The Case for Pushing DNS

Mark Handley and Adam Greenhalgh
UCL

In the beginning…

There was Jon Postel

And **`hosts.txt`**

And all was well.

Then came DNS

And scale.

And all was well.

Then came scale.

And all was was *not* well.

Then came DoS.

And scale.

And all was *not* well.

Lessons from Networking 101

- To make things scale, add hierarchy.
- To make things robust, avoid single points of failure.

DNS scales well because of its namespace hierarchy and elegant decentralized administration.

Because lookups follow the same hierarchy, DNS needs a root, and this is a potential single point of failure.

Lessons from History

- Unsuccessful large DoS attack against the root name servers in 2002.
  - Came close enough to be worrying.
  - Since then, anycast BGP has increased replication considerably. An attack of large enough scale would probably succeed though.

Cause for concern...

Thousands of companies are paying off online extortionists

2004 : 1m zombie machines.

RVICE ATTACKS

ZDNet UK

Oct 2004 : Criminal gangs use bot-nets to extort money.

D

By Gregg Keizer, TechWeb 24 October 2005 15:30 AEST Security

zombies
30,000+ internet connected zombie

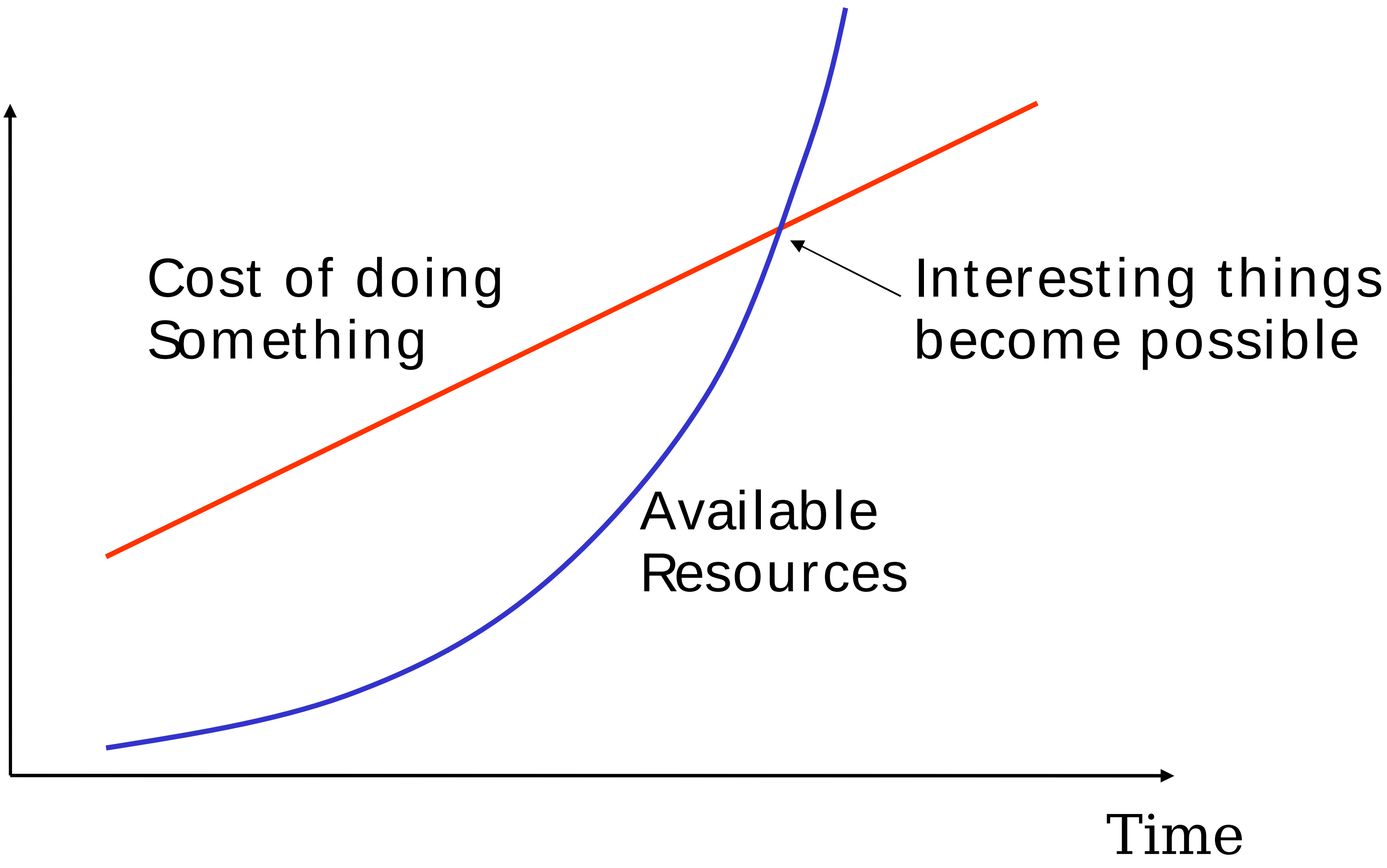
Oct 2005 : 1.5m host bot-net.

by Antony Savvas
Monday 7 November 2005

Broadband responsible for 93% increase in infected PCs in 2004

Nov 2005 : 0.4 m host bot-net conviction.

Generic Moore's Law Graph



Our idea

What's wrong with **`hosts.txt`**?

– Size of the data.

– Rate of change of the data.

– Centralized administration of the data.

– Distribution of the data to everyone.

Assertion:

– With careful design, none of these is a problem.

Size of the Data.

- Take the core of DNS: root, all TLDs only.
  - Only the nameserver records, SOA data.
  - This data is relatively stable.
  - (not A records – they are too dynamic)
- 76.9 million domains $\rightarrow$ 7.1 GB zone file.
  - Currently growing *linearly* at about 27K domains/day.
- Conclusion: any PC could store this without difficulty.

Centralized Administration of the Data.

- Don’t replace the existing DNS.
  - The administration process works reasonably well, modulo political issues.
- Just take the data already available and replicate it.

- Either:
  - Be Verisign.
  - Walk the DNS.
- Both are technically viable.

Distribution of Data

- Goals:
  - Replicate all 7GB of data to any DNS server in the world that wants it.
  - Do this at least once per day.

- Obvious solutions:
  - Multicast
  - Peer-to-peer.
- We chose the latter.

Back of the envelope…

77 million domains -> 7.1 GB zone file

If each peer sends to 3 neighbors then 20,000 DNS servers reached in ~ 10 generations.

To reach all servers in 24 hours, need to transfer from one node to another in 2.4 hours.

21 Mb/s outgoing from each server during transfers.

7 Mb/s with a compression ratio of 3:1.

Trust and Data Validity

- Simplest model:
  - Just sign the zone file.
  - Embed the public key in all peer-to-peer software.
  - Check the signature before passing data on.

- Nice properties:
  - A bad node can’t pass on bad data.
  - Trust model is same as current Verisign root model.

Data Replication

- Issues:
  - 7 Mb/s is a little high.
  - Have to receive 7GB of data before checking sig.
- Refinement: Split the zone file into 1MB signed chunks.
  - Can forward one chunk while receiving next one. This spreads forwarding over the entire day.
  - Can reduce the fan-out degree to 2 because more generations not such an issue.
- Result: compressed data rate is now 470 Kb/s.

The story so far...

- Data size is not an issue.
- Data administration not an issue.
- Data replication is not an issue.
- Data corruption is not an issue.

- Potential issues:
  - DoS by servers within the peer-to-peer mesh.
  - Trust: one signature is fragile.
  - Churn: how fast does the data change?

Potential Issue 1:

Insider attacks...

A server can’t corrupt data, but it can:

- Refuse to forward data.
- Sink data from many peers (sybil attack).
- Corrupt the structure of the peer-to-peer network.

To address the latter, use a mixture of peering types:

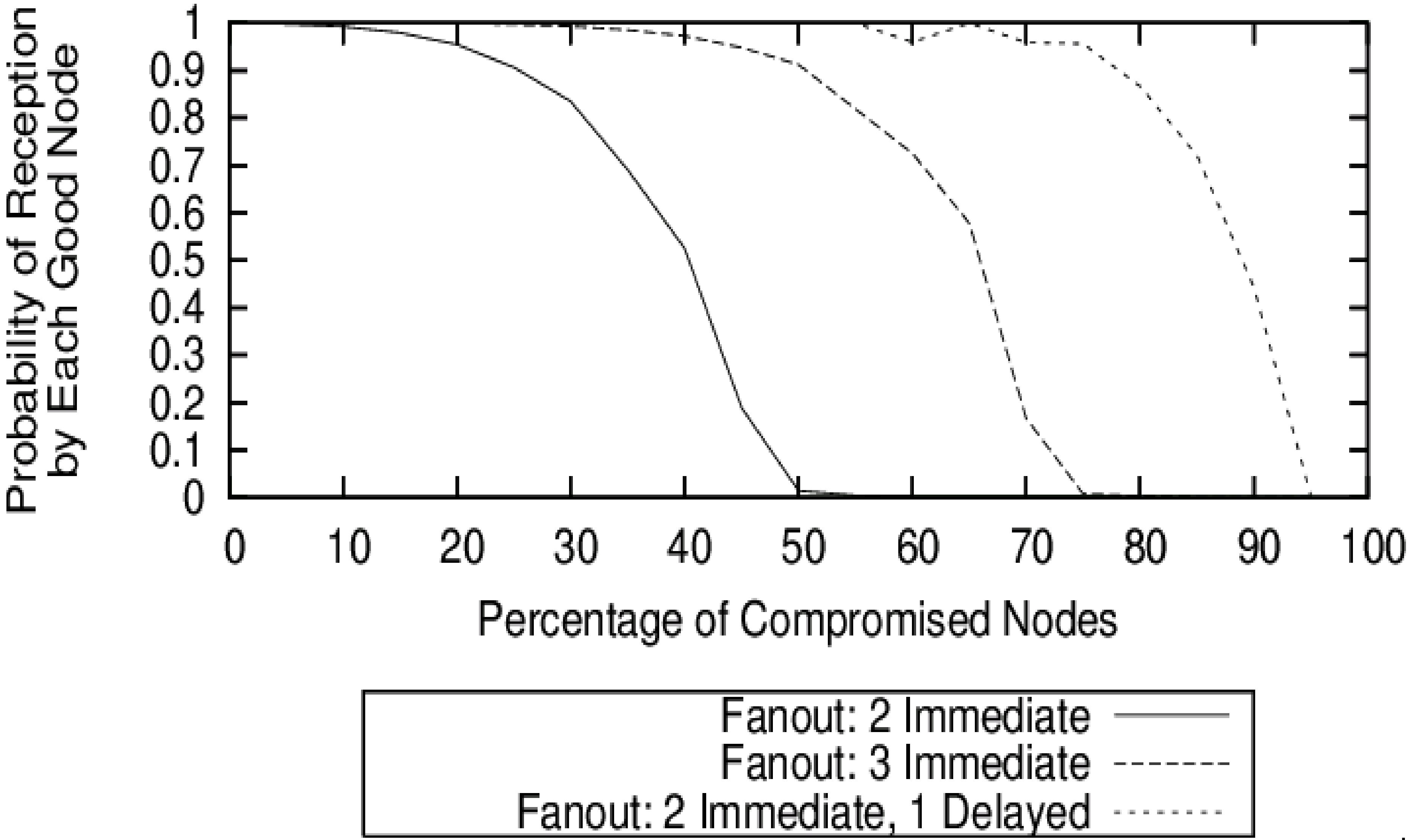
- Configured peerings, similar to NNTP
  - improve locality, not subject to structural attacks
- Randomised peerings
  - improve small world properties

Potential Issue 1:

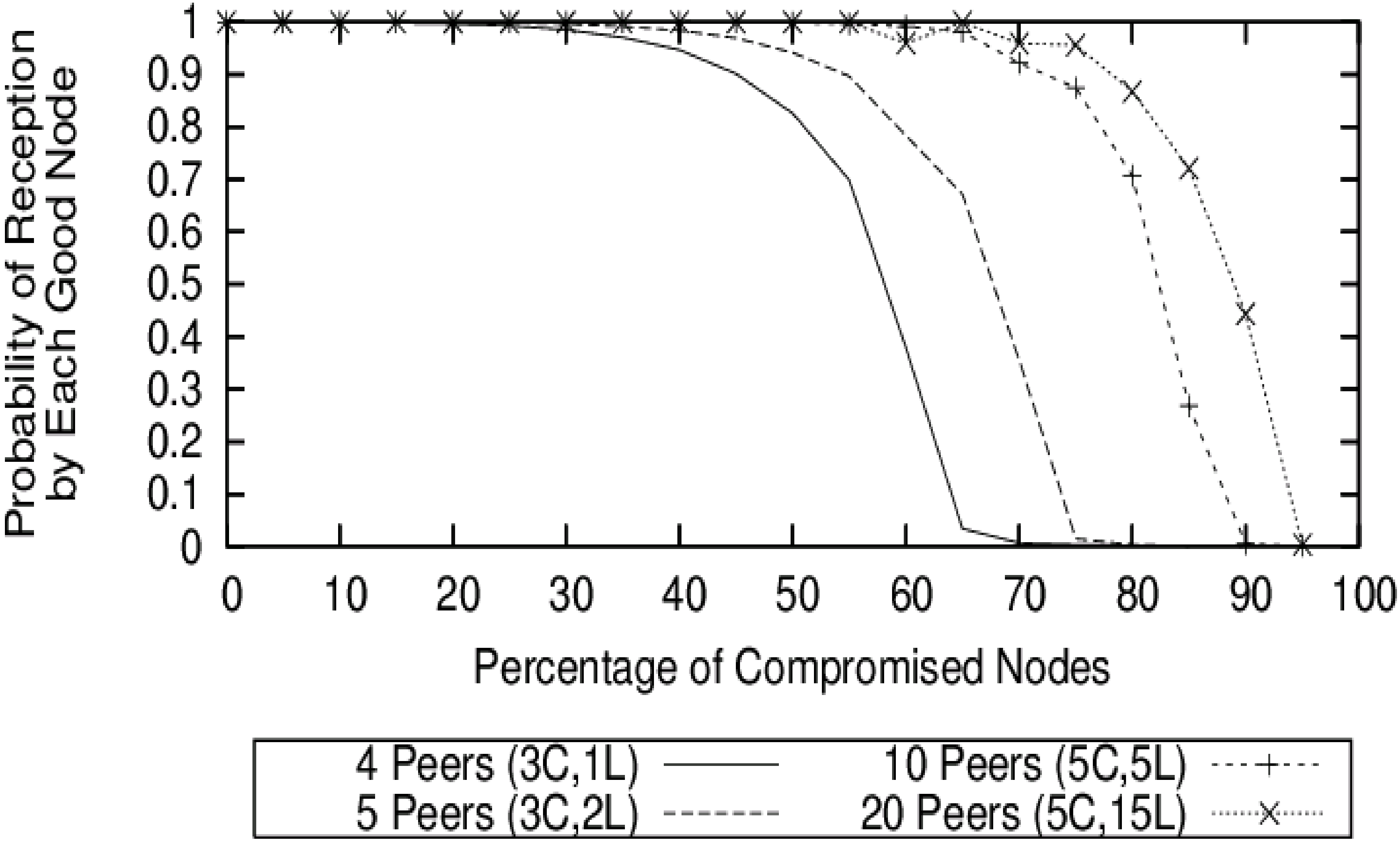
Insider attacks…

- Wrote a simple simulator to examine reliability in the face of a large number of malicious nodes within the peer-to-peer mesh.
- Evaluate:
  - Effects of number of peers of each type.
  - Strategies for choosing who to send to, and when to send.

Insider attacks…



Insider attacks...



Simulation Summary

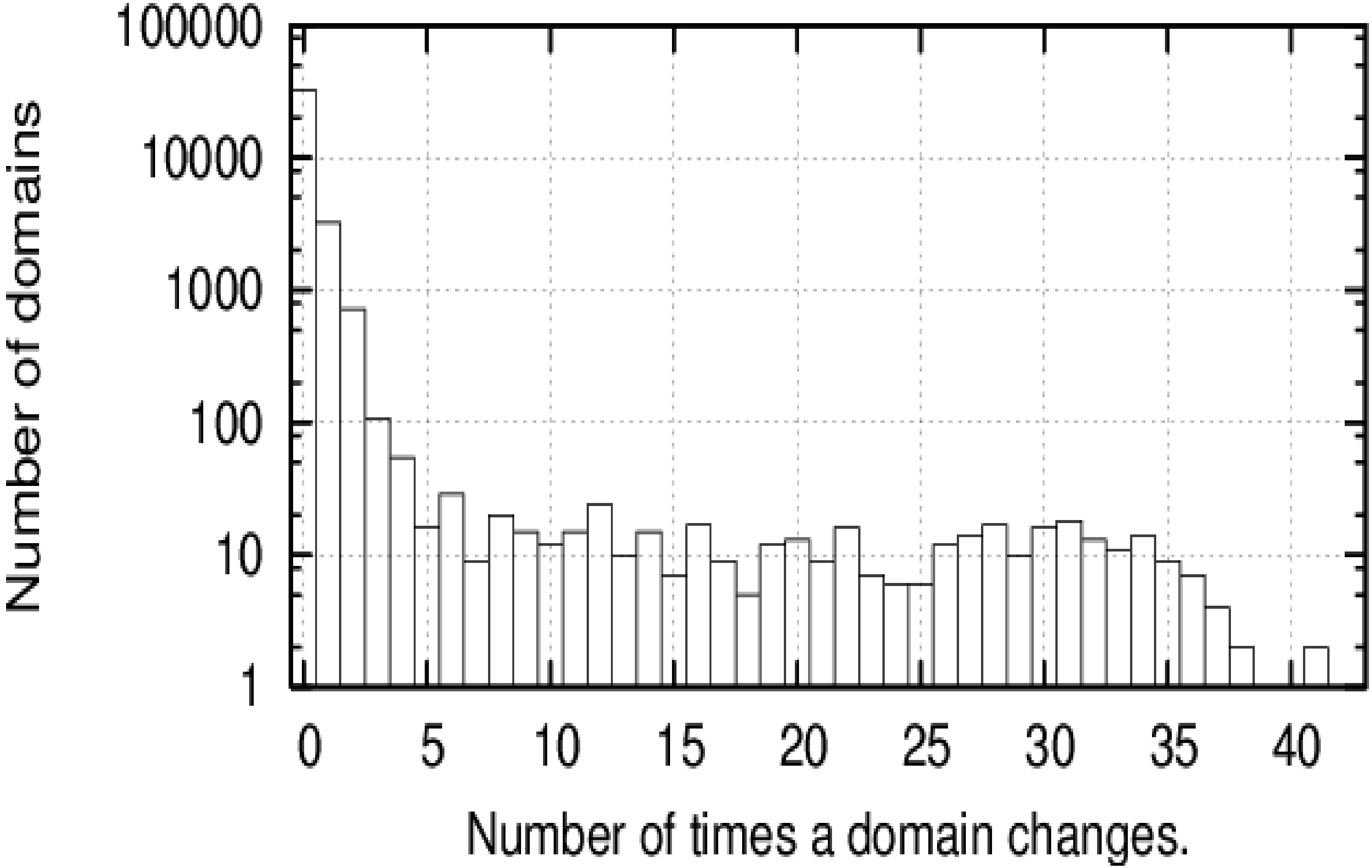
- Peer-to-peer flooding, done right, is *efficient* and *extremely robust* to insider attacks.

Potential Issue 2:

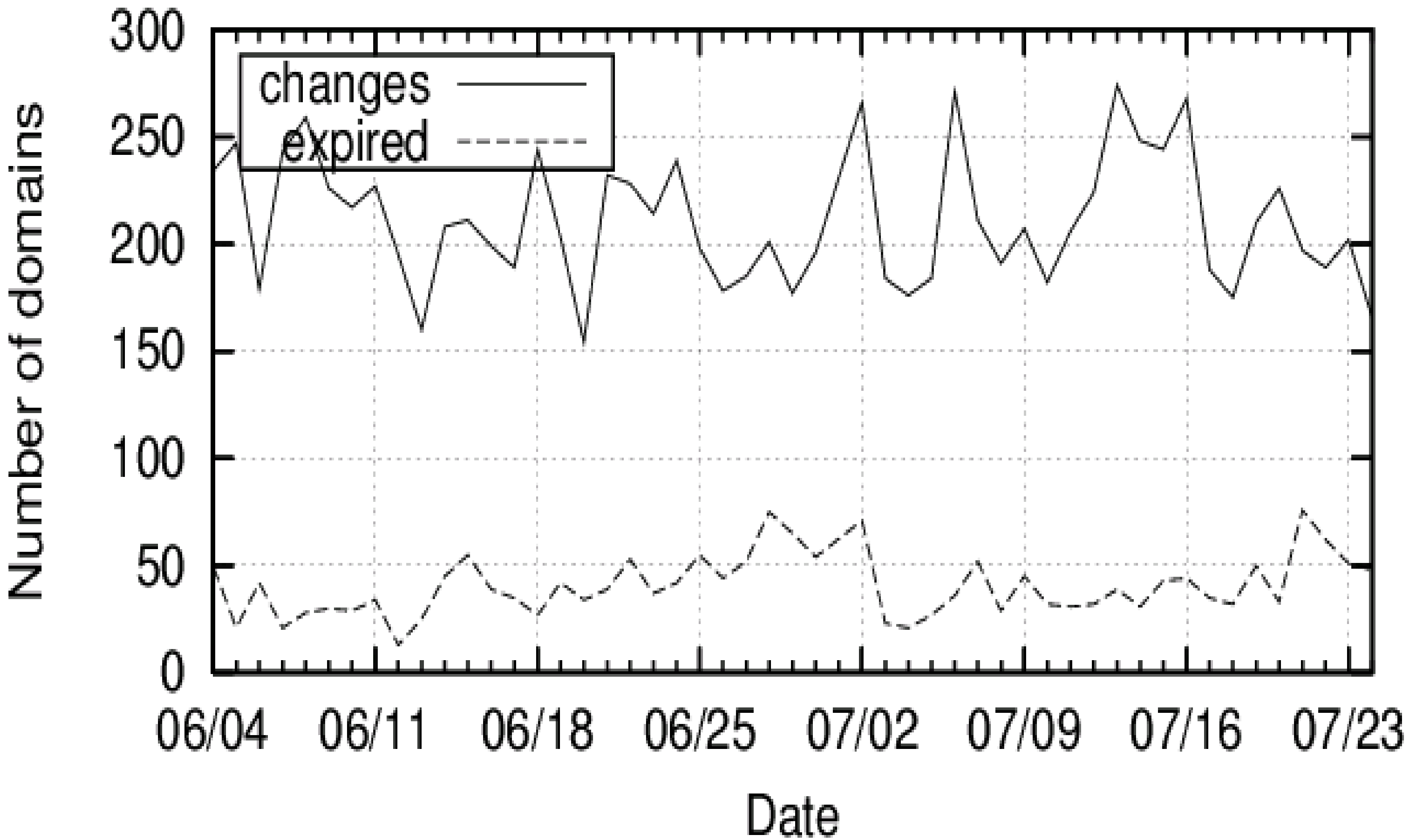
Rate of Change of Data

- We wrote a DNS monitor to observe how often DNS nameserver records actually change in the wild.
- 37,000 domains were monitored.
- Monitored domains for 60 days

Domain fluctuation



Changing domains and expiring



Rate of Change

- Each day :
  - 0.5% of domains change a nameserver entry.
  - 0.1% of domains expire.
- If we extrapolate to the entire DNS
  - 420,000 domains change per day
  - 100,000 domains expire per day
- Past growth figures suggest
  - 127,000 domains are created per day

Implications of Rate of Change

- Rate of change is not a big problem.
- But would be nice if updates didn’t have to wait 24 hours.
- Can send whole data set weekly, then send *cumulative* deltas (since last weekly update) on an hourly basis.
  - Cumulative updates are higher bitrate, but much more robust as you only need the most recent of them.
  - Required data rate is 850Kb/s to send to three peers.

Potential Issue 3:
Trust

- Single signing authority is fragile.
- In the long run, probably not politically viable.

- DNSpush architecture can support multiple signatories originating data.
  - Can majority vote if they disagree.
- One master which sends signed data
- Other signatories send :
  - signatures for the master data
  - diffs where they disagree with master.

Conclusions.

We have shown that :

- The dumb solution is viable and removes the current weak point in the DNS system.
- It provides resilience to significant numbers of zombies.
- It enables the introduction of a new trust model.
- The data rates are reasonable and manageable even for a DSL customer.