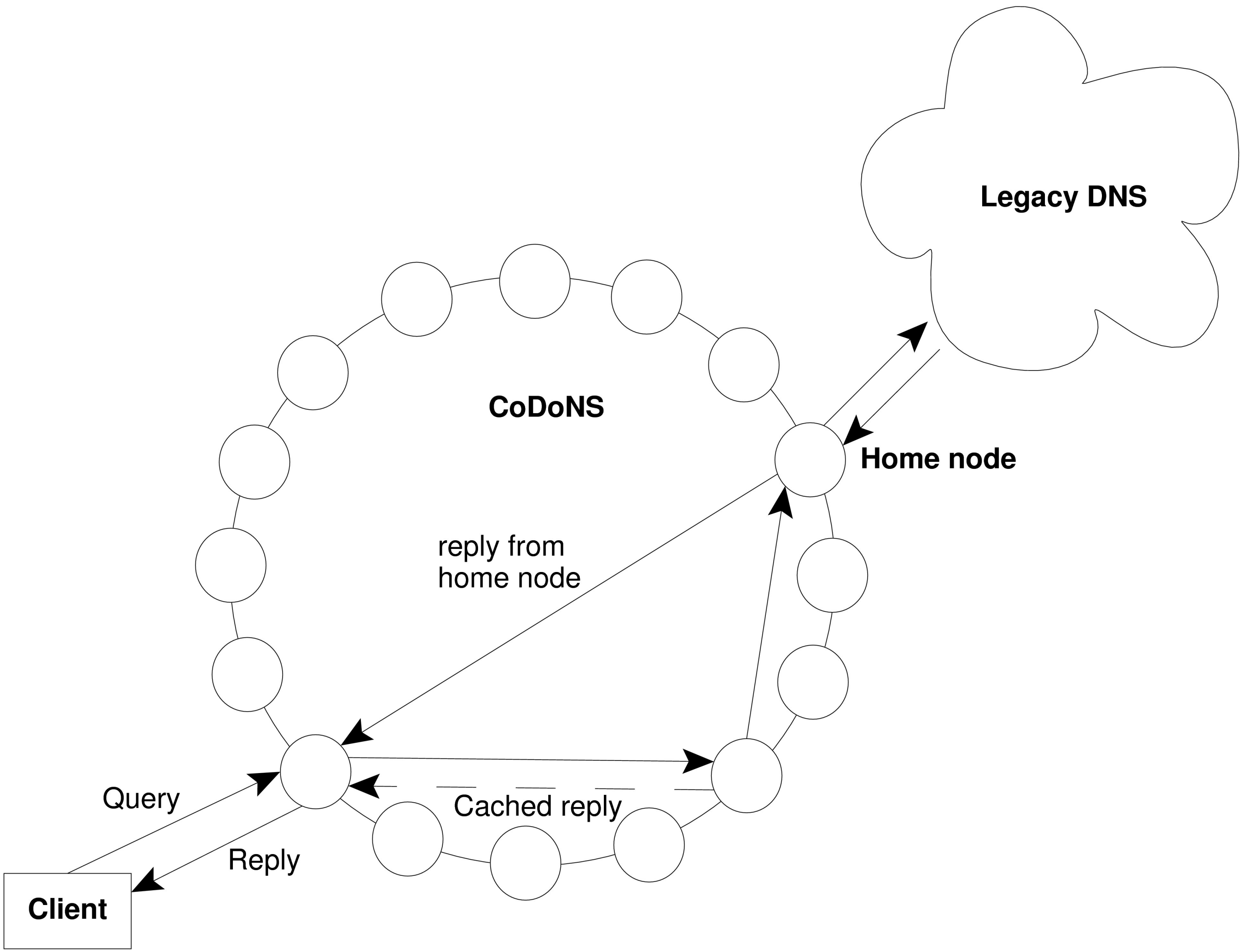
- Pastry using Beehive for replication
- Average query path will be reduced by one hop when an object is proactively replicated at all nodes logically preceding that node on all query paths
- On level 0 this is close to consistent hashing



CoDoNS

- Uses the same message format as DNS
- Is faster, more attack resilient but
  - If CoDoNS does not have the needed RRset, it queries DNS and caches the resulting RRset
  - Result bad latency

CoDoNS



Legacy DNS
CoDoNS
Home node
reply from
home node
Query
Cached reply
Reply
Client

DDNS

- Not to be mixed up with Dynamic DNS
- DDNS is DHash on top of Chord
- Upon insertation signatures are verified (DNSsec)
- Data is stored using sha-1(FQDN | resource type) as the key
- DHash is used to split the data into 14 block
  - IDA erasure codes
  - Retrieve any 7 blocks and the data can be reconstructed
  - Improved availability with minimal cost in extra storage and communication
- Chord determines the nodes that store the blocks

DDNS

- Seems a efficient replacement but
  - Is not able to support load balancing
    - I.E., load balancing for the service who's name was queried
    - E.G., google.com returns addresses from a pool in round robin
  - Needs authorative CA hierarchy to provide it with signatures for the DNSsec
    - Could use WoT approach suchs as PGP but ...

Motivation for new namespace

- IP has two roles
- The networks are dynamic and flexible
- ID/Locator split
- Name-to-identifier and identifier-to-locator
- DNS for frequent reads and occasional updates
- Mobile hosts need to update their records

Requirements #1: Flat namespace

- Name-to-identifier:
  - human-readable identifiers
  - Managed
  - Structured in a hierarchical way
- Identifier-to-locator:
  - Binary identifiers
  - Cryptographic identifiers
  - Little or no hierarchy

Basic reasoning behind the namespace

- The namespace should fully decouple the network layer from the higher layers
    - HIP is a waist between transport and network layer
- The names should have a fixed-length representation
- Should be affordable when used in protocols
    - computation and size
- Name collisions should be avoided as much as possible
- It must be possible to create names locally
    - anonymity
- The namespace should provide authentication services
- The names should be long-lived, but replaceable

The flat namespace

- Input := any bitstring
  Hash Input := Context ID | Input
  Hash := Hash_function(Hash Input)
  ORCHID := Prefix | Encode_100(Hash)
- Any bitstring is Host Identity, e.g., public key
- Context ID 0xF0EF F02F BFF4 3D0F E793 0C3C 6E61 74EA
- Hash SHA1
- Prefix 2001:10::/28
- Encode_100() take 100 middle bits
- Result is a Host Identity Tag
  2001:17:53ab:9ff1:3cba:15f:86d6:ea2e

Requirements #2: Rapid user-generated updates

- DNS would suffice for stationary nodes under administrative management
- Mobility needs short TTLs and disallowed caching

- Name-to-identifier in DNS
- Identifier-to-locator in a DHT

Requirements #3: Security

- Fraudulent behaviour in DNS:
  - Requires considerable effort
  - Usually limited to a sub-domain
  - DNSsec against spoofing
- Security in DHT (OpenDHT as example)
  - Open access for everyone and multiple values under each key
  - Flooding, poisoning, redirecting

Security problems for content in content agnostic DHTs

- E.G., OpenDHT stores one or several values under one key
  - Vulnerable to flooding
    - Malicious user stores much false or random information under the key
    - Effectively drowns the correct information
  - Vulnerable to index poisoning
  - E.G., malicious user uploads victims identifier under popular services redirecting the traffic from the popular services toward the victim
  -

Security problems for content in content agnostic DHTs

- Addition of signatures to the data
  - Signatures can be verified
  - Replay protection with sequence numbers
  - But can lead to extra work
    - Try to find the correct signature among the bogus ones returned from the system
- Content can be secured to a point in content agnostic DHTs and then the DHT has to take part in the security
  - Verify the ownership of the data that is modified
  - System signs/timestamps the data

Sybil attack against DHTs

- Malicious user in a peer-to-peer system can easily introduce a very large set of corrupt participants
  - “sybils”, from book title about a woman with dissociative identity disorder
  - In Chord malicious user could create identities so that it can populate the some nodes' finger table with it's sybils
  - Objective to be non-cooperative, flood, bogus work

Sybil attacks against DHTs

- What to do:
  - When queried return all nodes that we know instead of the closest one to the target
    - More control over the resolution but more space needed
    - Simple “trust profile”, e.g., # time the ID has been on a query path
  - Use of self-certifying IDs, possible to authenticate the target node and be certain it owns the ID
  - Add a cost to the join
  - Goverment/institution/ISP maintained DHTs
  - Who controls the identity creation?
    - Decentralized vs. Centralized

System design: Solutions

- The DHT is agnostic of the content
  - Hosts would have to implement security
  - Can be achieved with signatures over the mappings
  - Index poisoning / Flooding still a problem

- DHT enforces the correctness of mappings
  - DHT verifies the signatures before accepting
  - Requires replay protection

System Design: HIP

- HITs used in BEX checked against the HDRR
- SEQ into HDRR
- Authentication and basic DoS protection provided by HIP
- Only one modification to the DHT:
  - Check the equality of the HIT and the used key in the DHT

BEX + DoS protection

```
                      Initiator                     Responder

                         I1
                         ------------------------------->
                                                            Select precomputed R1
                                                            Remain stateless
                         R1: puzzle, key, sig
                        <--------------------------------
Check sig
Solve puzzles
                        I2: solution, {key}, sig
                        -------------------------------->
Compute D-H                                                 Check solution
                                                            Check sig
                        R2: sig,
                        <-------------------------------
Check sig                                                   Calculate session key

                        ESP
                        <------------------------------>
```

Evaluation: Feasibility

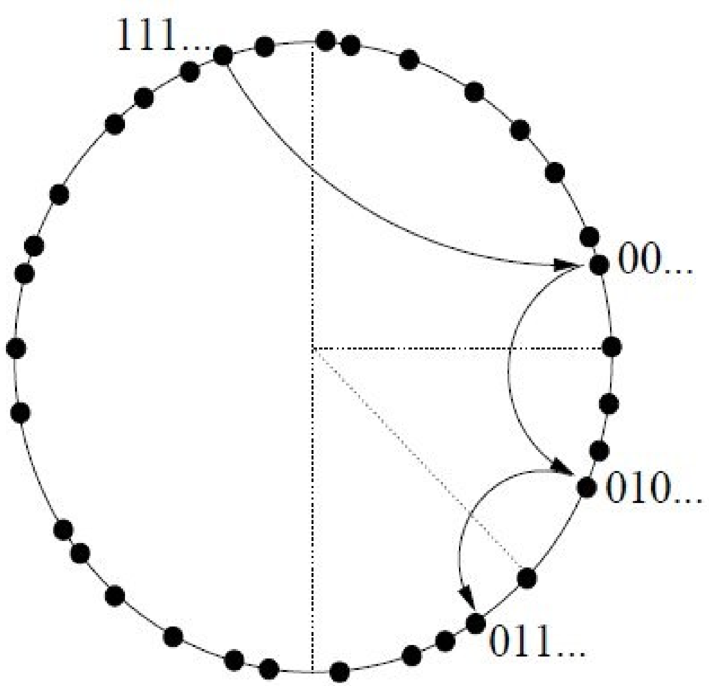
- Quad Core Intel Xeon 5130 2Ghz, 2 GB

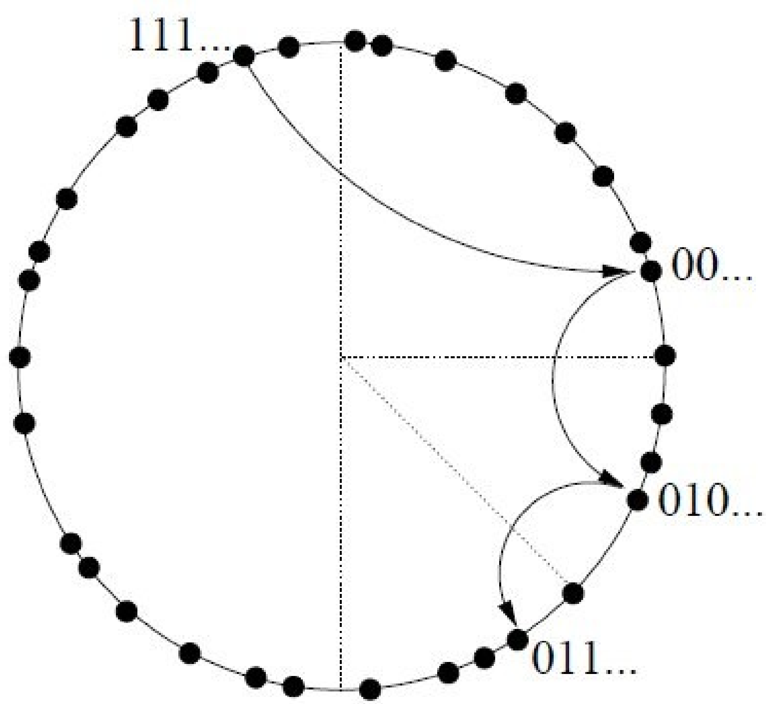
	operations per sec.	ms/operation
HMAC(MD5)	42267	0.02
SHA-1	30809	0.03
DSA signature	1887	0.53
DSA verify	1645	0.61
RSA signature	890	1.12
RSA verify	18502	0.05
DH key generation	51	19.64

- RSAs + RSAv + 2 * SHA-1 + DH ~= 20.8ms
- 150 node DHT could process
  28,800 updates/s

Bamboo-DHT is pastryish?

- Leaf set (dashed), Routing table (solid)
- Routing and lookups are the same



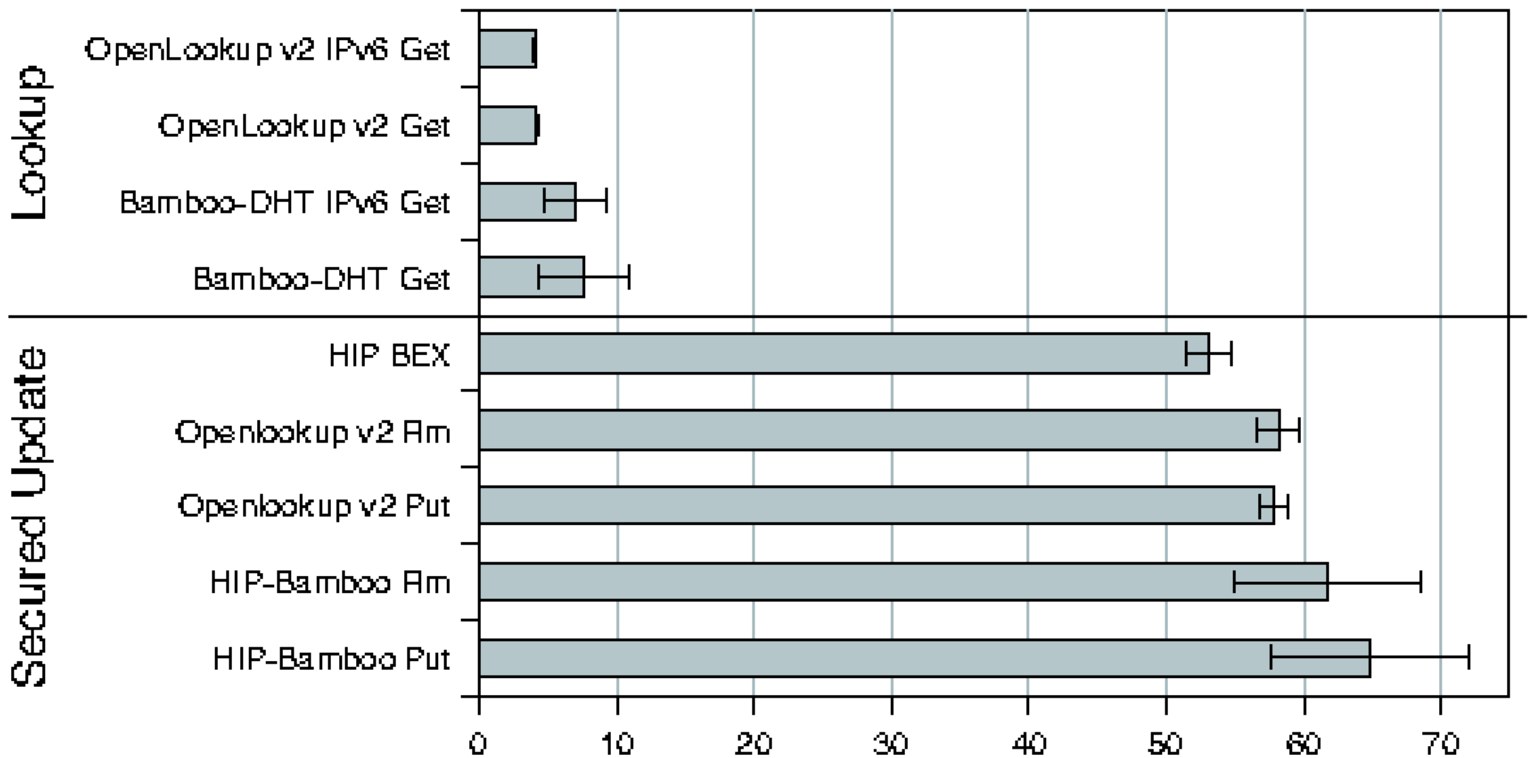


Bamboo-DHT is pastryish?

- Churn handling has the differences
  - Reactive versus periodic recovery from neighbor failure
    - Be cautious about declaring neighbors failed
    - Are you tricked into recovering a non-faulty node by network congestion
    - Lots of churn in large system do periodic recovery
    - Low churn reactive recovery
  - Proximity neighbor selection
    - Routing table can be sub-optimal but leafs are important so priority on the leafs
    - Discover one nearby node, then that node’s neighbors are probably also nearby

Evaluation: Delay

- HIPL and Client Intel Core 2 2Ghz, 2GB



Conclusions

- HIP integrates nicely to the system
- HIP introduces a notable delay (53 ms)
- Eliminates index poisoning and spoofing
- No additional administrative measures
- Resilient to DoS
- Bottomline: HIP enabled DHT based resolution
  - Is feasible
  - Acceptable performance
  - Considerably increased security

Thanks

- Any questions?
- samu.varjonen@helsinki.fi