Measuring anycast server performance

The case of K-root

Agenda

- Introduction
- Latency
  - Client-side
  - Server-side
- Benefit of individual nodes
- Stability
- Routing issues

Why anycast?

- Root server anycast widely deployed
  - C, F, I, J, K, M at least
- Reasons for anycasting:
  - Provide resiliency (e.g. contain DOS attacks)
  - Spread server and network load
  - Increase performance
- But is it effective?
  - Resiliency is a given
  - What about stability and performance?

Latency

Measuring latency

- Ideally: for every given client, BGP should choose the node with the lowest RTT. Does it?

- From every client, measure RTTs to:
  - Anycast IP address (193.0.14.129), $\mathbf{RTT_K}$
  - Service interfaces of global nodes (not anycasted), $\mathbf{RTT_i}$ (i =1, 2, …)

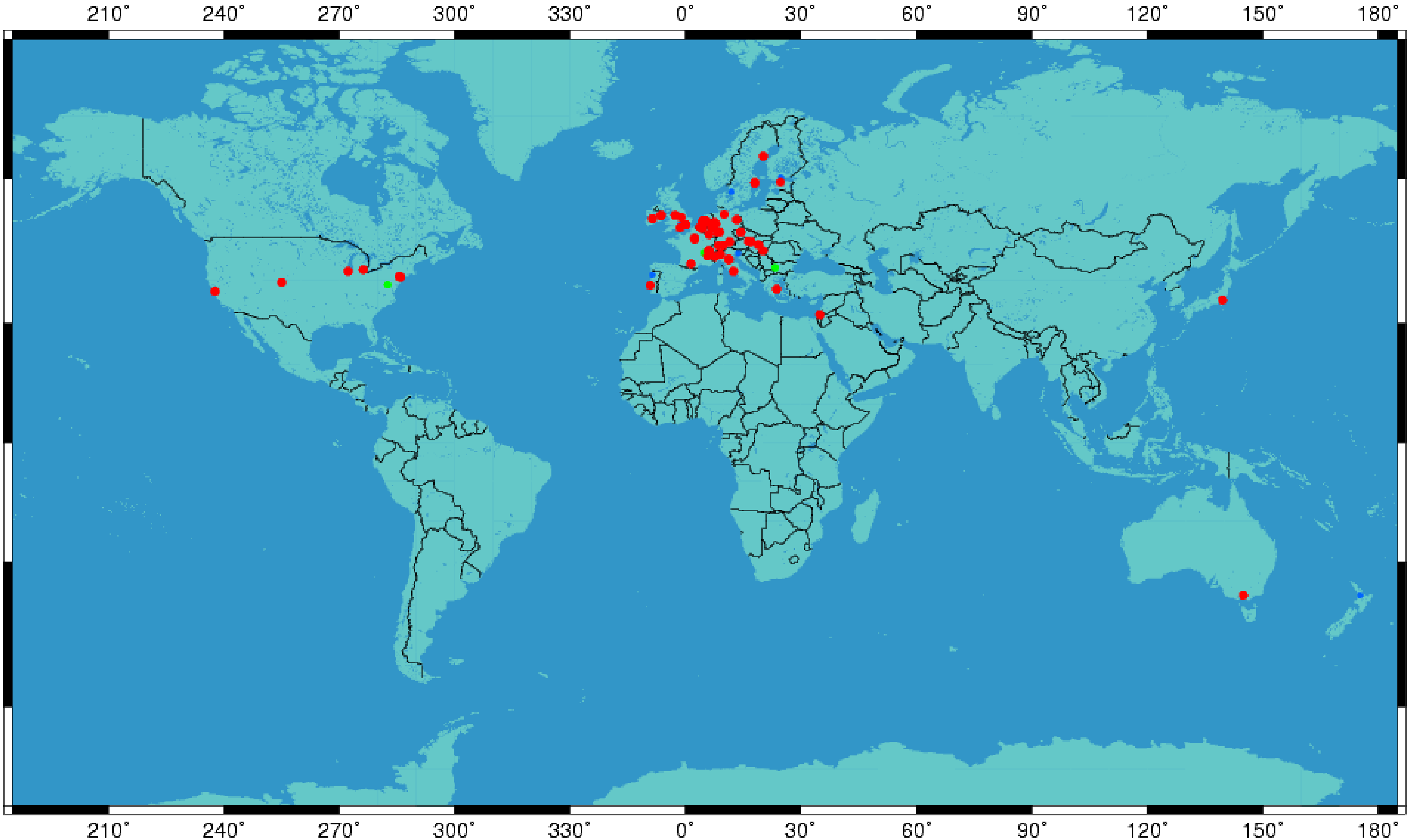
- For every client, compare K RTT to RTT of closest global node
- $\alpha = RTT_K / \min(RTT_i)$
  - $\alpha \approx 1$: BGP picks the right node
  - $\alpha > 1$: BGP picks the wrong node
  - $\alpha < 1$: local node?

Latency with TTM: methodology

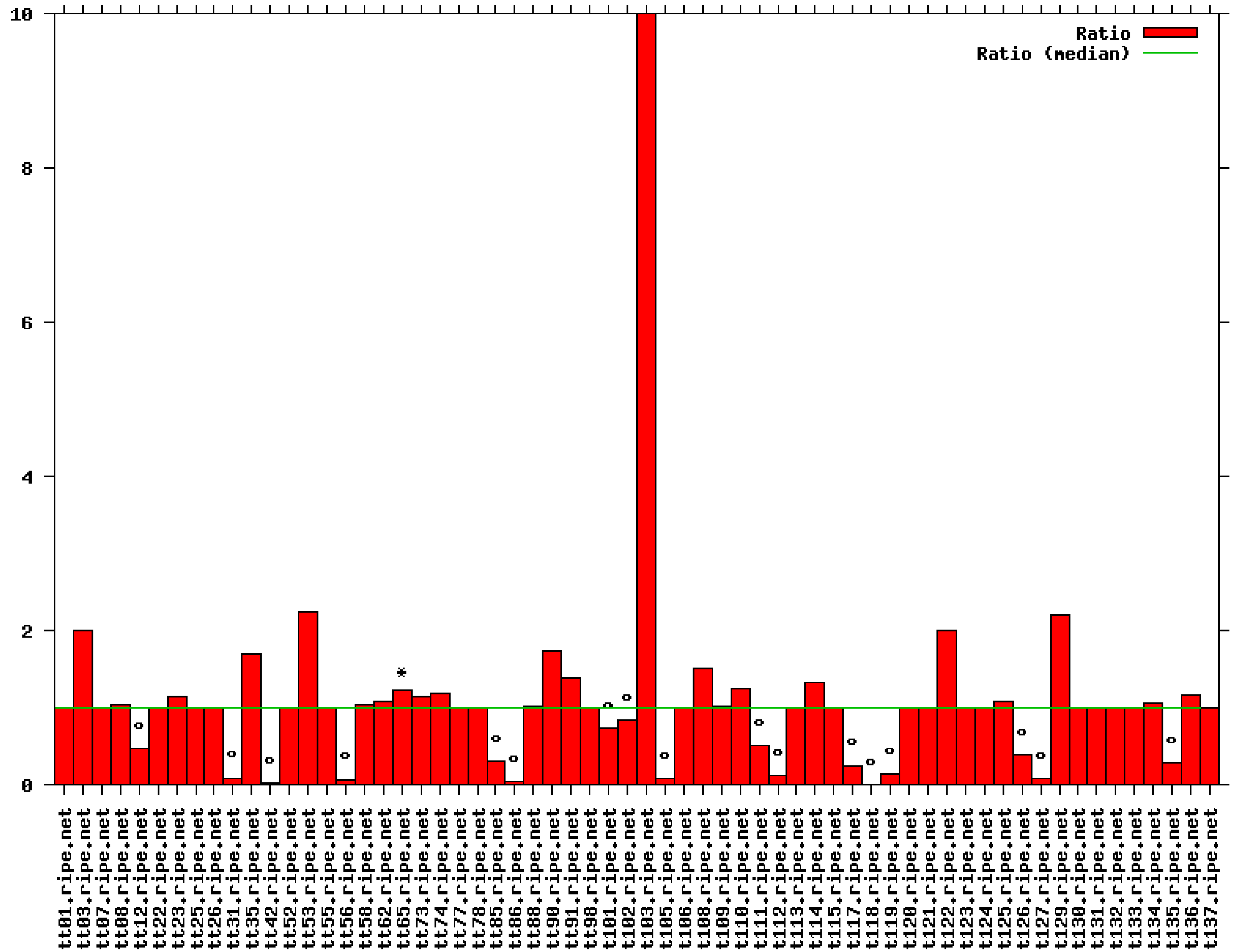
- Send DNS queries from ~100 TTM test-boxes
- For each test-box:
  - For each K-root IP:
    - Do a “dig hostname.bind”
    - Extract RTT
    - Take minimum value of 5 queries
  - Calculate α

- To make sure this is apples to apples:
  - Are paths to service interfaces the same as to production IP?
  - According to the RIS, “mostly yes”

TTM: probe locations

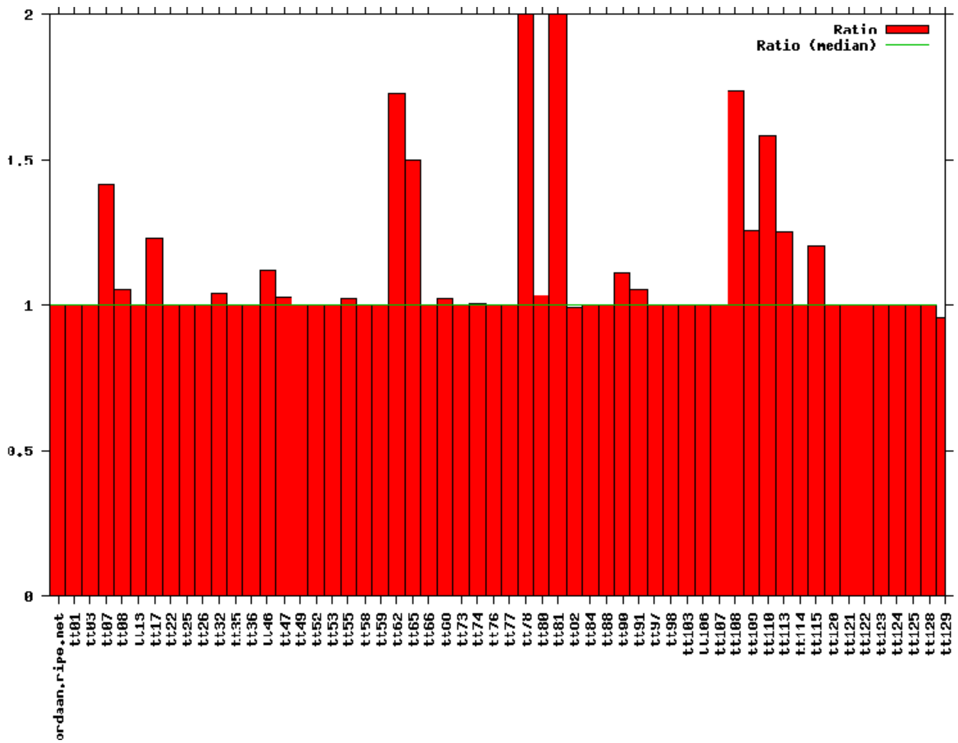


Latency with TTM: results (5 nodes)

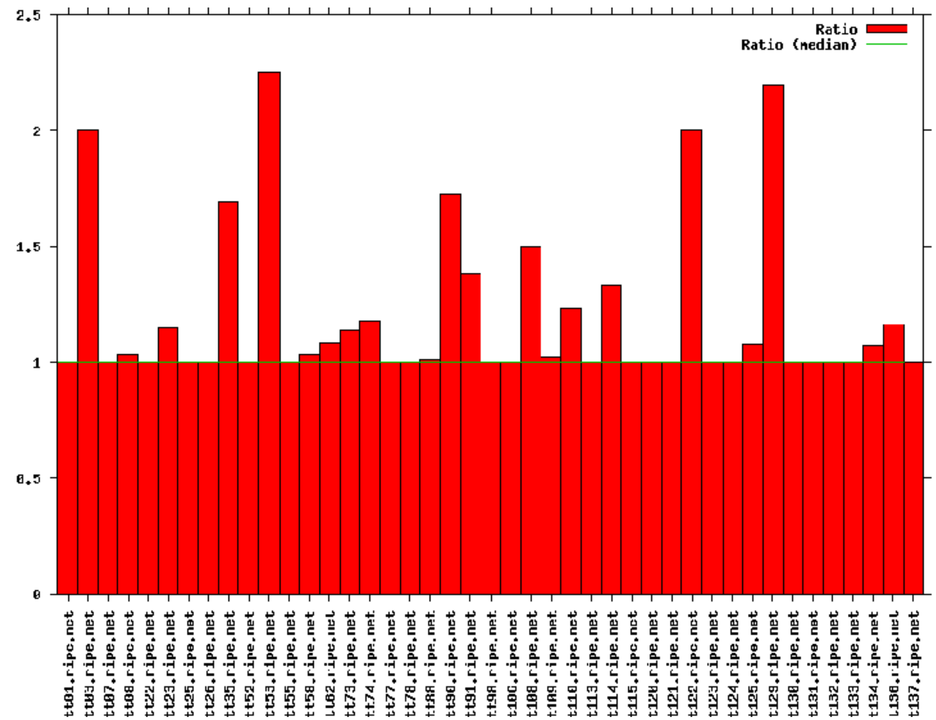


RIPE NCC

From two nodes to five nodes



2 nodes (April 2005)



5 nodes (April 2006)

Essentially no different

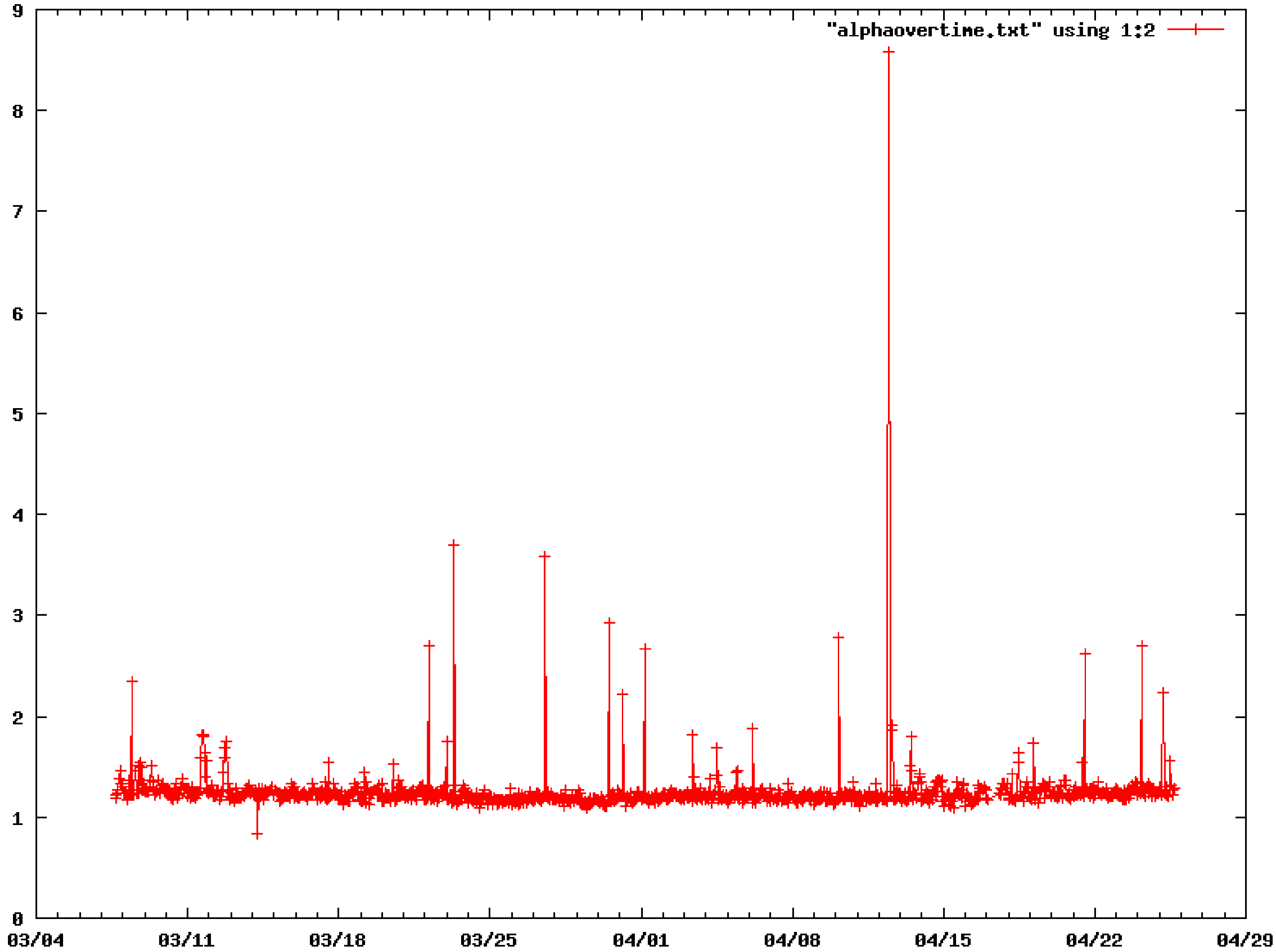
Consistency of α over time

- Is this a chance event or is this behaviour consistent?

- Plot average α over time
  - Collect α for all test-boxes every hour
  - Take average (excluding tt103)
  - Plot over time

- Results:
  - Average: 1.25, median: 1.22
  - BGP is fairly consistent

Average α over time



Server-side latency

Measuring from servers

- TTM latency measurements not optimal
  - Locations biased towards Europe
  - Only limited number of probes (~100)
  - Do not necessarily reflect K client distribution

- How do we fix this?

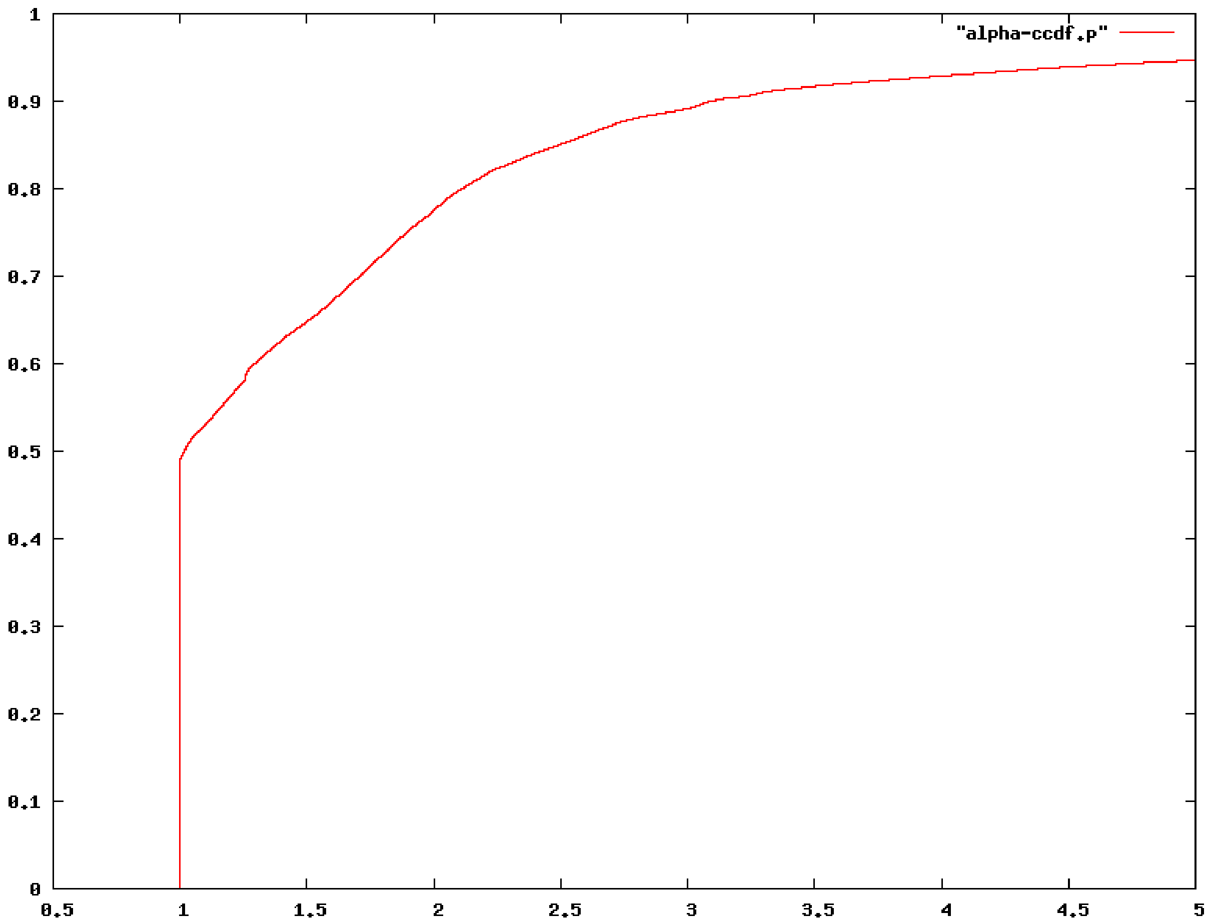
- Ping servers from clients
  - Much larger data set (~100 -> ~ 1M)
  - Measures the effect K's actual clients

Methodology

- Methodology:
  - Process packet traces on K global nodes
  - Extract list of client IP addresses
  - Ping all addresses from all global nodes
  - Plot distribution of α

- Results:
  - 6 hours of data
  - 246,769,005 queries
  - 845,328 IP addresses

CDF of α seen from servers



- Results not as good as seen by TTM
  - Only 50% of clients have $\alpha = 1$

Latency: conclusions

- 5-node results comparable to 2-node results
- TTM clients (= Europe) very well served by K
- If we look at total K client population, things not so rosy

Incremental benefit of nodes

How many nodes are enough?

- Does it make sense to deploy more instances?
  - Have we reached the point of diminishing returns?
- Evaluate benefit of existing instances
  - Hope this will tell us at what point in the curve we're on
- How do we measure the benefit of an instance?
  - We can quantify how much performance would worsen if that instance did not exist

Methodology

- Assume optimal instance selection
  - That is, every client sees closest instance
  - This is an upper bound to benefit
    - Consistent with our aim of seeing whether we have reached the point of diminishing returns

- For every client, see how much its performance would suffer if a given instance did not exist
  - We can do this because we ping all clients from all instances

Loss factor

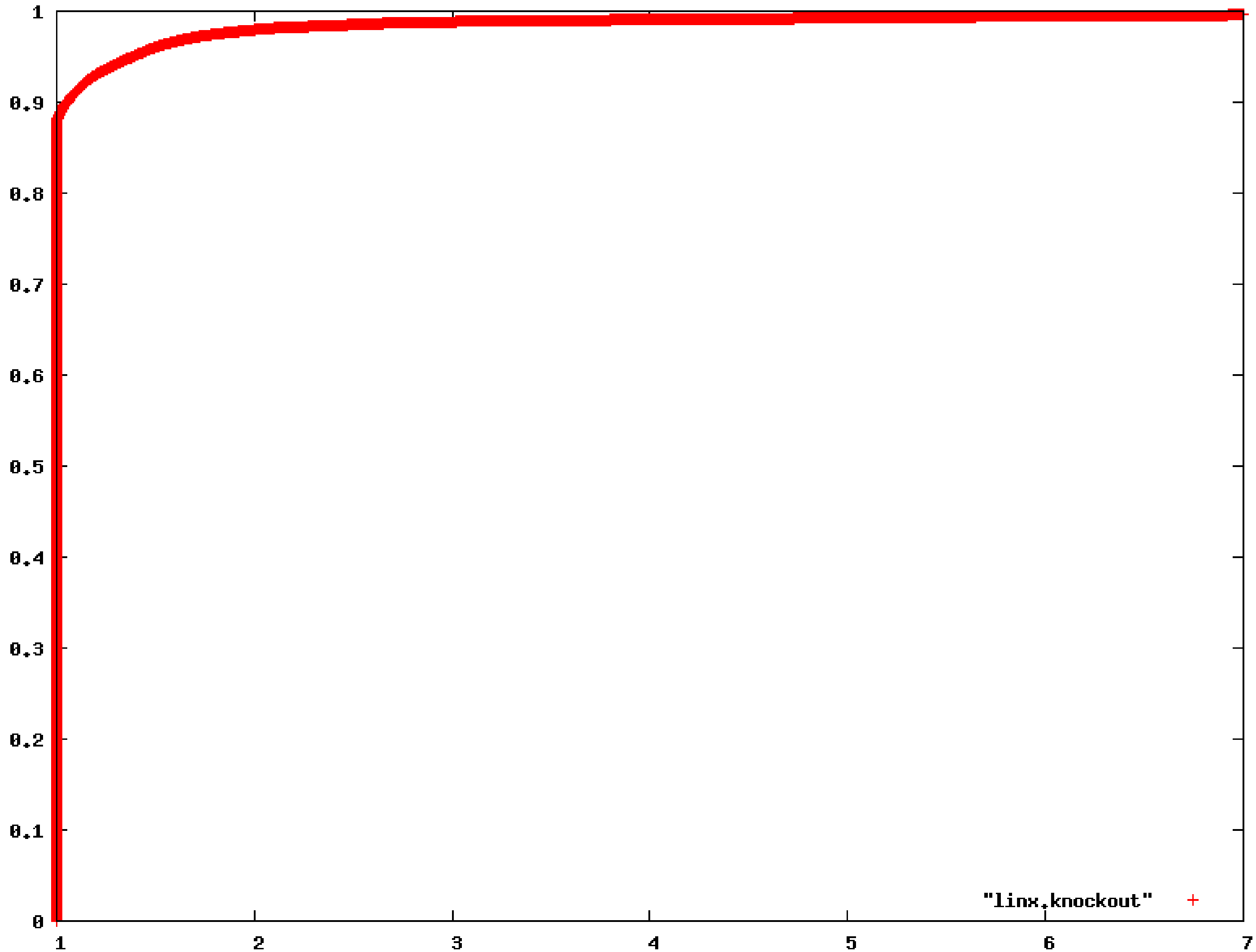
- “Loss factor” β determines how much a client would suffer if an instance were knocked out

$$\beta = \frac{RTT_{knockout}}{RTT_{best}}$$

- If β = 1, the client would see no loss in performance
- If β = 2, the client sees double RTT

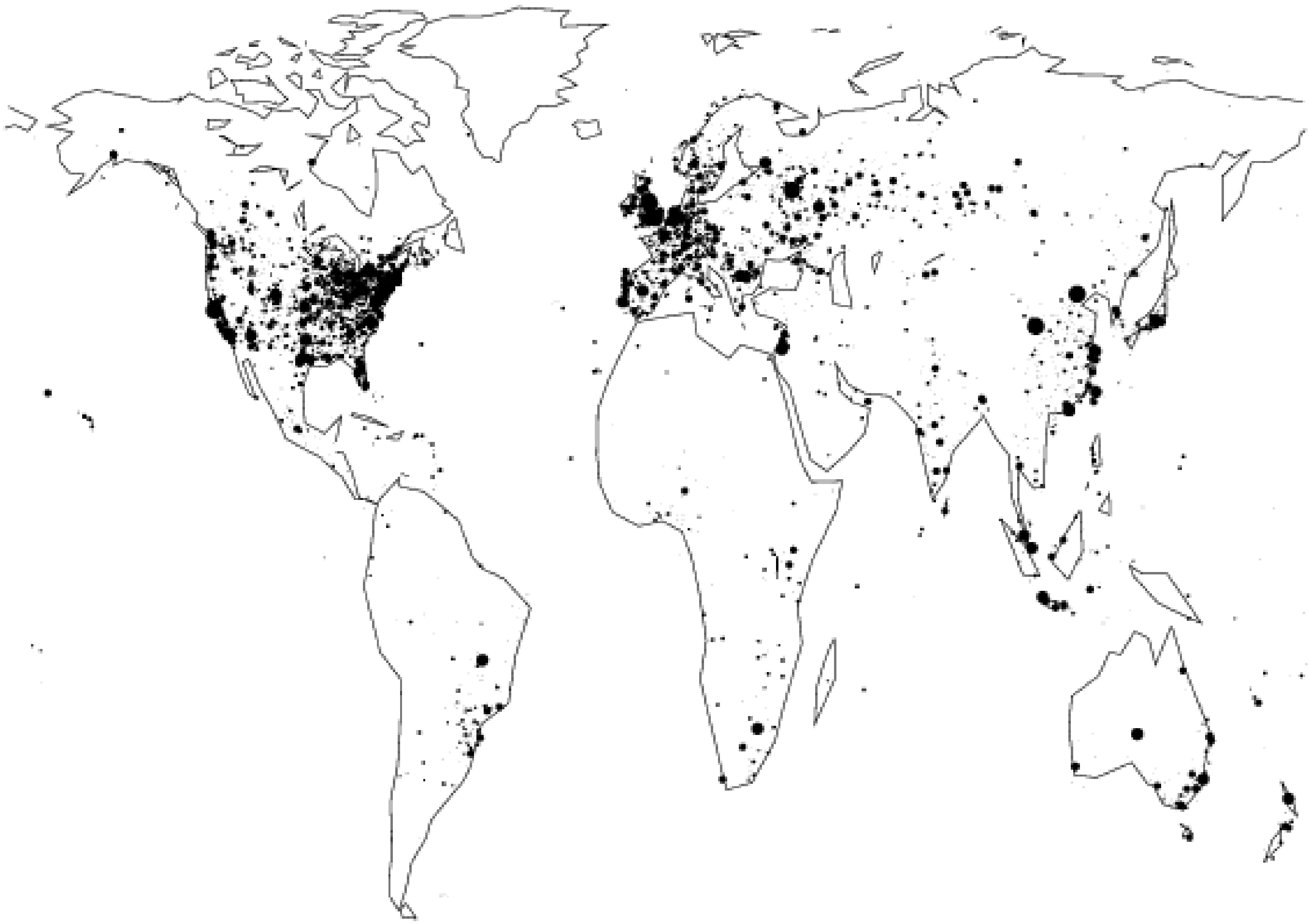
- Plot CDF of β for every node
- This gives us an idea of how “important” a node is

Results: LINX

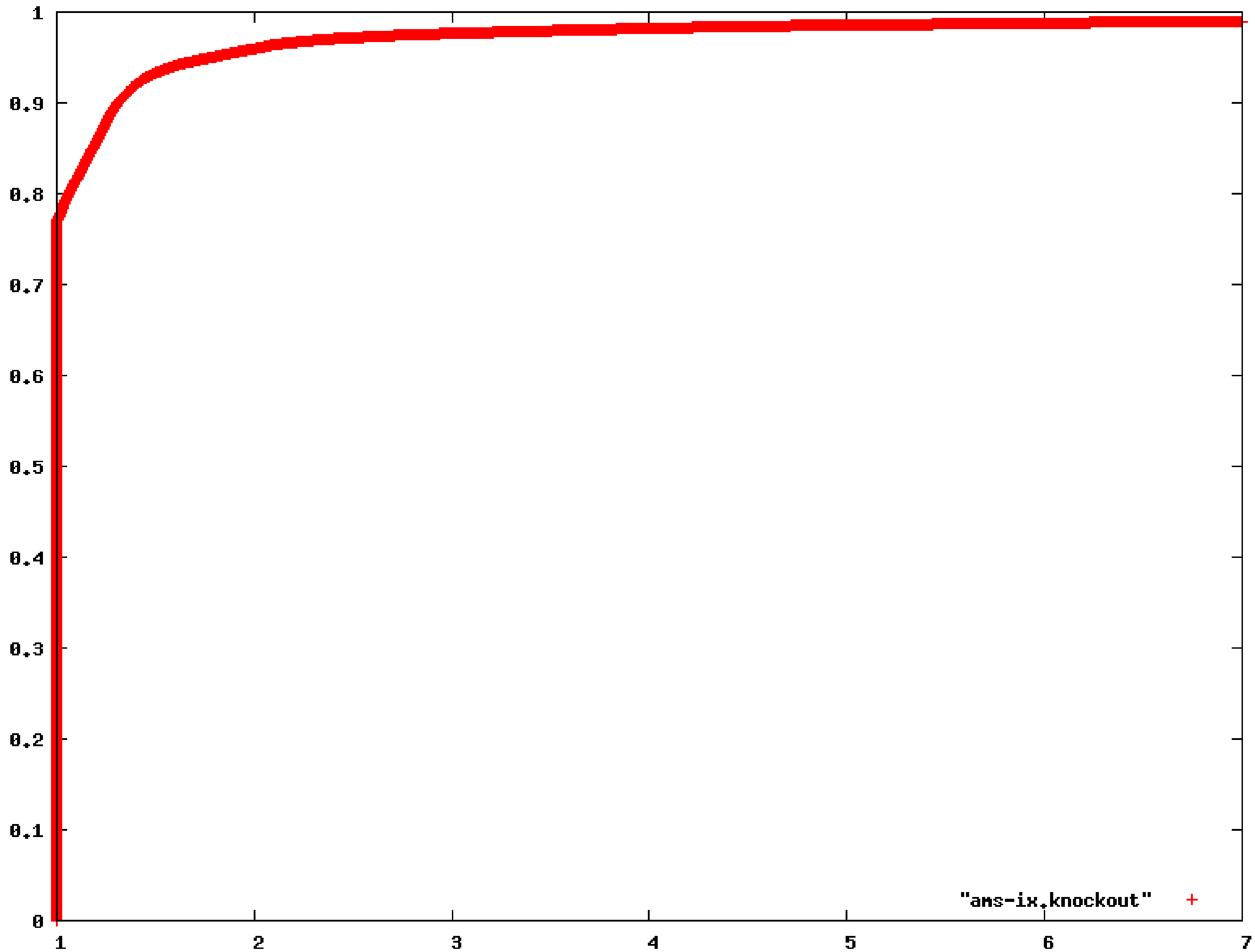


Not much benefit on its own

Geographic distribution: LINX



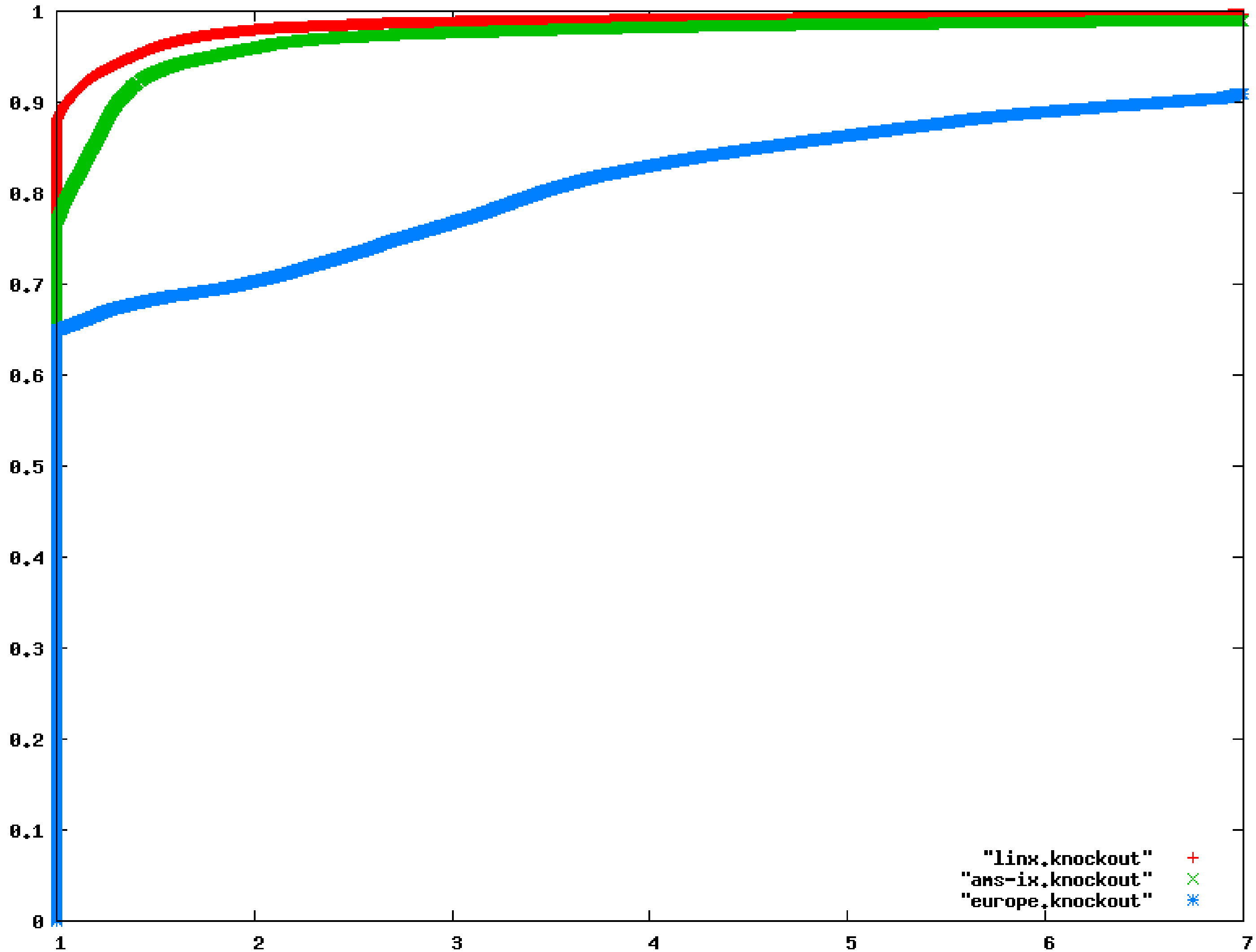
Results: AMS-IX



Not much benefit on its own

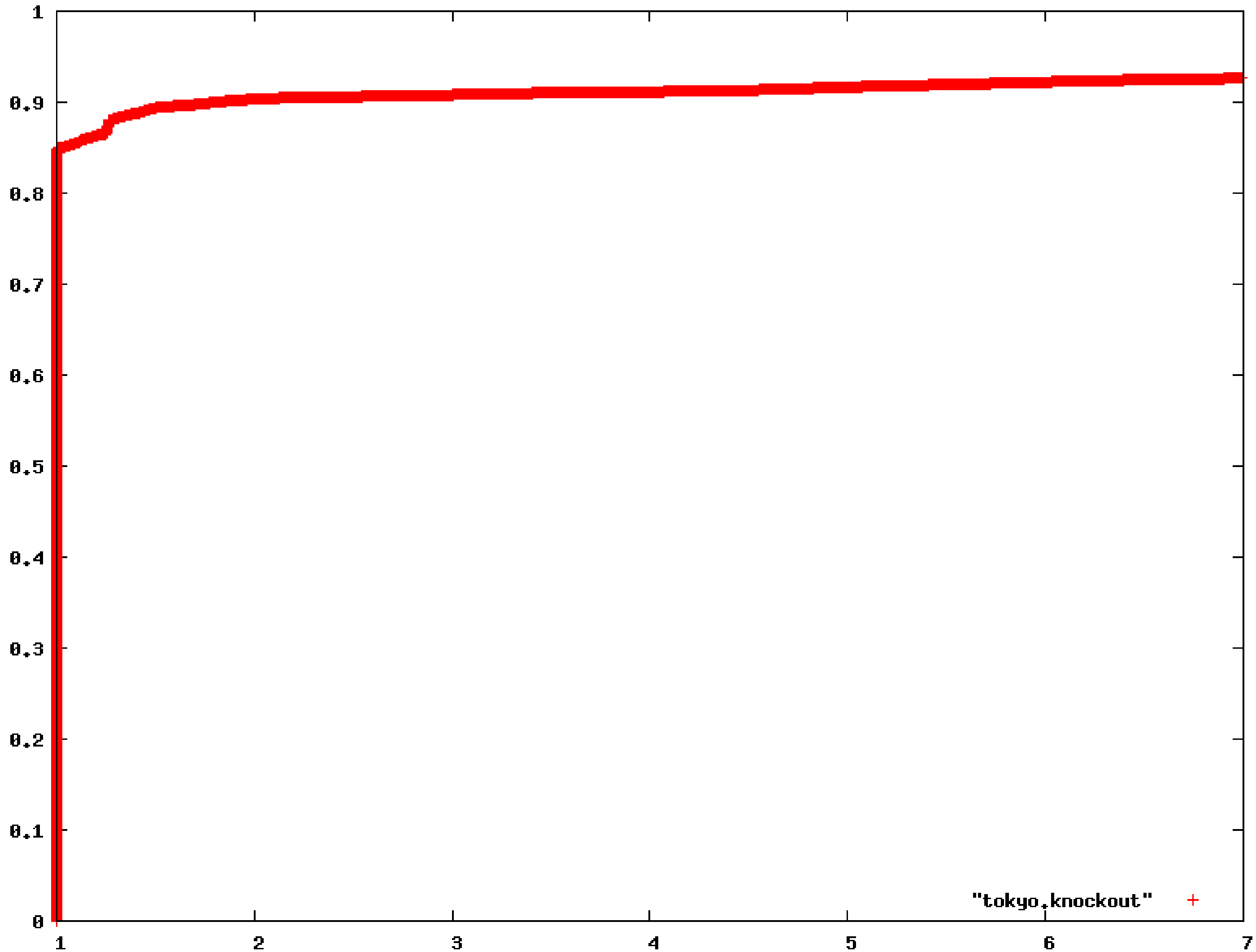
Geographic distribution: AMS-IX

Results: LINX and AMS-IX



But wait, LINX and AMS-IX are important taken together…

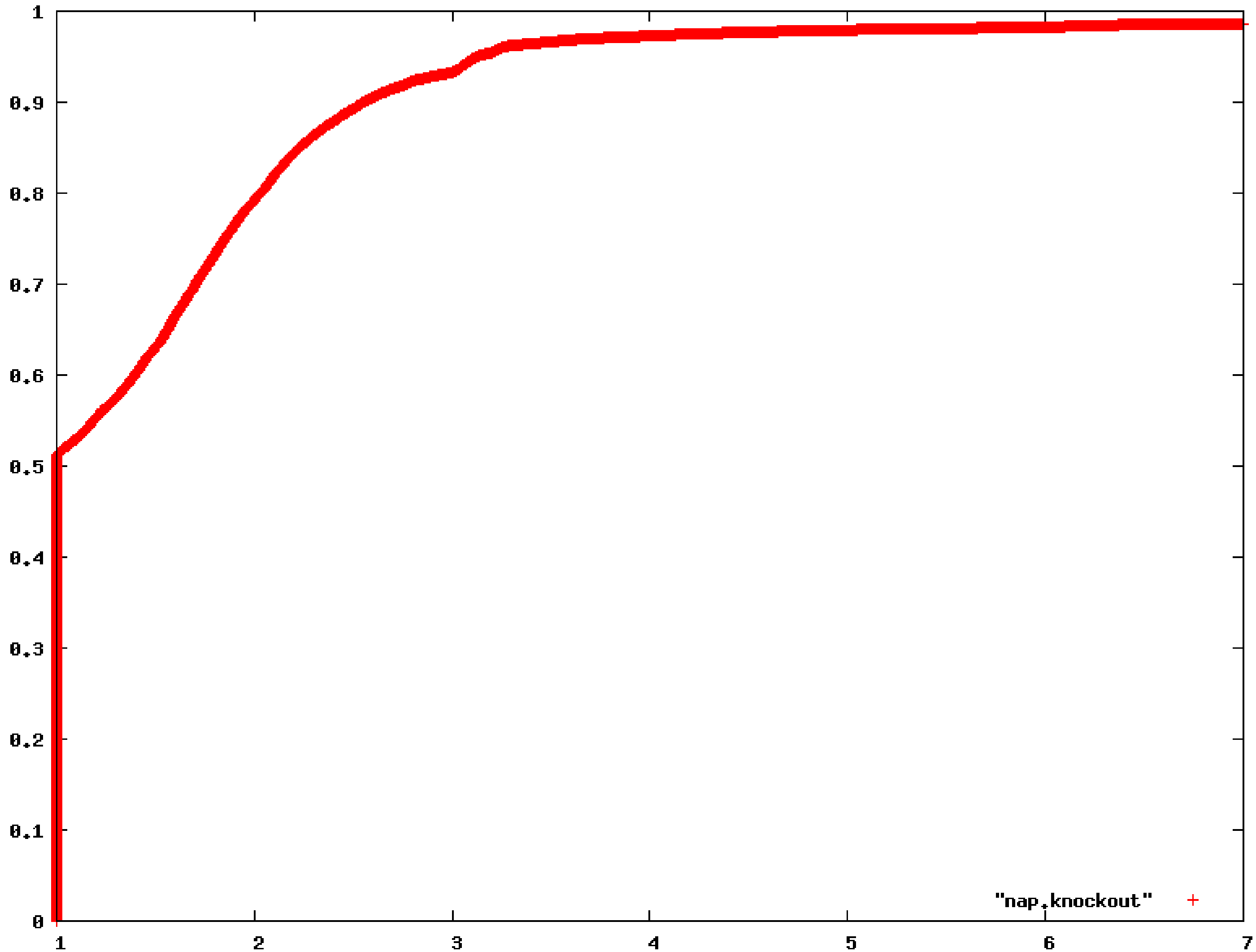
Results: Tokyo



Few clients, but very badly served by other nodes

Geographic distribution: Tokyo

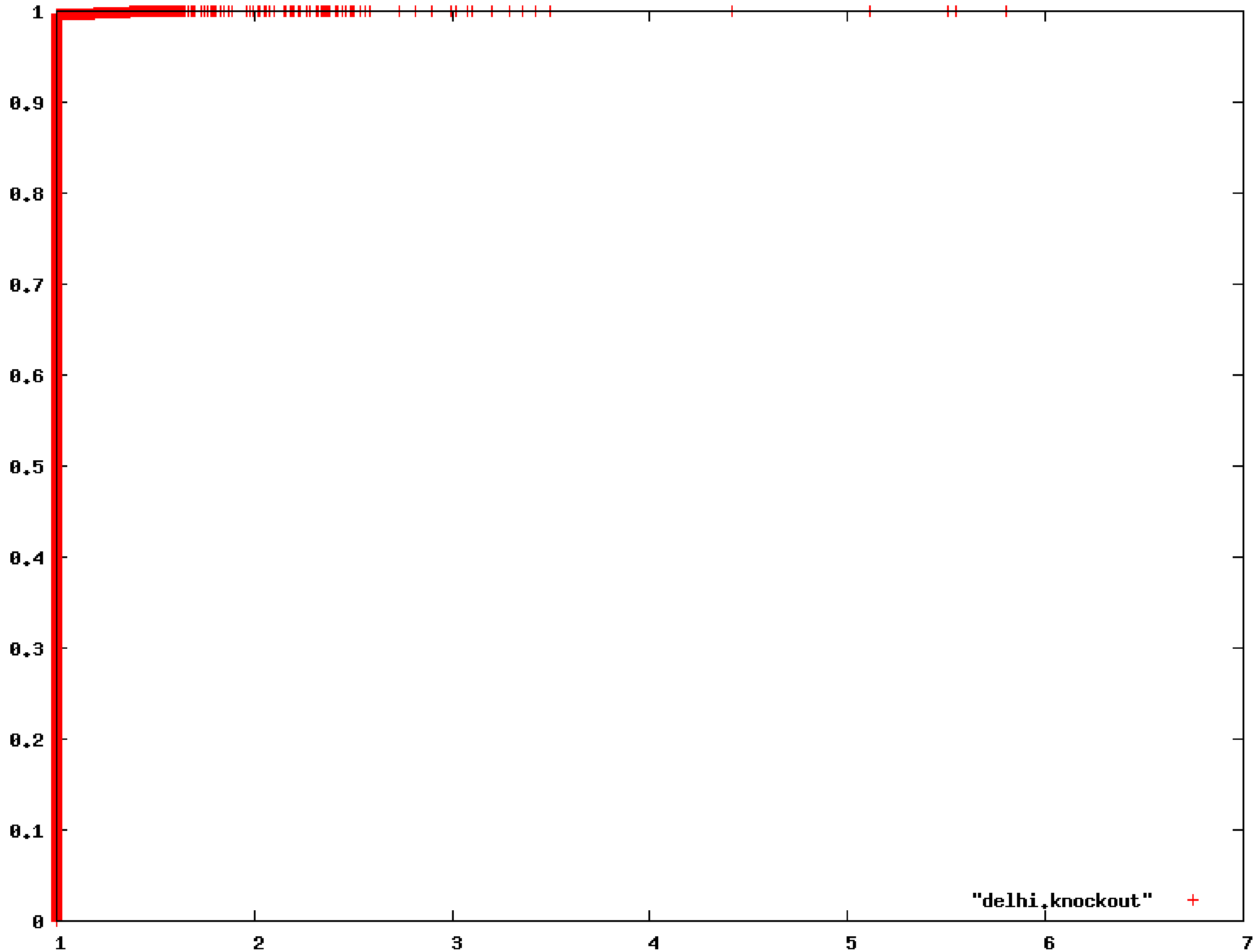
Results: Miami



Moderately better for some clients

Geographic distribution: Miami

Results: Delhi



Not very effective

Geographic distribution: Delhi

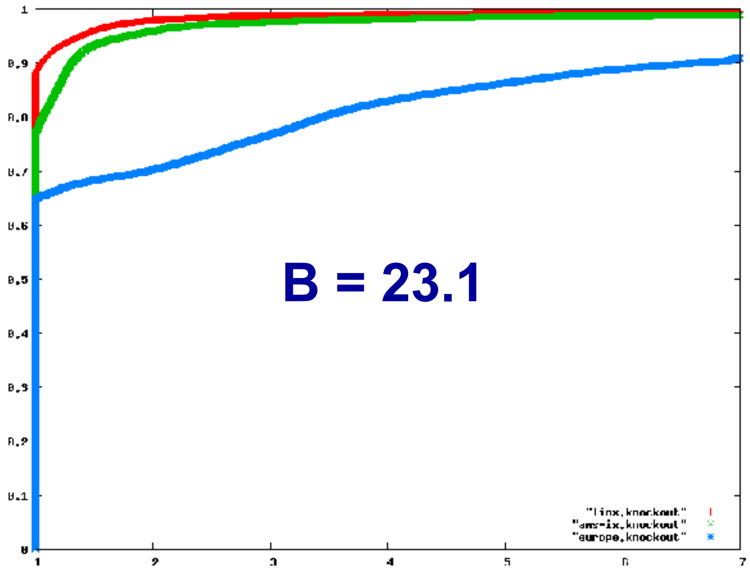
Incremental benefit of a node

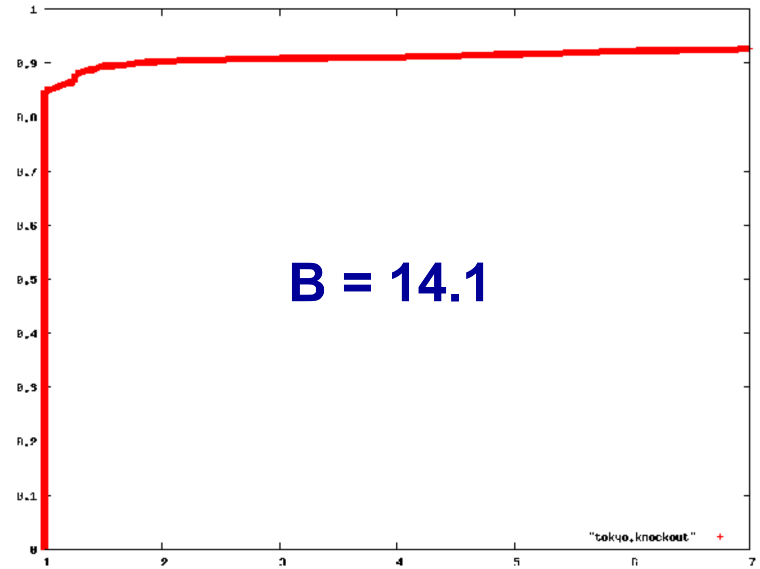
- Take β values for all clients
- Take the weighted average, where the weights are the number of queries seen by each client

$$B = \frac{\Sigma_i \beta_i Q_i}{\Sigma_i Q_i}$$

Values of B

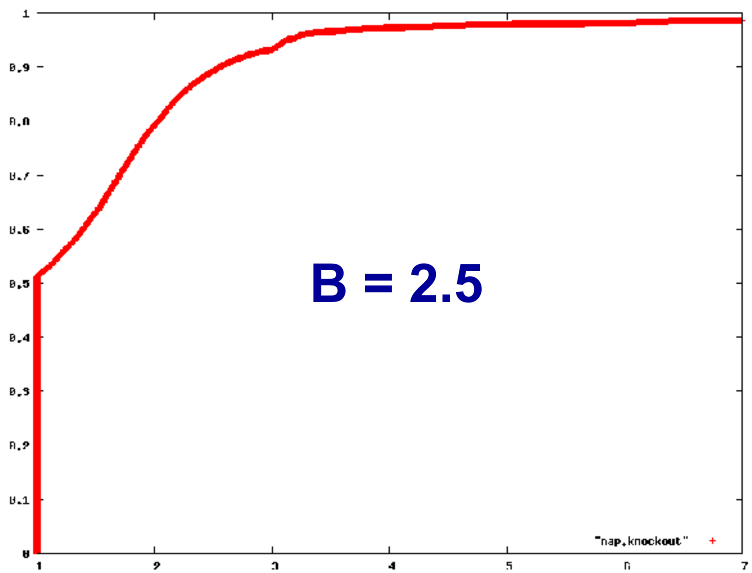
Europe

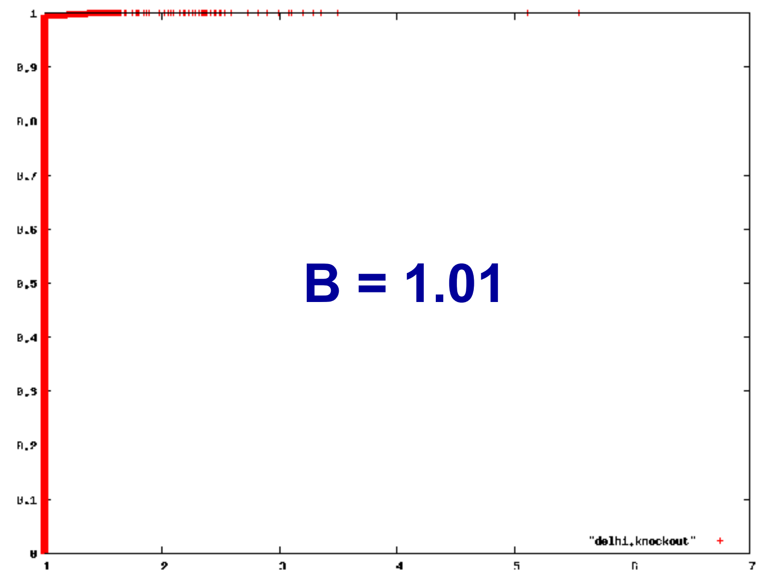




Tokyo

NAP

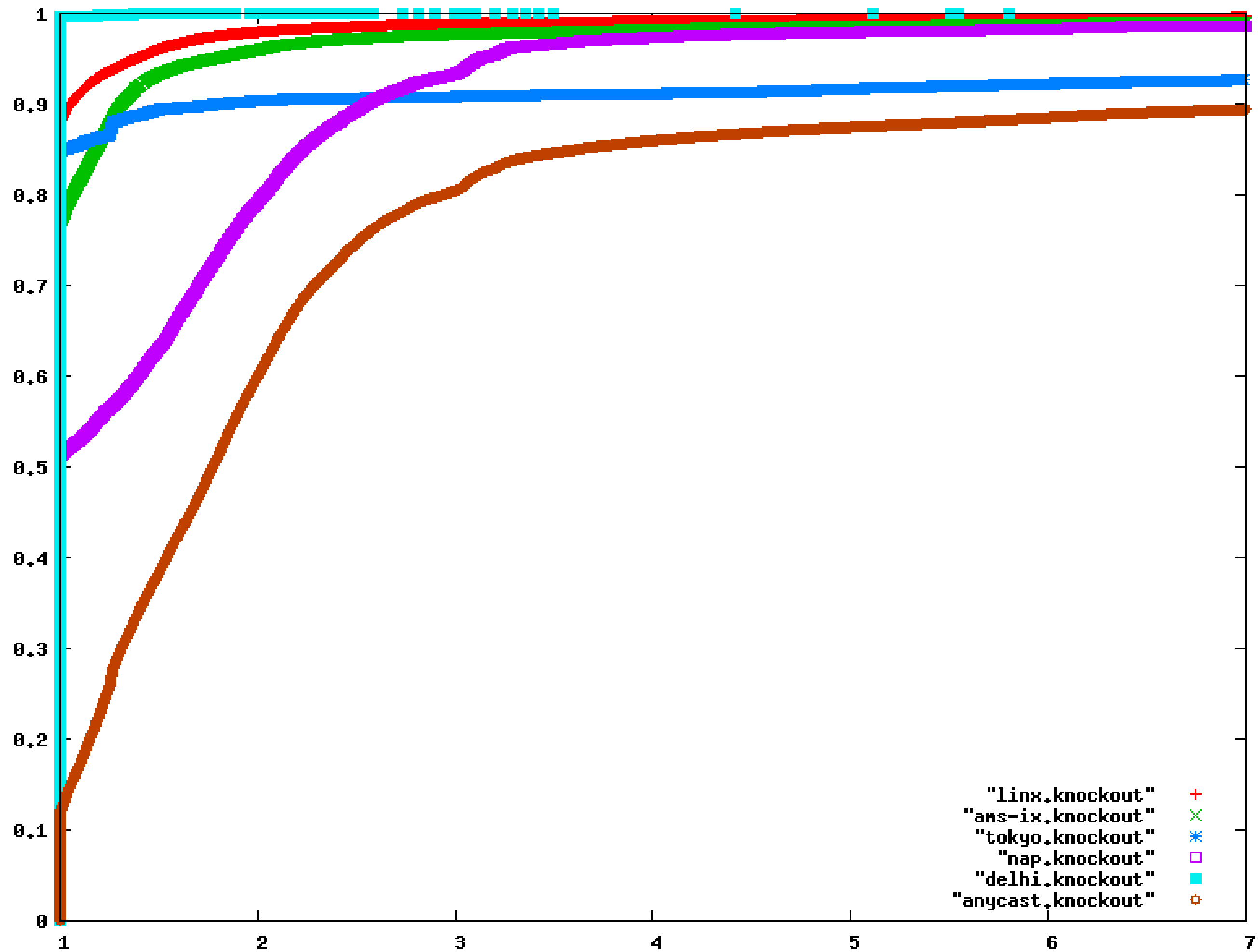




Delhi

Does anycast provide any benefit?

- What if we didn't do anycast at all?
- Knock out all except LINX: dark red curve
- B = 18.8
- For K, anycast **works well**



Stability

Stability

- What about stability?
  - The more routes competing in BGP, the more churn
  - Doesn’t matter for single-packet exchanges (UDP)
  - Does matter for TCP queries
- How frequent are node switches?

- Measure at the server
- Look at node switches that actually occur

Measuring node switches

- Methodology:
  - Look at packet dumps
    - At the time, there were only 2 global nodes
  - Extract all port 53/UDP traffic
  - For each IP address, remember where it was last seen
  - If the same IP is seen elsewhere, log a switch

- Caveats:
  - K nodes are only NTP synchronized

Node switch results for K

2 nodes (April 2005)

- 24 hours of data:
  - 527,376,619 queries
  - 30,993 switches (~0.006%)
  - 884,010 IPs seen
  - 10,557 switchers (~1.1%)

5 nodes (April 2006)

- ~5 hours of data:
  - 246,769,005 queries
  - 150,938 switches (0.06%)
  - 845,328 IPs seen
  - 2,830 switchers (0.33%)

Does not seem a serious problem for K clients

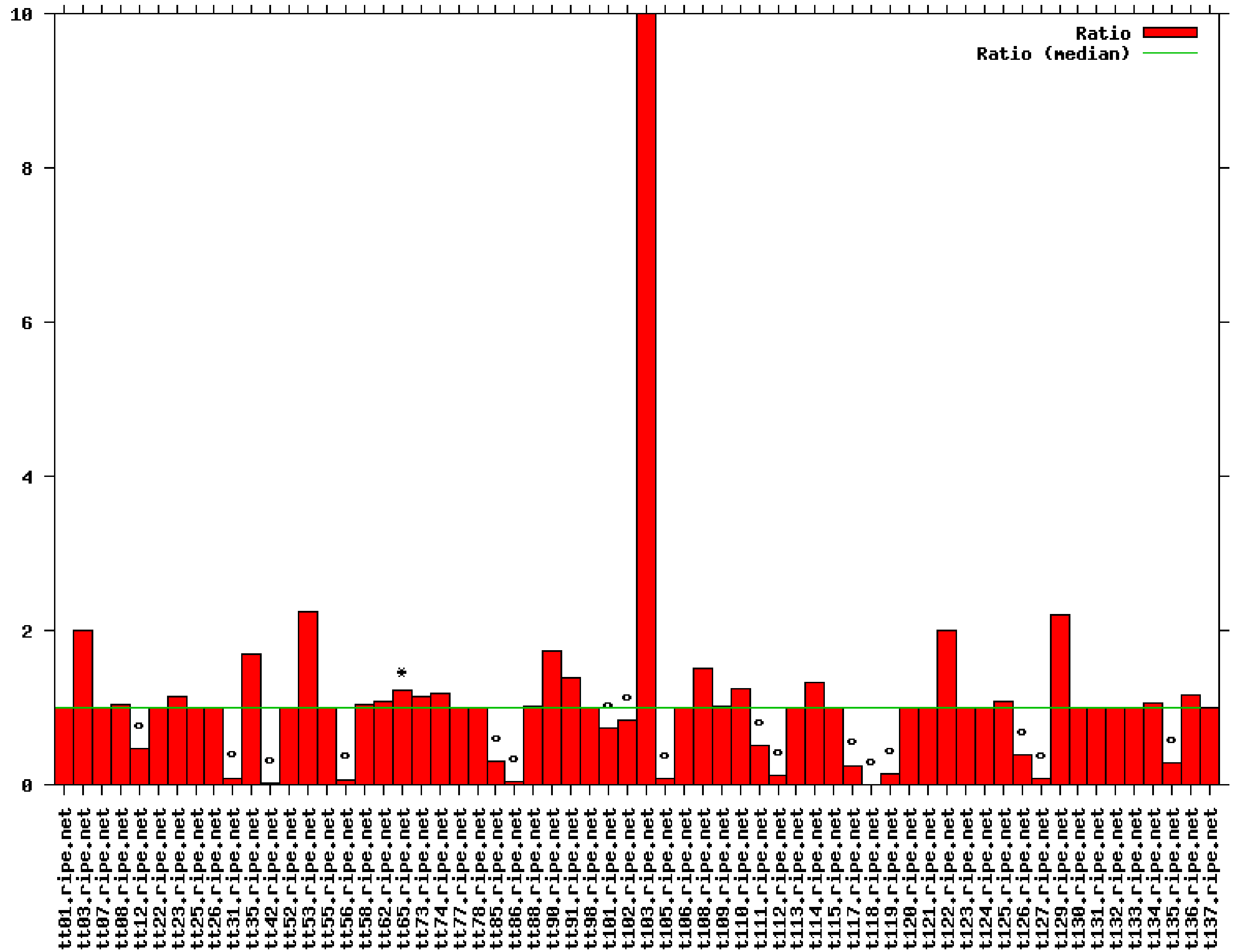
Routing issues

Routing issues

- K-root deployment structure:
  - 5 global nodes (prepended)
    - LINX, AMS-IX, Tokyo, Miami, Delhi
  - 12 local nodes (announced with no-export)
    - Frankfurt, Athens, Doha, Milan, Reykjavik, Helsinki, Geneva, Poznan, Budapest, Abu Dhabi, Brisbane, Novosibirsk

- Different prepending values can lead to high latency
- No-export can cause problems:
  - Loss of reachability if honored
  - Bad performance if ignored

Different prepending values



Prepending problems

```
results/200604120000 $ cat tt103.ripe.net
193.0.14.129 k1.delhi 422 k1.delhi 416 k1.delhi 423 k1.delhi 428 k1.delhi 419
[...]
203.119.22.1 k1.tokyo 2 k1.tokyo 2 k1.tokyo 2 k1.tokyo 2 k1.tokyo 2
```

- tt103 is in Yokohama
  - Tokyo is 2ms away
    - But it goes to Delhi
    - ... through Tokyo, Los Angeles and Hong Kong

- RTT = 416 ms, $\alpha = 208$

Problem: different prepending lengths

- Got BGP paths from AS2497
  - Thanks to Matsuzaki and Randy Bush
- Problem: bad interaction of different prepending lengths
  - Tokyo:
    - 2914 25152 25152 25152 25152
    - 4713 25152 25152 25152 25152
    - 6461 25152 25152 25152 25152
  - Delhi:
    - 2200 9430 25152 25152
- We need to fix prepending on Tokyo node

No-export and leaks

- Local nodes can be worse than global nodes

```
$ cat tt89
193.0.14.129 k2.denic 29 k2.denic 30 k2.denic 29 k2.denic 30 k2.denic 29
193.0.16.1 k1.linx 4 k1.linx 3 k1.linx 3 k1.linx 3 k1.linx 3
193.0.16.2 k2.linx 3 k2.linx 3 k2.linx 3 k2.linx 3 k2.linx 4
193.0.17.1 k1.ams-ix 12 k1.ams-ix 11 k1.ams-ix 12 k1.ams-ix 13 k1.ams-ix 13
193.0.17.2 k2.ams-ix 12 k2.ams-ix 13 k2.ams-ix 11 k2.ams-ix 12 k2.ams-ix 13
```

- What's going on here?
  - Local node announcements get announced to customers

    …and customers of customers
  - They compete with announcements from global nodes

    …which lose out due to prepending

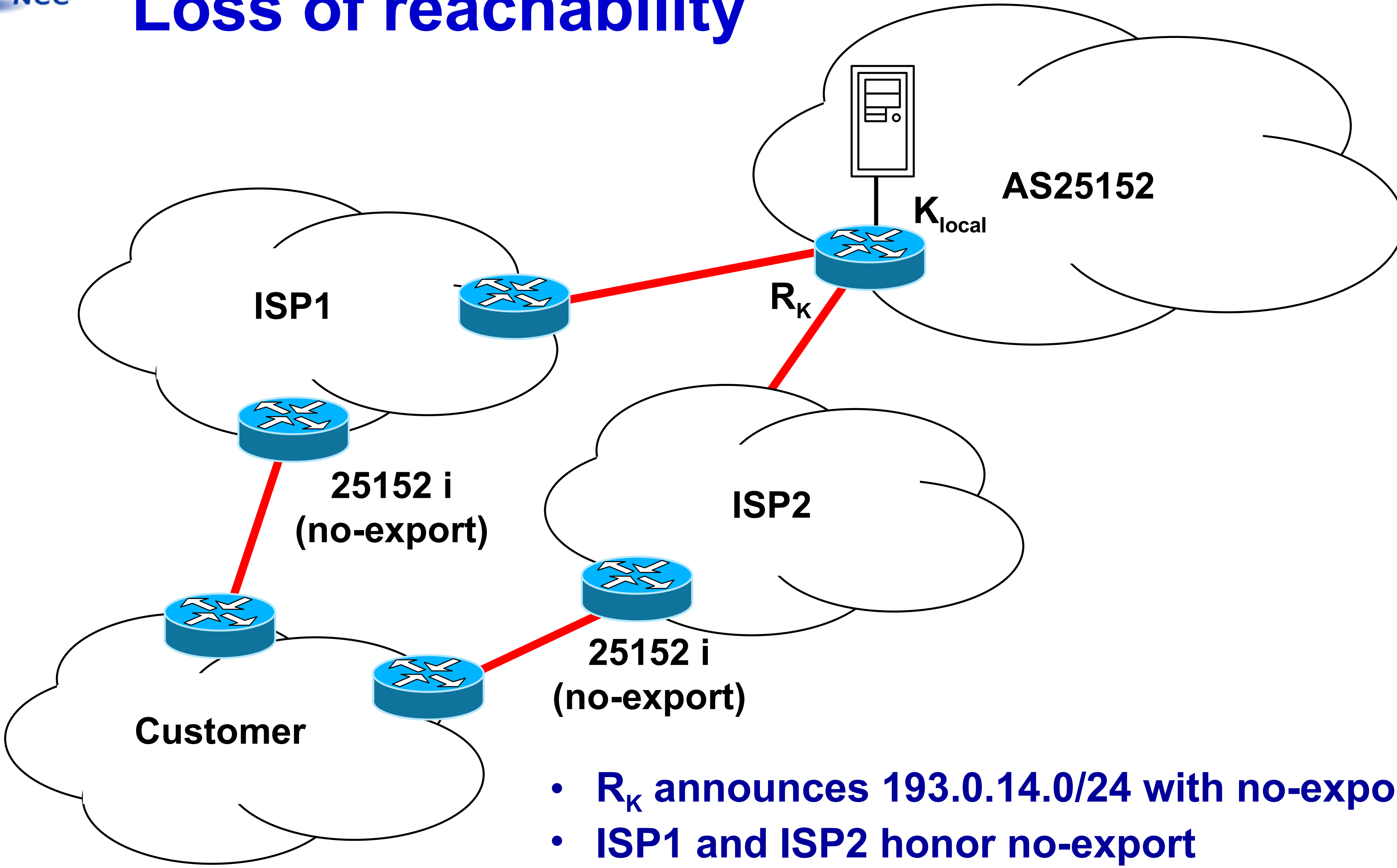
No-export and loss of reachability

- Problem pointed out by Randy Bush
  http://www.merit.edu/mail.archives/nanog/2005-10/msg01226.html

- Problematic interaction of no-export with anycast
  - We use no-export to prevent local nodes from leaking
  - But if we have an AS:
    - Whose providers all peer with a local node
      - And honor no-export
  - They might see no route at all!

- Fixed by announcing a less-specific from AMS-IX node
  - The customer will choose an upstream based on that ISP and reach the local node chosen by that ISP

Loss of reachability



- R_K announces 193.0.14.0/24 with no-export
- ISP1 and ISP2 honor no-export
- Customer has no route to 193.0.14.0/24

Questions?