CoDoNS: Replacing the DNS Hierarchy with Peers

Venugopalan Ramasubramanian (Rama)
Emin Gün Sirer

Computer Science Dept., Cornell University

Why change the DNS?

- DNS is largely successful
  - Two decades of operation
  - High scalability
- Requirements have increased
  - Constant availability
  - High performance
  - Security

DNS: Problems

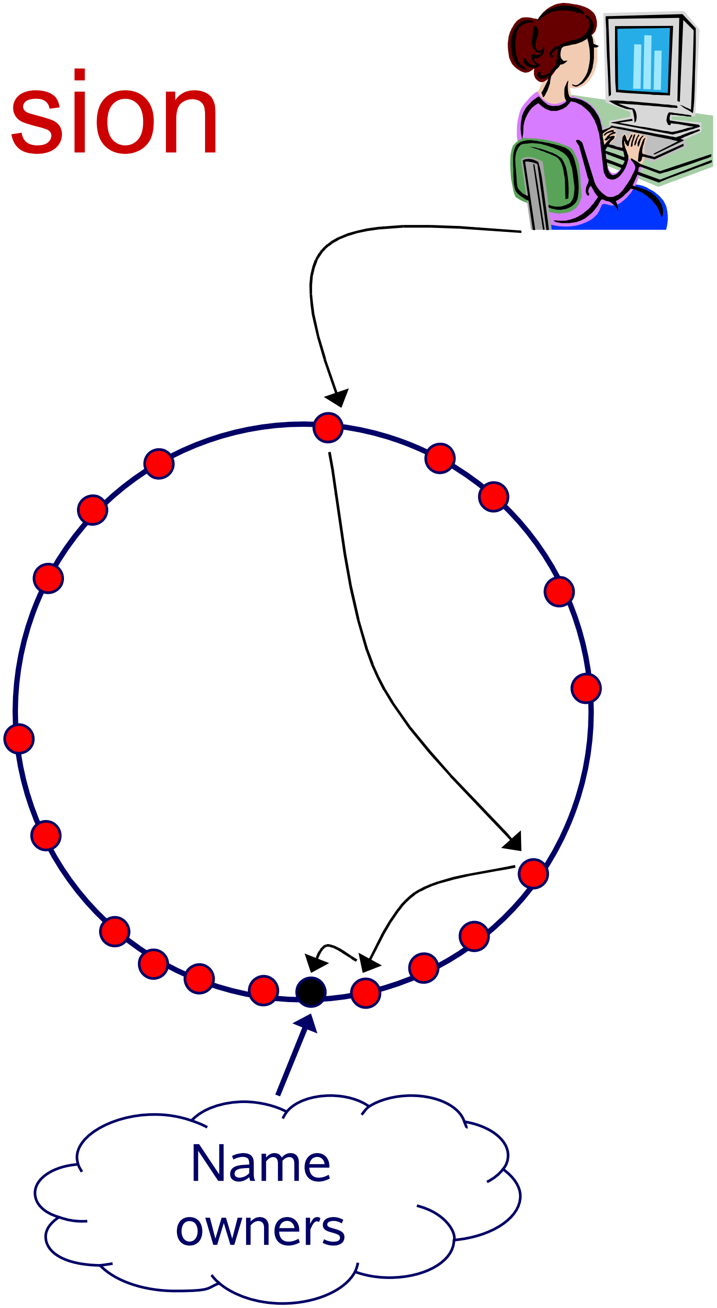
- Poor availability
  - 80% of domain names bottle-necked at 2 servers
  - 30% of domain names bottle-necked at 1 gateway
- High latencies
  - Long tail in response time
  - Stale bindings remain for a long time
- Vulnerable to attacks
  - Cache poisoning, transitive trust
  - Denial of Service (DoS)

Insight and Solution

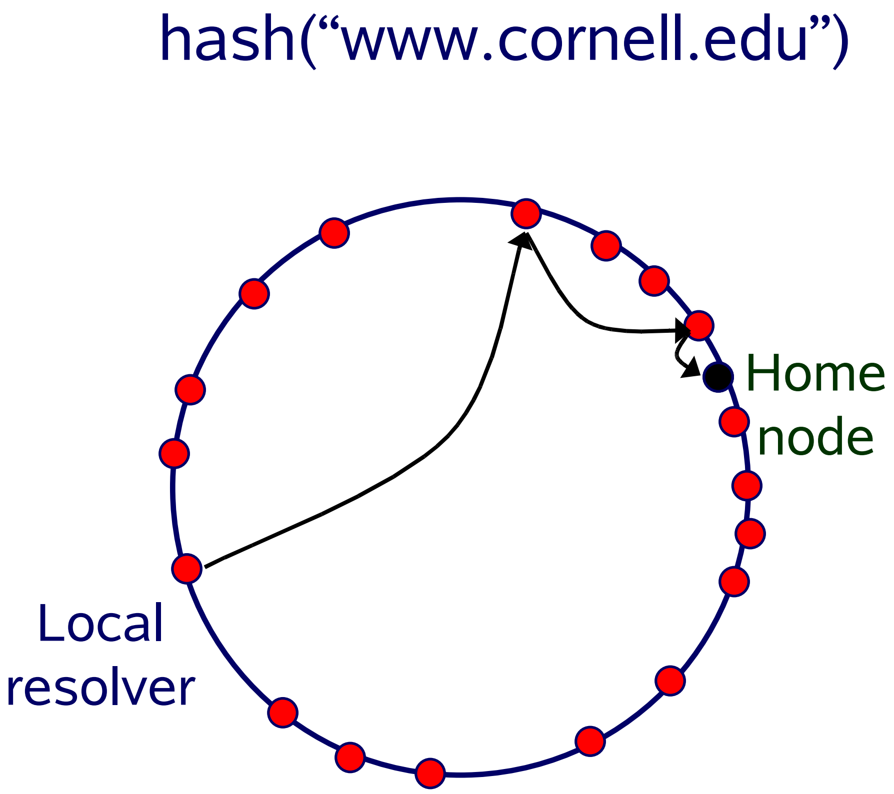
- ✗ Hierarchical, delegation-based name resolution
- ✓ Separate namespace management from name resolution
- Hierarchical, decentralized namespace
  - Scalable, easy to manage
- Efficient name resolution service
  - High availability, performance, and security

CoDoNS: Vision

- Peer-to-peer DNS
- Composed of DNS resolvers and name servers
- Self-certifying data
  - DNSSEC

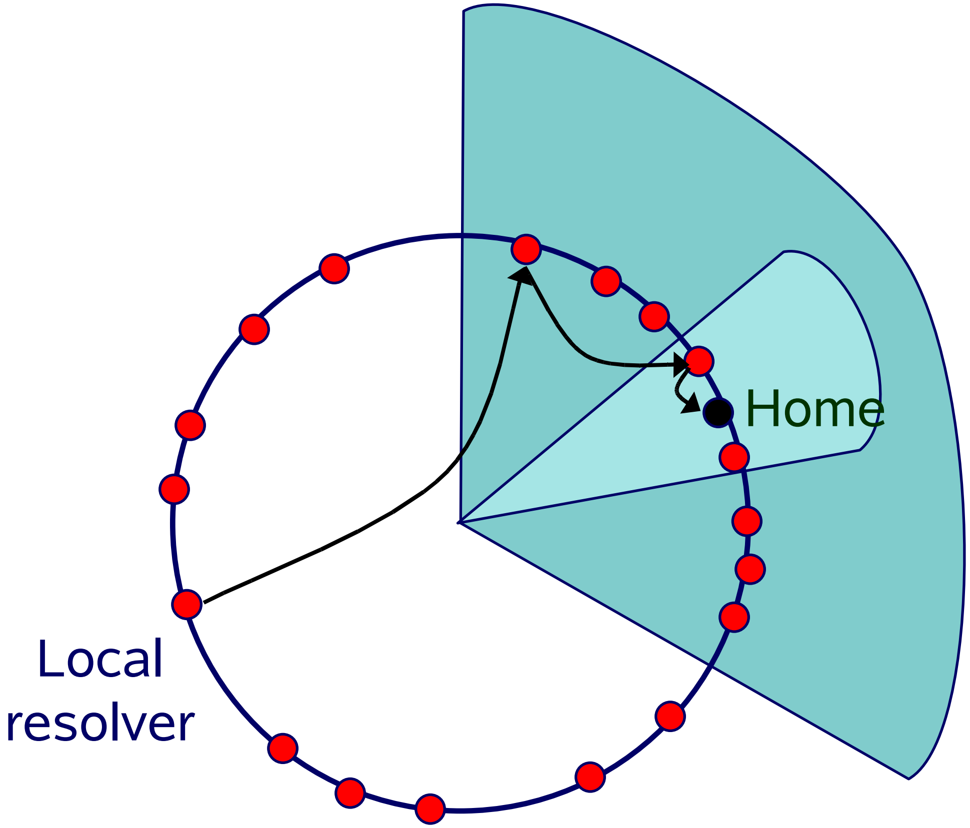


CoDoNS: Structured Overlays



- Self-organization
  - Failure resilience
  - Scalability
- Well-defined structure
  - Bounded lookup time
  - $\log_b N$ hops
  - 4 hops for a million node network

CoDoNS: Informed Caching



- Proactive caching
  - Bindings pushed in anticipation
- Proactive updates
  - No timeouts
  - Immediate propagation of updates

CoDoNS: Informed Caching

- System-wide performance goals become mathematical optimization problems

Min. Overhead s.t. Performance = Target

Max. Performance s.t. Overhead ≤ Capacity

- Performance = lookup latency
- Overhead = bandwidth or memory

CoDoNS: Deployment

- Incrementally deployable
  - Uses legacy DNS to populate resource records on demand
  - Signs and introduces bindings so that CoDoNS nodes do not corrupt data (stop-gap)
- Retains DNS management infrastructure
  - DNS registries, Root authority
- Supports legacy clients

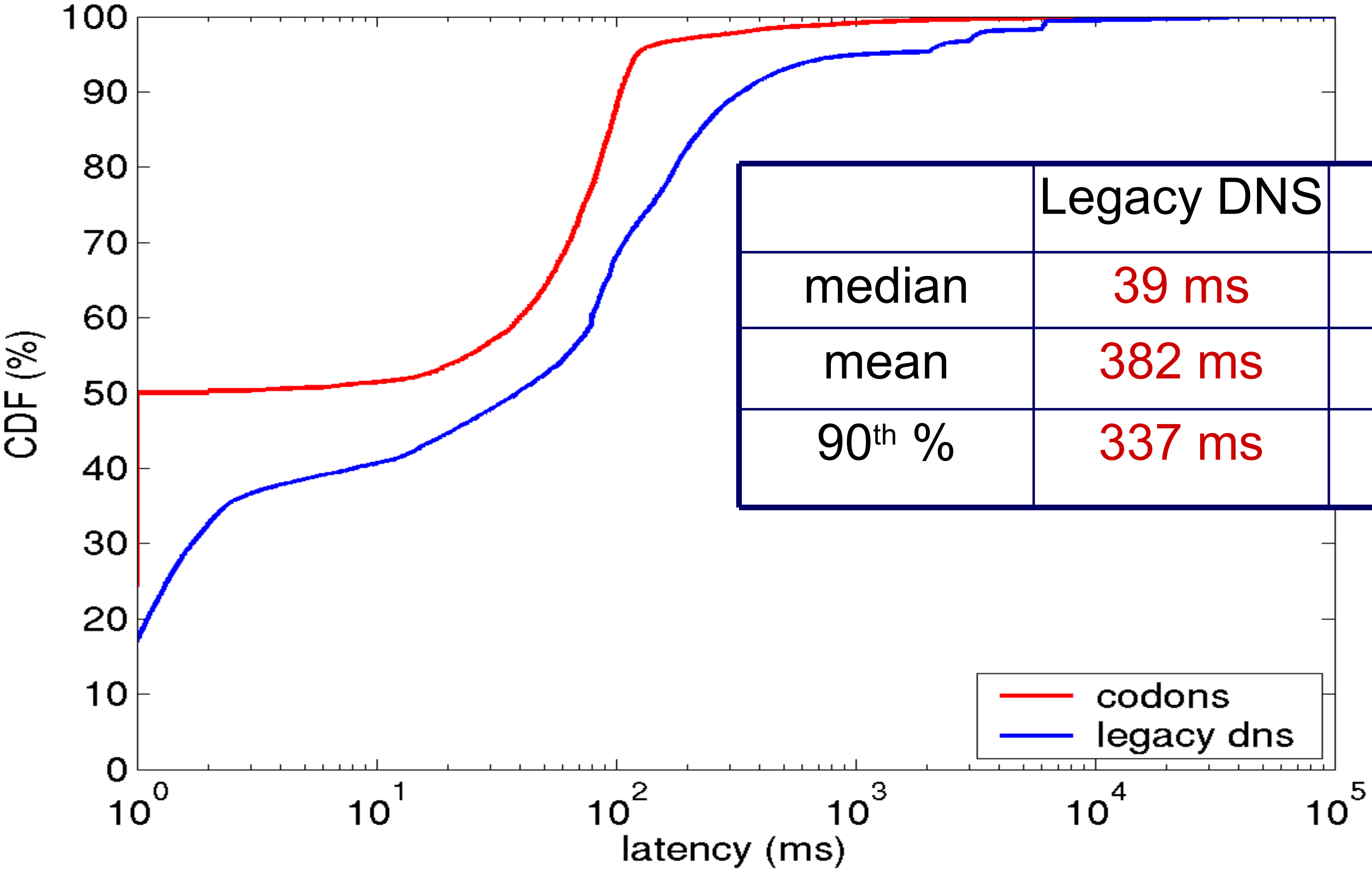
CoDoNS: Miscellaneous

- Negative responses
  - Cached temporarily

- Local names treated specially
  - Queries resolved locally without introducing load into the ring

- Server-side computation supported
  - Low-TTL records not cached, replaced with forwarding pointers
  - Supports Akamai and other CDN trickery

CoDoNS: Lookup Latency



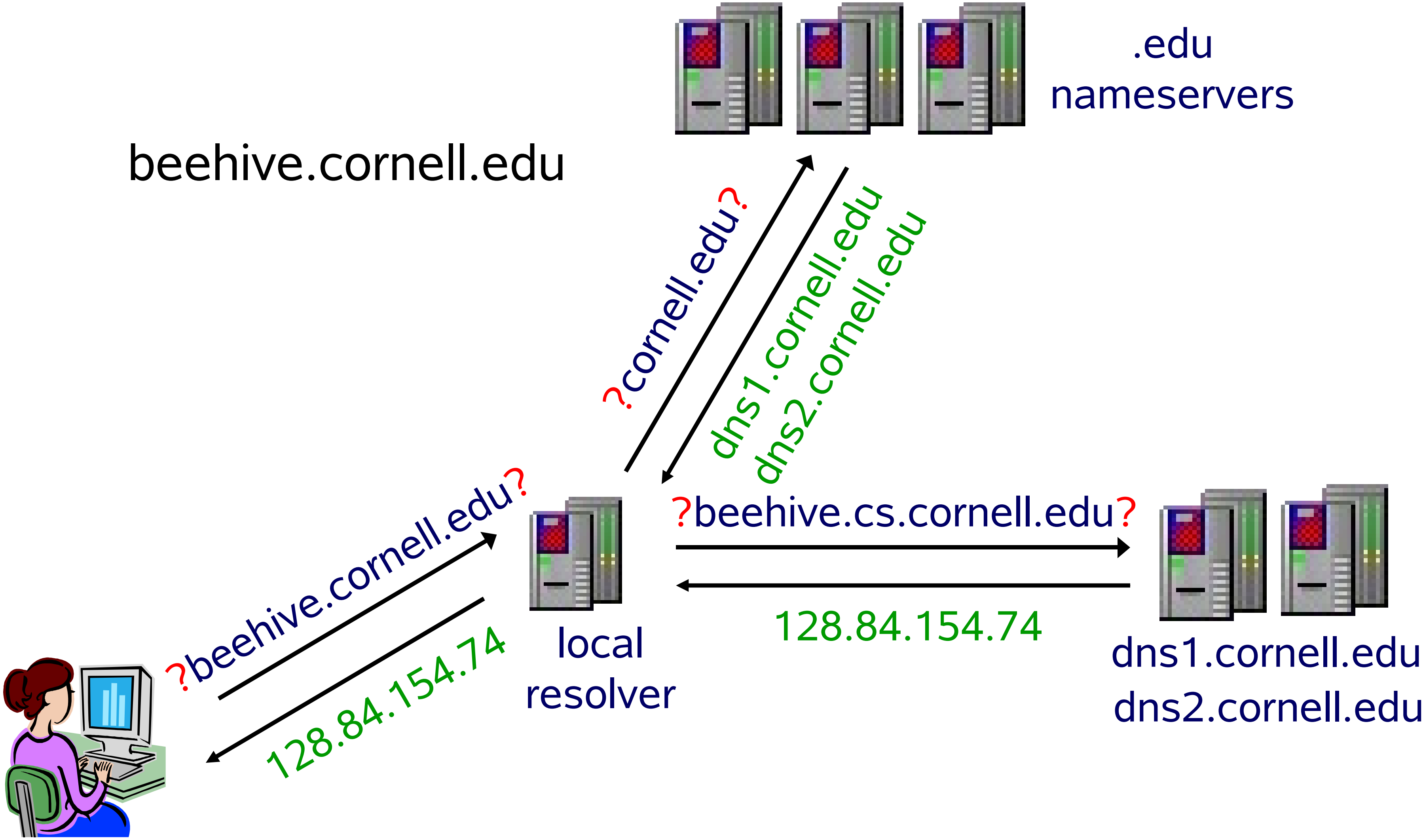
	Legacy DNS	CoDoNS
median	39 ms	2 ms
mean	382 ms	199 ms
90th %	337 ms	213 ms

Summary

- Separate namespace management from name resolution
- Use peer-to-peer architecture for name resolution
  - High availability, performance, and scalability

http://www.cs.cornell.edu/people/egs/beehive/

DNS: overview



beehive.cornell.edu

.edu
nameservers

?cornell.edu?

dns1.cornell.edu
dns2.cornell.edu

?beehive.cornell.edu?

128.84.154.74

local
resolver

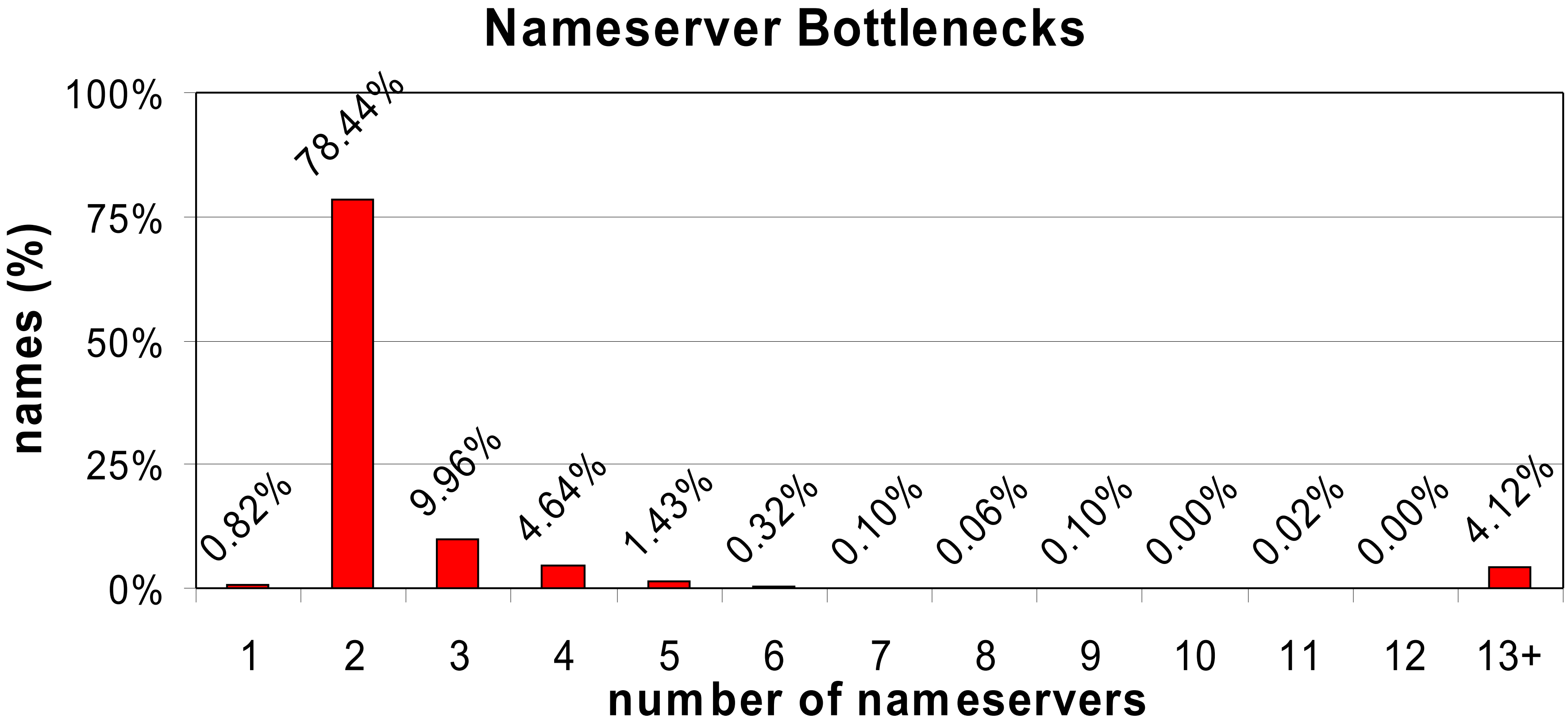
?beehive.cs.cornell.edu?

128.84.154.74

dns1.cornell.edu
dns2.cornell.edu

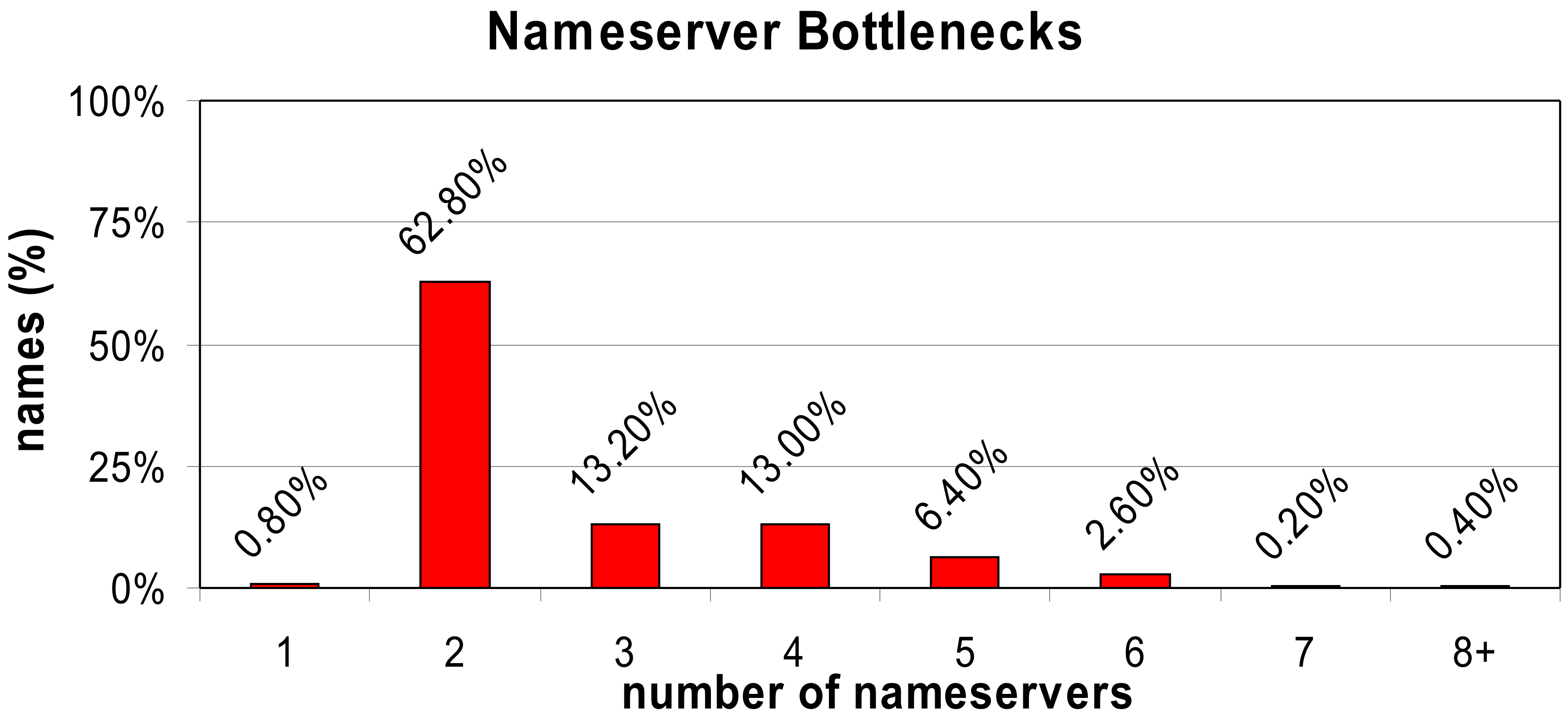
delegation bottlenecks (1/2)

- survey: 593160 domain names, 164089 nameservers
- 75% of names have a bottleneck of two nameservers



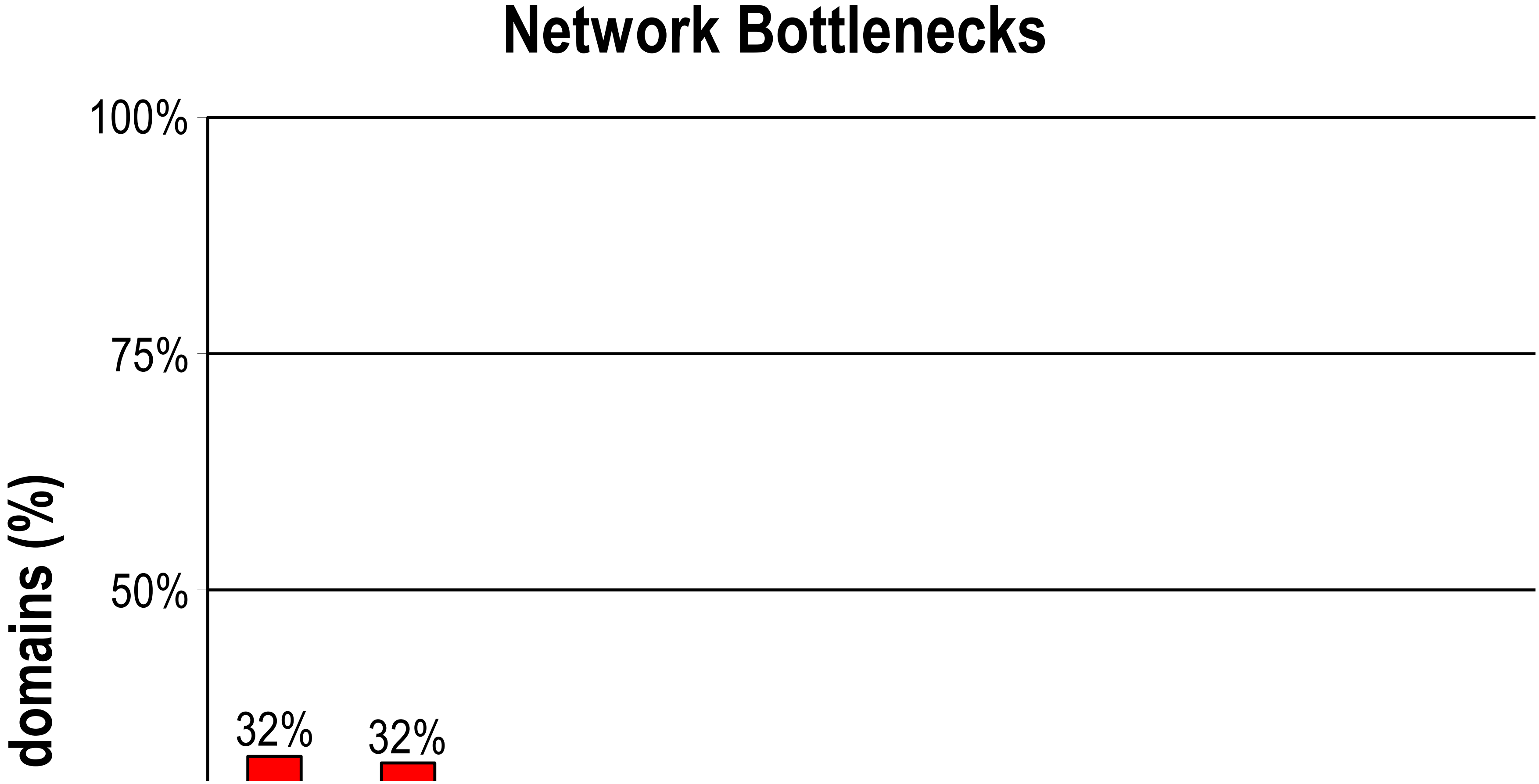
delegation bottlenecks (2/2)

- 60% of top-500 web sites have small bottlenecks



physical bottlenecks

- 30% of domains bottlenecked at one network link



DoS attacks

- delegation and network bottlenecks make DoS attacks feasible
  - january 2001 attack on Microsoft nameservers
- DoS attacks high up in the hierarchy can affect the whole system
  - october 2002 attack on root servers
  - roots are already disproportionately loaded [Brownlee et al. 01a, 01b]
- root anycast helps but does not solve the fundamental problem

performance

- dns lookups affect web latency
  - ~20-40% of web object retrieval time spent on DNS
  - ~20-30% of DNS lookups take more than 1s
  - [Jung et al. 01, Huitema et al. 00, Wills & Shang 00, Bent & Voelker 01]
- lame delegations
  - manual administration leads to inconsistencies
  - 15% of domains have lame delegations [Pappas et. al. 01]
  - introduces latency up to 30 sec
- server selection
  - disables caching with small timeouts (30 sec)
  - increases latency up to 2 orders of magnitude [Shaikh et. al. 01]

consistency

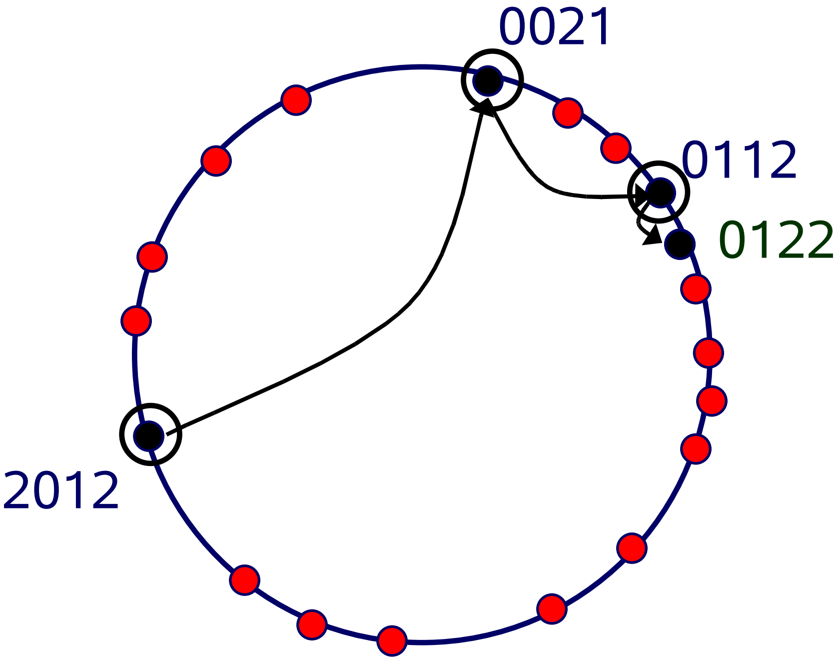
- DNS caching is timeout-driven
  - conflict in choosing timeouts
- fundamental tradeoff between lookup and update performance
- large timeouts
  - an emergency remapping/redirection cannot be performed unless anticipated
  - 86% of records have TTLs longer than 0.5 hours
- small timeouts (< 10 min)
  - increased lookup latency [Jung et. al. 01, Cohen et. al. 01]

CoDoNS: Structured Overlays

- supplement and/or replacement for legacy DNS
- based on distributed hash tables (DHTs)
  - self-organizing
  - failure resilient
  - scalable
  - worst-case performance bounds
- naïve application of DHTs fails to provide performance comparable to legacy DNS

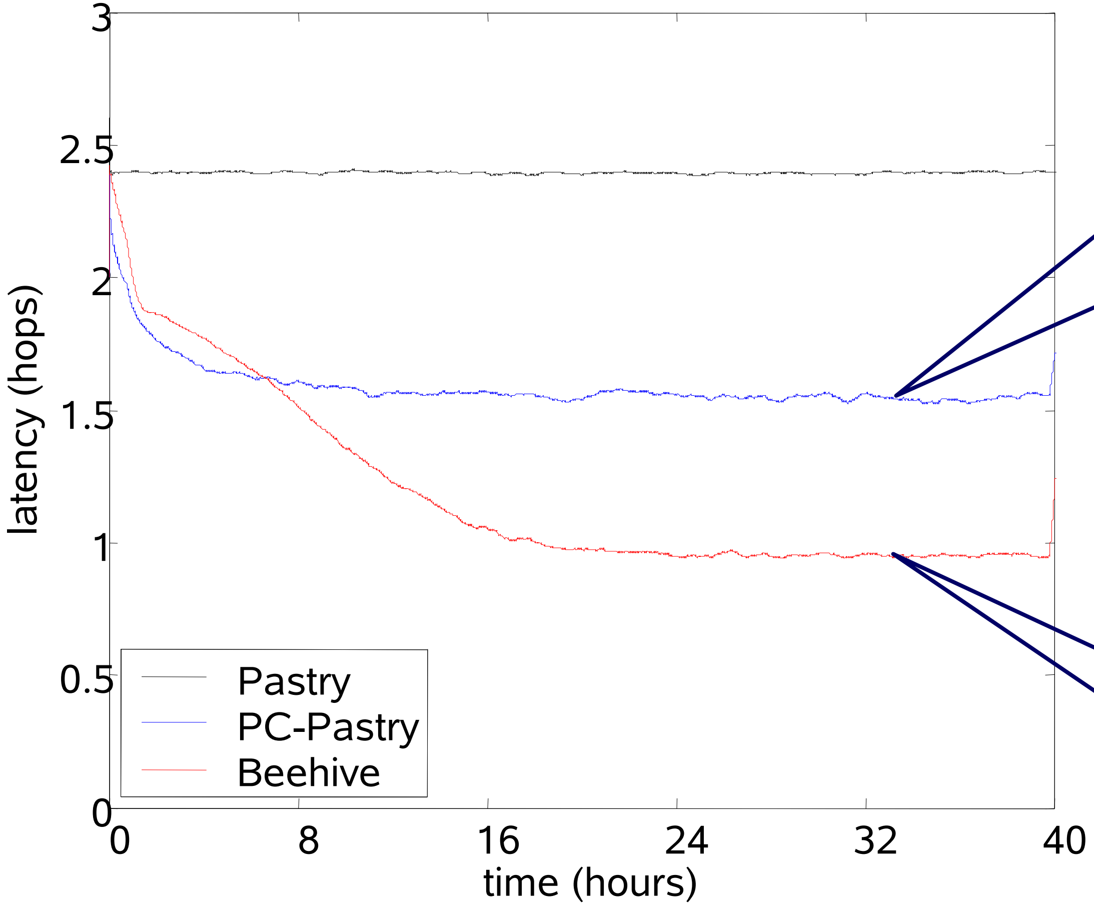
prefix-matching DHTs with caching

object 0121 =
hash("beehive.cornell.edu")



- cache along the lookup path
  - may improve lookups
- simulations [NSDI 04] show limited impact
  - heavy-tailed query distribution
  - TTL expiration

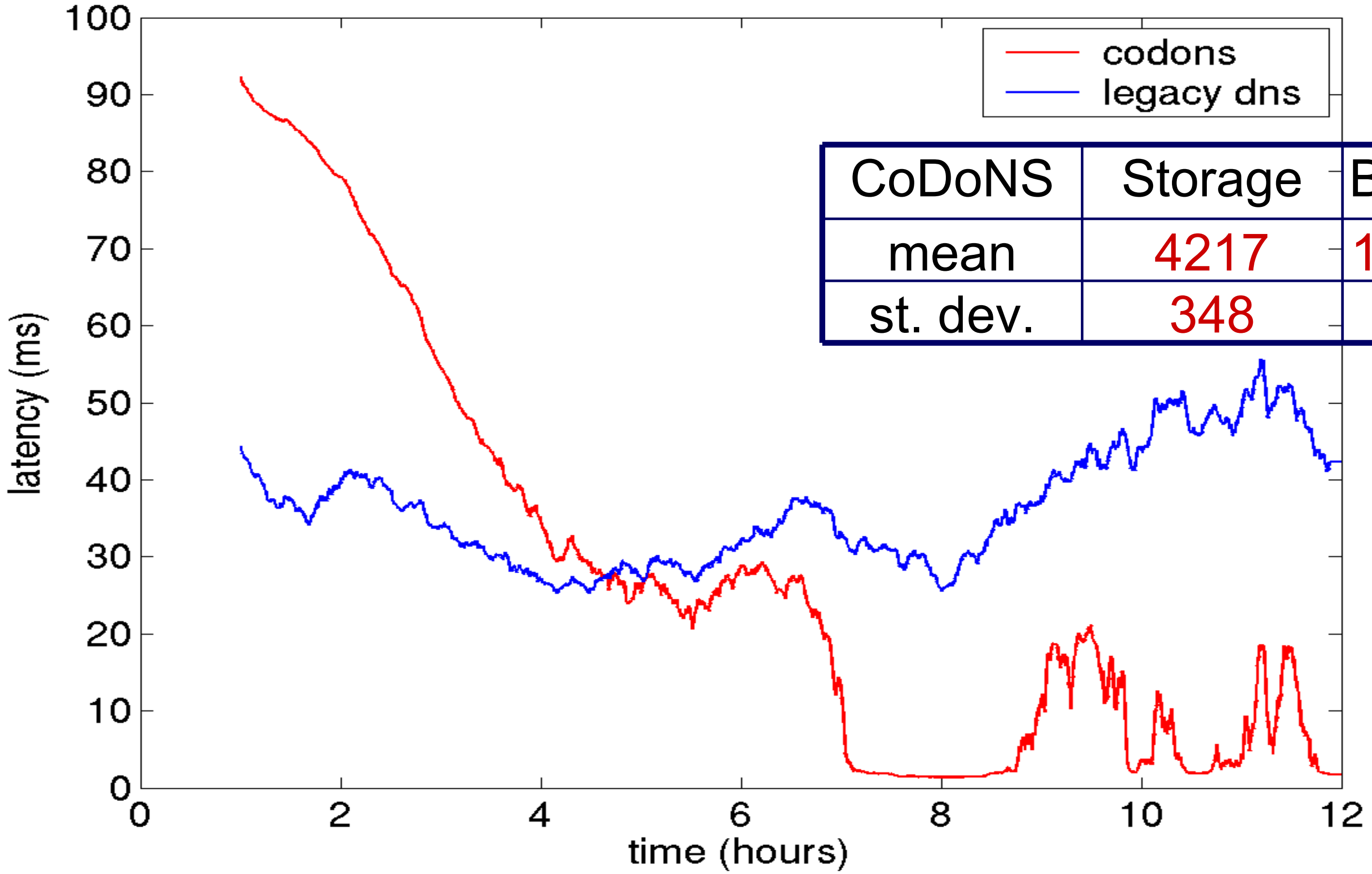
Beehive: lookup performance



passive caching is not very effective because of heavy tail query distribution and mutable objects.

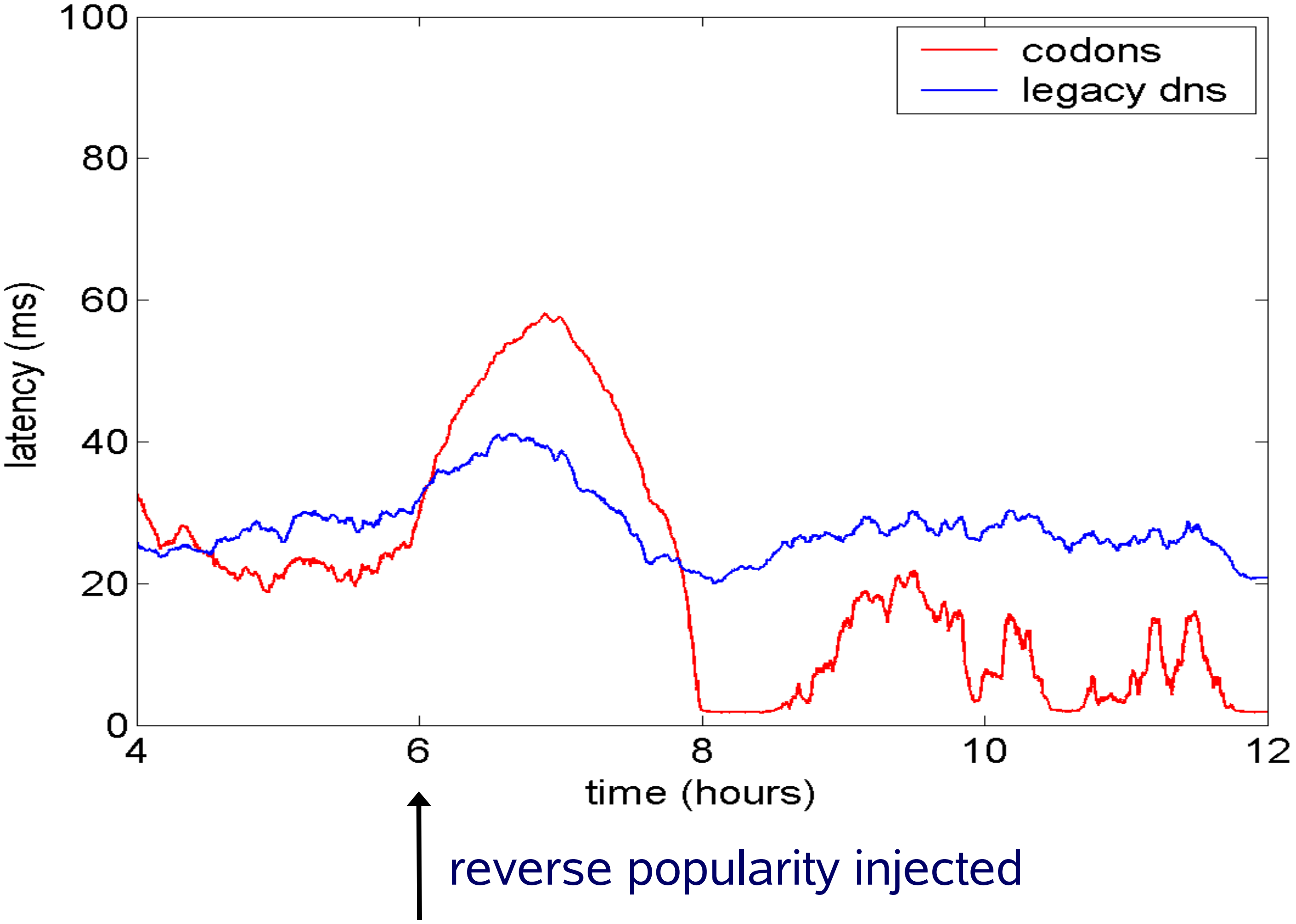
beehive converges to the target of 1 hop

CoDoNS: lookup performance (1/2)



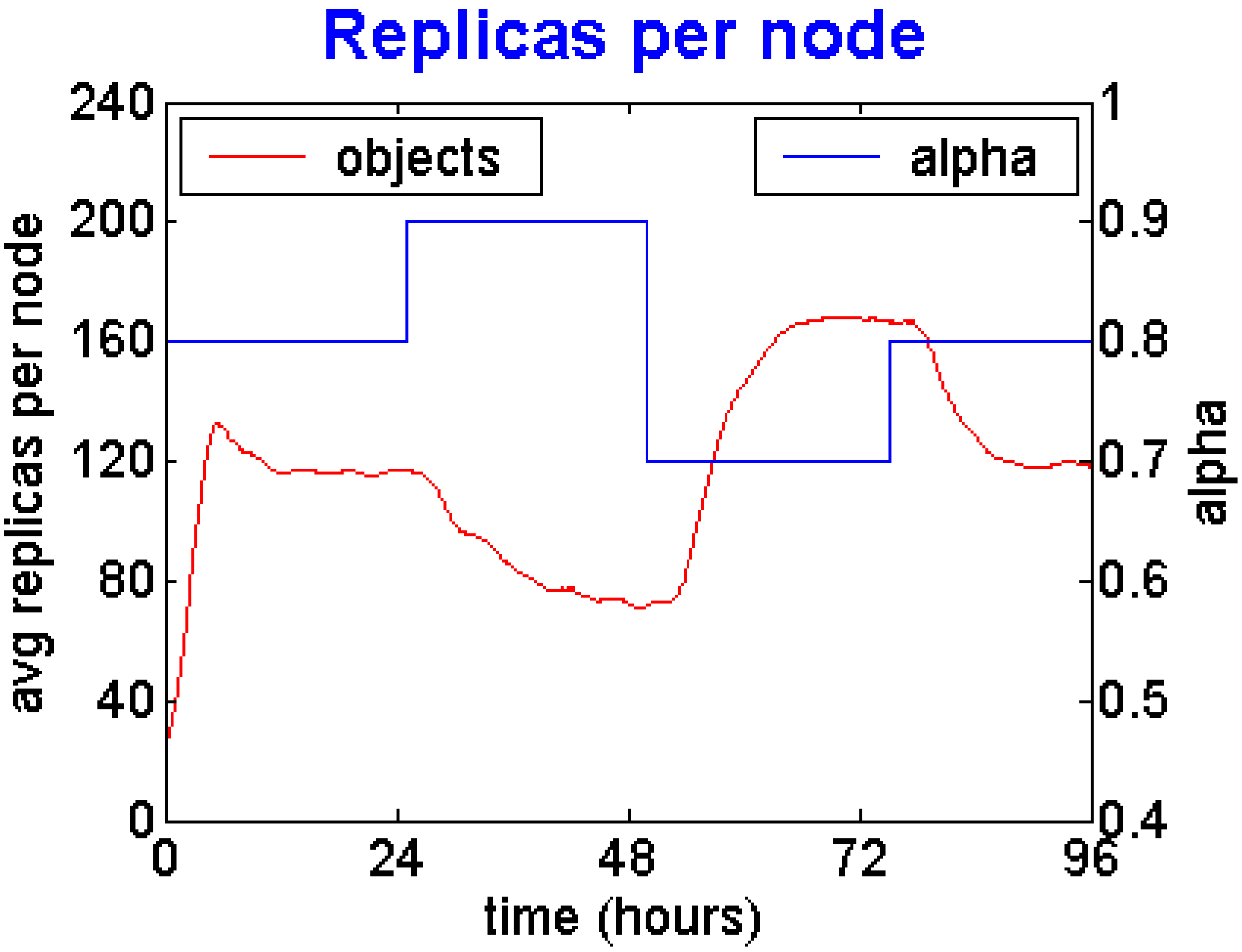
CoDoNS	Storage	Bandwidth
mean	4217	12.2 KBps
st. dev.	348	2.26

CoDoNS: flash crowds



reverse popularity injected

Beehive: zipf parameter change



structured DHTs (1/2)

Name	Lookup	Storage	Structure
CAN	$O(d\ N^{1/d})$	O(d)	d-dimenstional Torus
Pastry, Tapestry, Kademlia	O(log N)	O(log N)	prefix-matching
Chord	O(log N)	O(log N)	finger tables
Skipnet	O(log N)	O(log N)	skip list
Viceroy	O(log N)	O(1)	butterfly
Koorde, [Wieder & Naor 03]	O(log N/loglog N)	O(log N)	de Bruijn graphs

O(1) structured DHTs (2/2)

Name	Lookup	Storage
Farsite	d hops	$O(d\ N^{1/d})$
[Mizrak, Cheng, Kumar, Savage]	1-2 hops	$O(\sqrt{N})$
Kelips	1-2 hops	$O(\sqrt{N})$
[Gupta, Liskov, Rodrigues]	1 hop	$O(N)$

CoDoNs security

- not an issue in a single administration domain
  - e.g. akamai, google, msn, etc.

- attacks targeted at the DHT
  - Castro et al. ’02 work on secure DHTs

- attacks targeted at Beehive
  - outlier elimination
  - limited impact

- attacks targeted at CoDoNs
  - DNSSEC signatures
  - threshold cryptography

proactive, analysis-driven caching

- optimization problem

  minimize: total overhead, s.t.,

  average lookup performance $\leq$ C
- O(1) lookup latency
  - configurable target
  - continuous range, better than one-hop
- leverages object popularity to achieve high performance
- DNS follows zipf-like popularity distribution [Jung et. al. 01]

optimization problem

- level of replication (l):
  - object replicated at all nodes with l matching prefix digits
  - incurs at the most l hops per lookup
- min: $\sum s_i / b^{l_i}$ s.t., $\sum q_i . l_i \leq C$

s_i: per object overhead

- object size, update frequency, or number of replicas ($s_i = 1$)

q_i: relative query rate of object i

b: base of DHT

analytical solution: Zipf

minimize: (number of replicas)

$$x_0 + x_1/b + x_2/b^2 + \ldots + x_{K-1}/b^{K-1}$$

s.t., $K - (x_0^{1-\alpha} + x_1^{1-\alpha} + x_2^{1-\alpha} + \ldots + x_{K-1}^{1-\alpha}) \leq C$

x_i: fraction of objects replicated at level i

α: parameter of zipf distribution

$$x^*_i = \left[\frac{b'^i (K - C)}{1 + b' + \ldots + b'^{K-1}} \right]^{\frac{1}{1-\alpha}}$$

where $b' = b^{(1-\alpha)/\alpha}$

K: highest level of replication

computational solution

- relax integrality on variables
  - use linear-programming or steepest-descent to find optimal solution
  - fast O(M logM) time for M objects
- round-up solution to nearest integer
  - at the most replicates one extra object per node

- handle any popularity distribution
- include fine-grained overhead
  - object size, update frequency

CoDoNS operation (1/2)

- home node initially populates CoDoNS with binding from legacy DNS
  - upper-bound (K) on replication level ensures resilience against home-node failure
- proactive caching in the background replicates binding based on analytical model
  - local measurement and limited aggregation to estimate popularity of names and zipf parameter
  - discards bindings or pushes bindings only to neighbors

CoDoNS operation (2/2)

- dynamic adaptation
  - continuously monitor popularity of names and increase replication to meet unanticipated demand
  - handles DoS attacks and flash-crowds

- fast update propagation
  - replication level indicates the locations of all the replicas
  - the home node initiates a multicast using entries in DHT routing tables

CoDoNS name security

- all records carry cryptographic signatures
  - if the nameowner has a DNSSEC nameserver, CoDoNS will preserve the original signature
  - if not, CoDoNS will sign the DNS record with its own master key
- malicious peers cannot introduce fake bindings
- delegations are cryptographic
  - names not bound to servers

CoDoNS implications

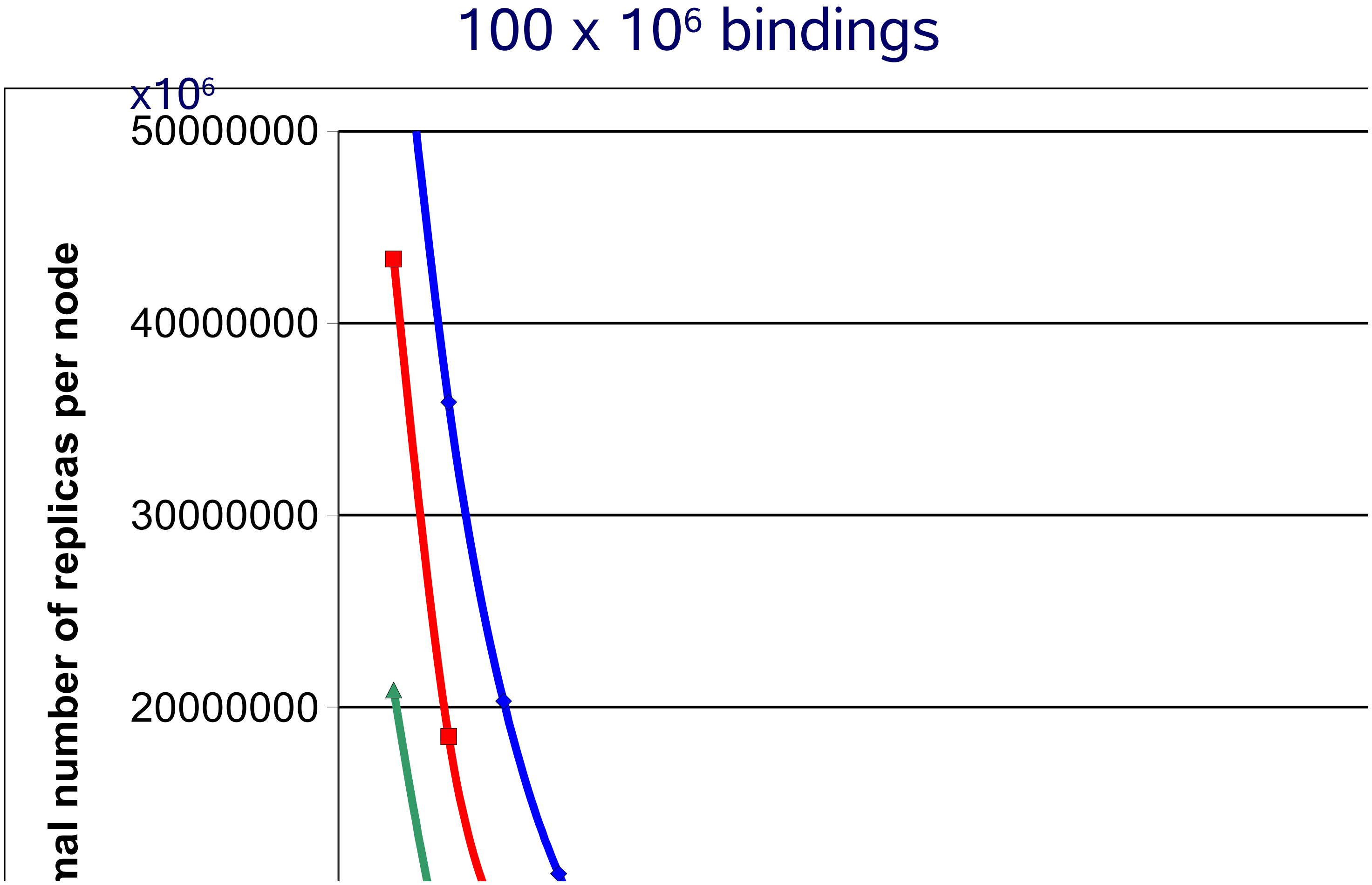
- name delegations can be purchased and propagated independently of server setup
- naming hierarchy independent of physical server hierarchy
- domains may be served by multiple namespace operators
  - competitive market for delegation services

evaluation

- MIT trace
  - 12 hour trace, 4th December 2000
  - 281,943 queries
  - 47,230 domain names
- Beehive: Simulation
  - 1024 nodes, 40960 objects
- CoDoNS: Planetlab deployment
  - 75 nodes

- Lookup performance
- Adaptation to changes in popularity
- Load balance, Update propagation [SIGCOMM 04]

latency vs. overhead tradeoff



advantages of CoDoNS

- resilient
  - self configures around host and network failures
  - resilient against denial of service attacks
  - load balances around hotspots

- high performance
  - low lookup latency
  - updates can be propagated at any time

- autonomic
  - no manual configuration, no lame delegations