The Spoofer Project

Rob Beverly and Steve Bauer
{rbeverly,bauer}@mit.edu
MIT ANA

NANOG34

Background

Spoofing

- Attackers/compromised hosts forge or “spoof” source address of an IP packet for:
  - Anonymity
  - Reflector attacks [Paxson01]
  - BGP/TCP Resets
- High-profile spoofing-based DDoS attacks in 2000-2004:
  - Yahoo, Ebay, E*trade
  - Shaft, TFN, trinoo, Stacheldraht, RingZero
  - Protx online payment site, Nov 2004

Spoofing

- Does Spoofing *matter* in 2005?
  - All ISP filter, right? (RFC2827, uRPF)
  - Zombie Farms (little additional anonymity)
  - Prevalence of NATs (headers rewritten, spoofing useless)
- Backscatter [Moore01][Pang04] shows ***continued, strong spoofing activity***

The Spoofer Project

- Tracking Spoofs is *operationally difficult*:
  - [Greene, Morrow, Gemberling NANOG 23]
  - ICMP traceback [Bellovin00]
  - Hash-based IP traceback [Snoeren01]
- Enter: The Spoofer Project
- Internet-wide active measurement project:
  - Quantify extent and nature of source address filtering on the Internet

The Spoofer Project

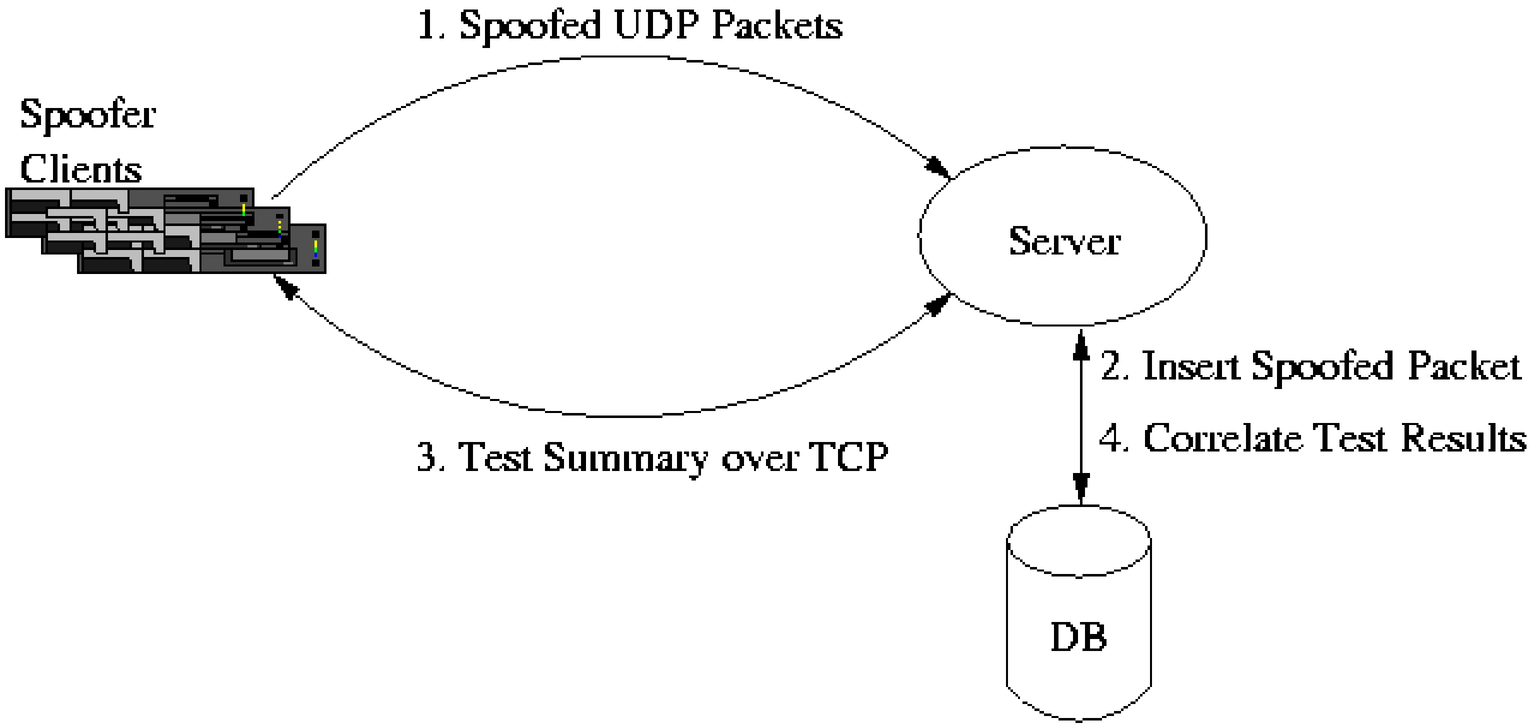
http://momo.lcs.mit.edu/spoofer

- Clients run “spoofer” program:
  - Binaries, source publicly available
- Availability advertised to e.g. NANOG, CAIDA, dshield, etc. mailing lists

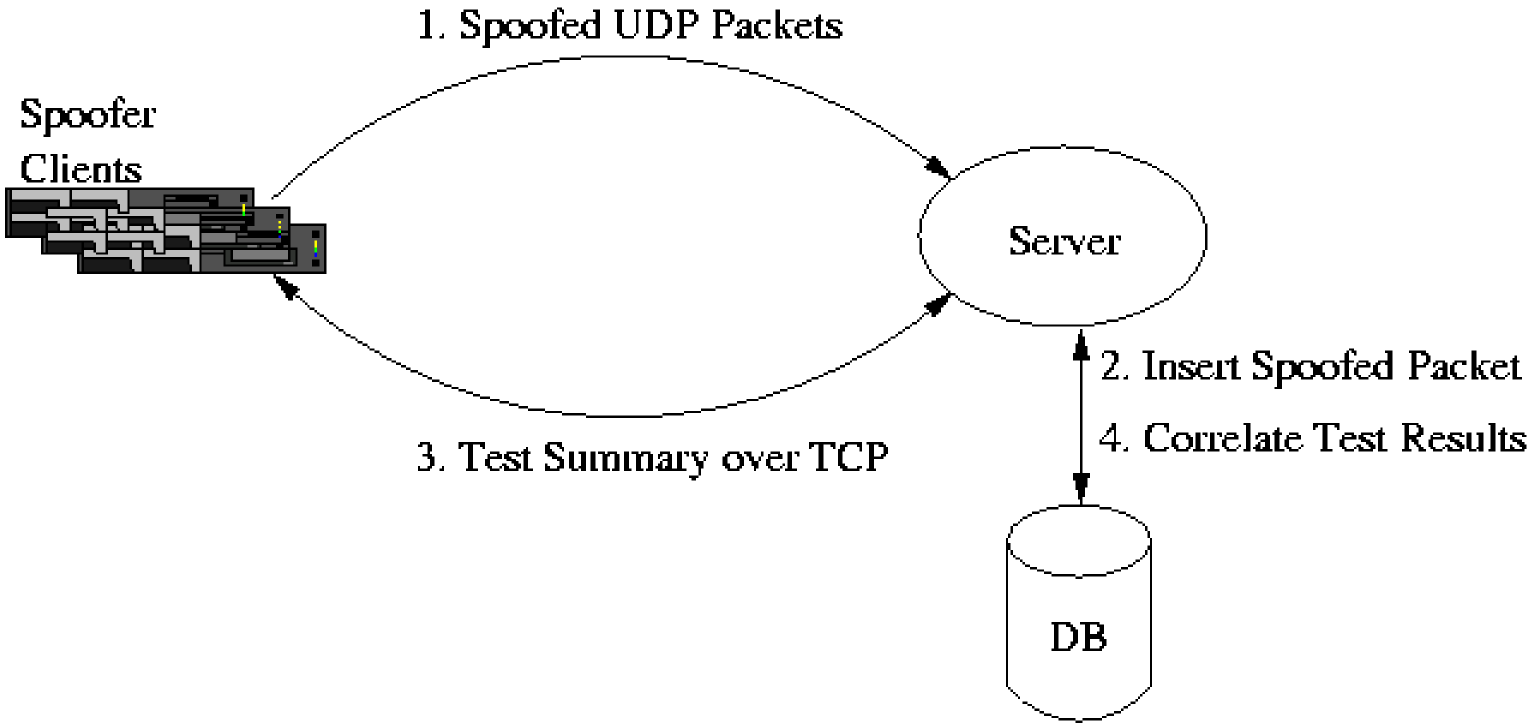
Spoofer Project: Key Results

- ~23% of observed netblocks corresponding to ~24% of observed ASes allow spoofing
- Filtering is frequently applied inconsistently or with automated methods that allow partial spoofing

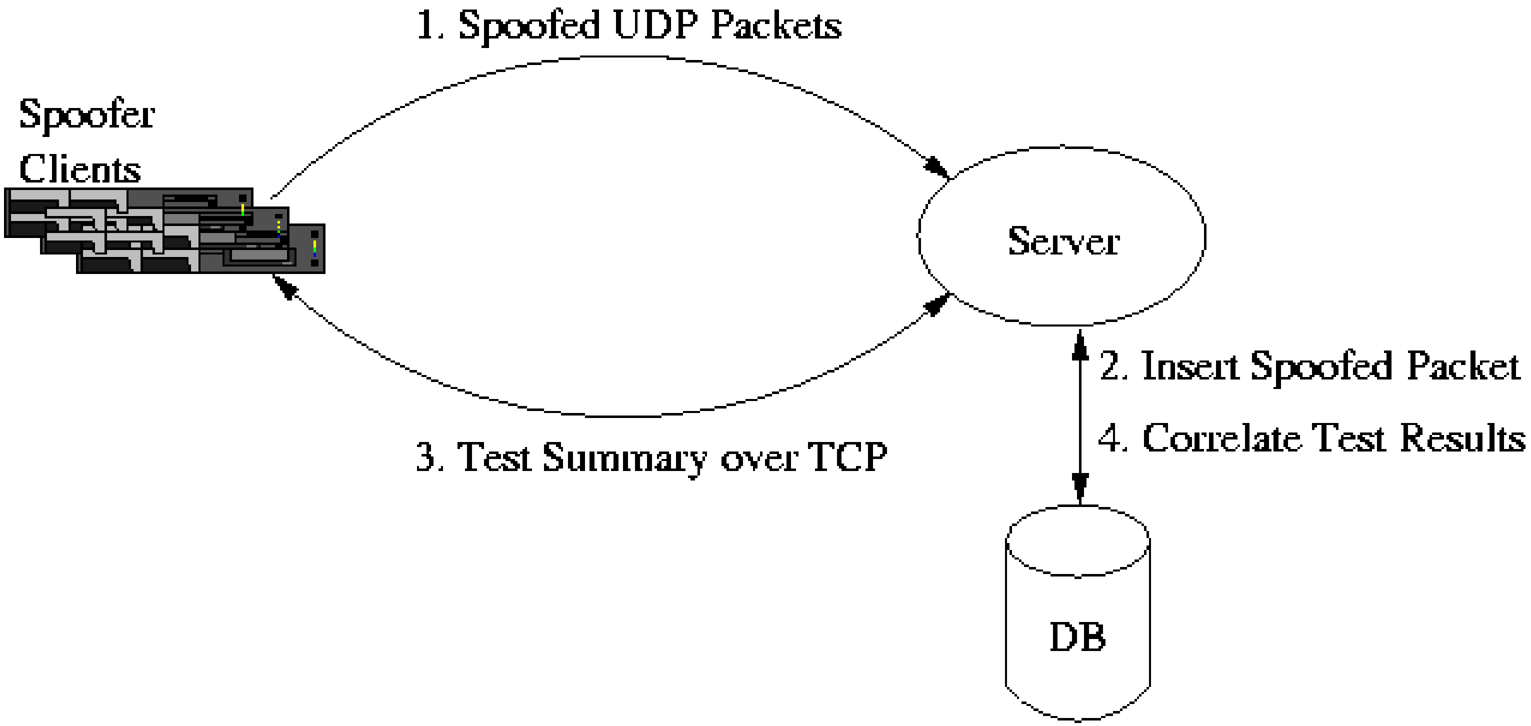
Spoofer Operation



- Send series of spoofed UDP packets to server on campus
  - Five of each with random inter-packet delay
  - Payload includes unique 14 byte identifier
  - If received, packets stored in DB



- Send TCP report of spoofed packets to server
- Send traceroute to server
- Use UDP port 53, TCP port 80 to avoid secondary filtering effects



- Identifiers allow us to disambiguate received packets
- Analysis, web pages driven off DB

Spoofed Packets

- Chosen to infer specific filtering policies

Spoofed Source	**Description**
1.2.3.4	Bogon (Not in BGP table)
6.1.2.3	Valid (In BGP table)
172.16.1.100	Martian (RFC1918 private address)
IP $\oplus$ (2^N) for 24>N>0	Neighbor Spoof

Example Client Run

```
[root@coco spoofer]# ./spoofer
>> Spoofing Tester v0.2
>> Source 5 spoofed packets (IP: 1.2.3.4) (Seq: g8cb4gc6ojezw1)...
>> Source 5 spoofed packets (IP: 172.16.1.100) (Seq: 09kamtjjugxwvy)...
>> Source 5 spoofed packets (IP: 6.1.2.3) (Seq: 0dzpw2obc80ff3)...
>>
>> Checking spoofing result...
>> Server response: HOWDY 5am11w18zzc86g
>> Server response: COOL 3
>> Server response: FOUND g8cb4gc6ojezw1
>> Server response: FOUND 09kamtjjugxwvy
>> Server response: FOUND 0dzpw2obc80ff3
>> Running Trace (please wait): /usr/sbin/traceroute -n 18.26.0.235
traceroute to 18.26.0.235 (18.26.0.235), 30 hops max, 38 byte packets
>> Server response: SEND-TRACE LINUX
>> Server response: BYE 5am11w18zzc86g

Test Complete.
Your test results:
  http://momo.lcs.mit.edu/spoofer/report.php?sessionkey=5am11w18zzc86g
```

Results

Results

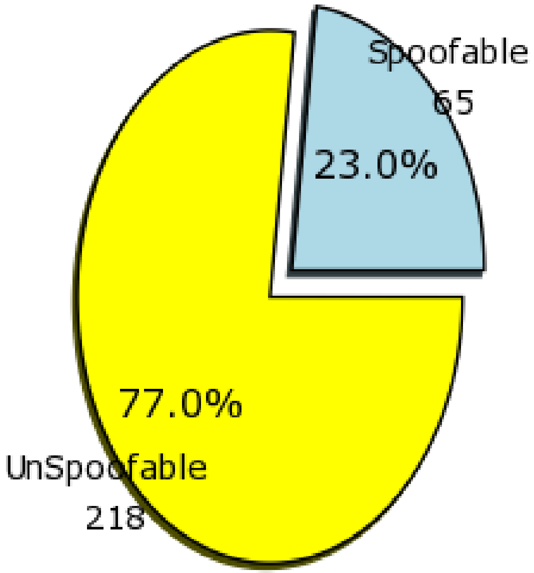
- From March 2005 to Present:
  - 566 client reports
  - 438 unique
- Netblocks for which we receive reports, but see no evidence of spoofing are labeled “believed unspoofable”
- Use U Oregon Routeviews tables to determine prefix size, AS, etc.

Failed Spoofs

- We exclude these from our results:
- Raw socket blocked by WinXP SP2: 118 of 209 reports
- Socket blocked by other OS: 19 clients
- Hosts behind NATs: 108
- Totals: 269, approximately 2/3rds, failed to spoof any packets

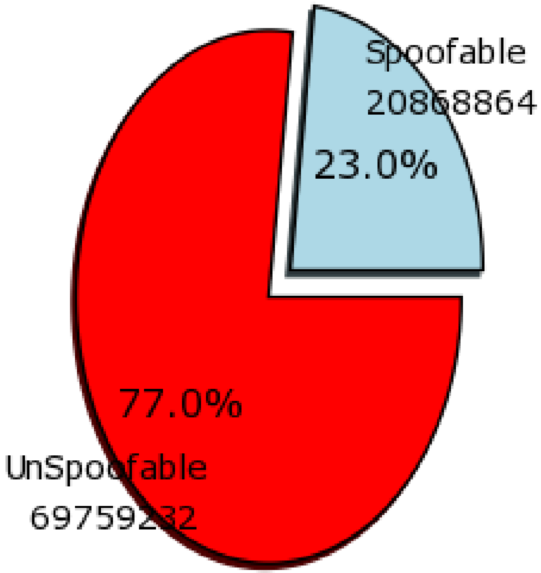
Spoofing Coverage

Netblocks



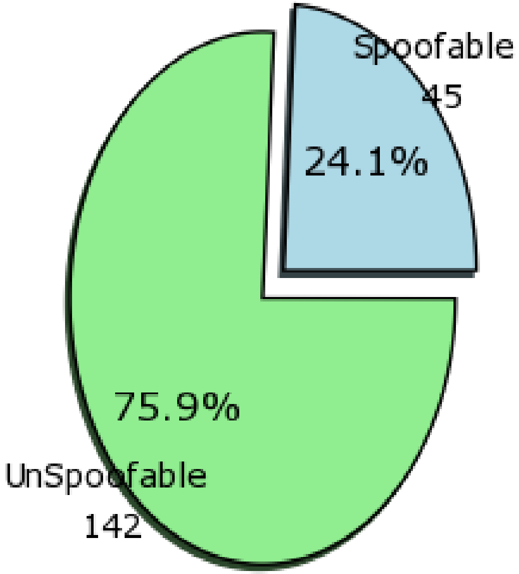
Estimated
50350 out of 168868
Netblocks Spoofable

IP Addresses



Estimated
477 million out of 1.59 billion
IP Addresses Spoofable

Autonomous Systems



Estimated
5700 out of 18009
ASes Spoofable

Inconsistent Filtering

- Block only RFC1918 (time-invariant, static policy easy to maintain)
- Automated filtering:
  - Spoof only valid (IANA assigned, in routing table) addresses
  - Cymru bogon route-server project

Frequency of Inconsistent Filtering

RFC1918	**Bogon**	**Valid**	**Count**
-	-	NF	17
-	NF	-	0
-	NF	NF	49
NF	-	-	0
NF	-	NF	0
NF	NF	-	0

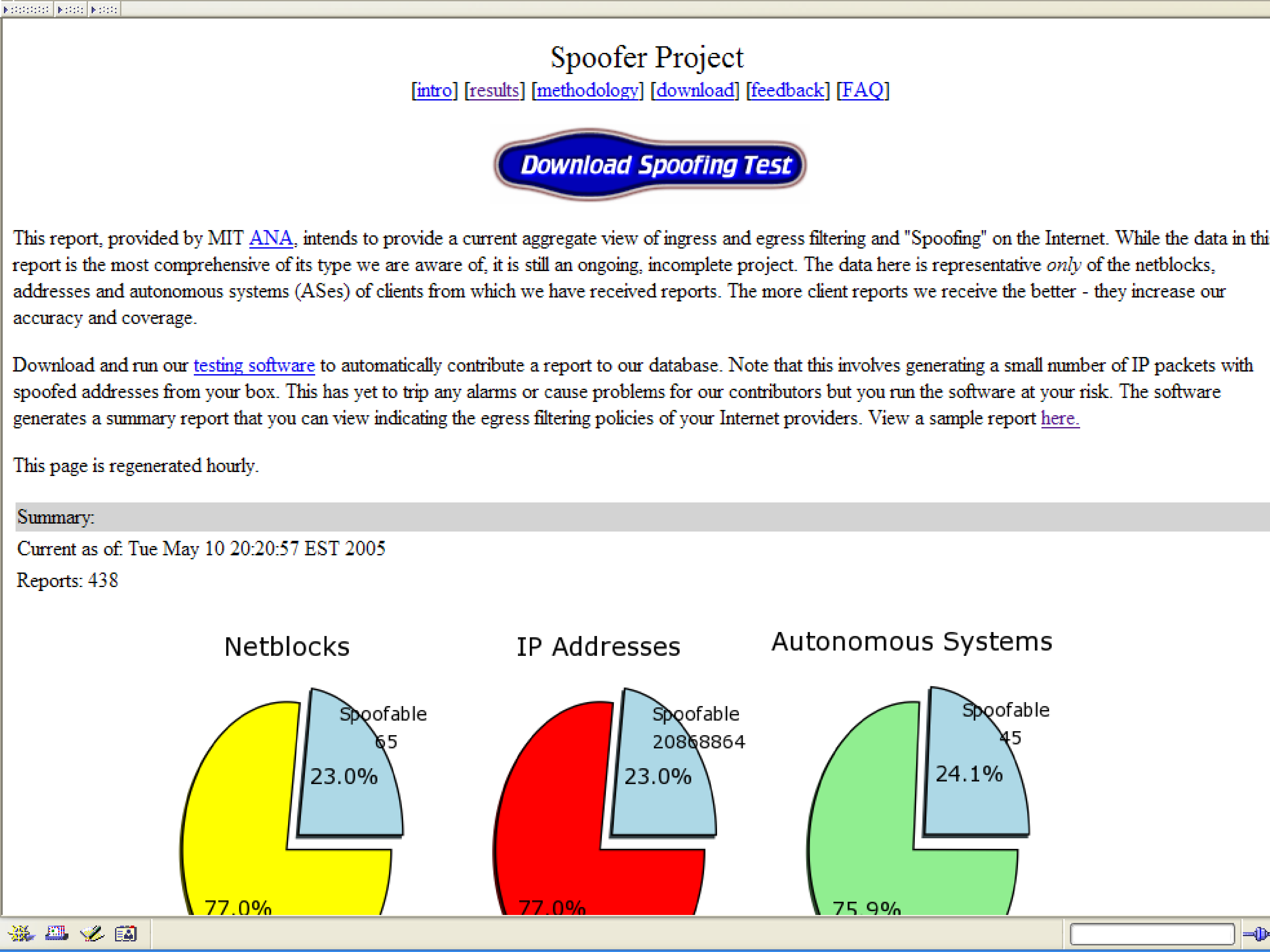
** NF = Not Filtered

Example: providers that automate filtering by only forwarding packets sourced with valid address (in BGP table)

State of IP Spoofing

http://momo.lcs.mit.edu/spoofer/summary.php

- Hourly-updated web page
- Summarizes current state of IP spoofing
- Goal: continue collecting reports to improve accuracy, detect trends, etc.
- We need operator help to expand coverage and gain more data!



Spoofer Project

[intro] [results] [methodology] [download] [feedback] [FAQ]

Download Spoofing Test

This report, provided by MIT ANA, intends to provide a current aggregate view of ingress and egress filtering and "Spoofing" on the Internet. While the data in thi report is the most comprehensive of its type we are aware of, it is still an ongoing, incomplete project. The data here is representative *only* of the netblocks, addresses and autonomous systems (ASes) of clients from which we have received reports. The more client reports we receive the better - they increase our accuracy and coverage.

Download and run our testing software to automatically contribute a report to our database. Note that this involves generating a small number of IP packets with spoofed addresses from your box. This has yet to trip any alarms or cause problems for our contributors but you run the software at your risk. The software generates a summary report that you can view indicating the egress filtering policies of your Internet providers. View a sample report here.

This page is regenerated hourly.

Summary:

Current as of: Tue May 10 20:20:57 EST 2005

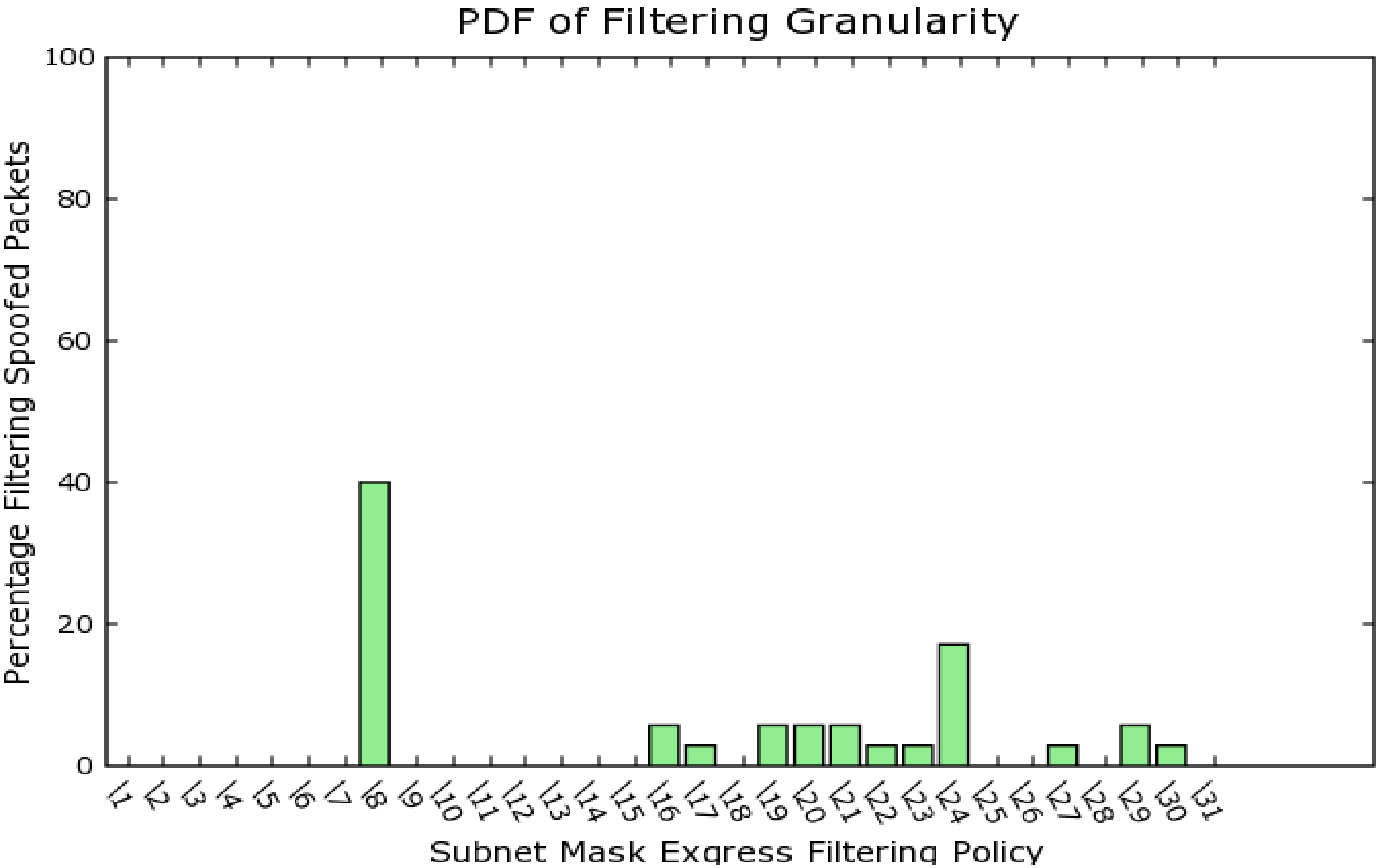
Reports: 438

Filtering Granularity

Filtering Granularity

- In general, always able to spoof immediate neighbor IP address (your address +/- 1)
- Neighbor spoofing test allows us to infer the filtering boundary

Filtering Boundaries



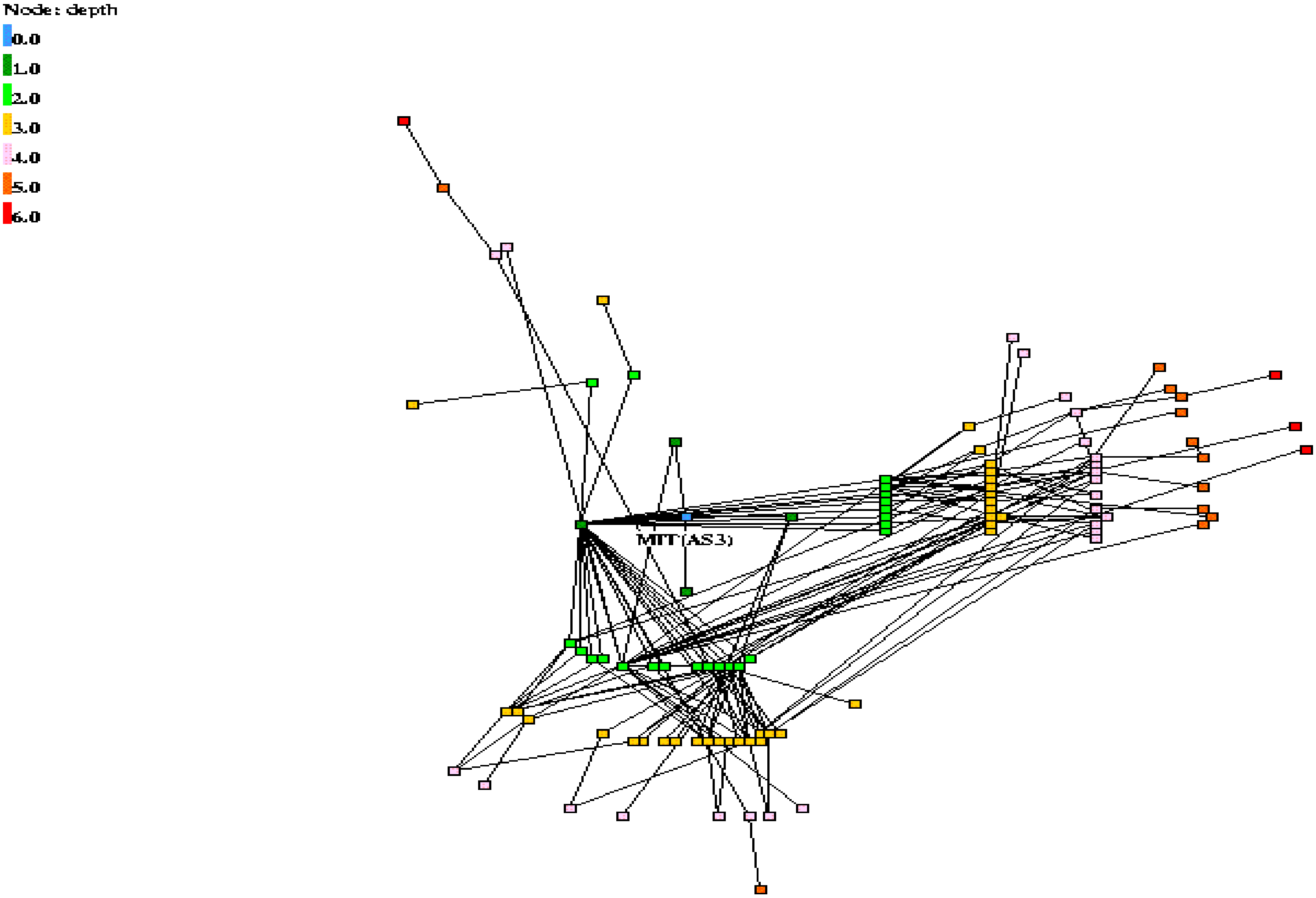
Visualizing Spoofing Perimeter

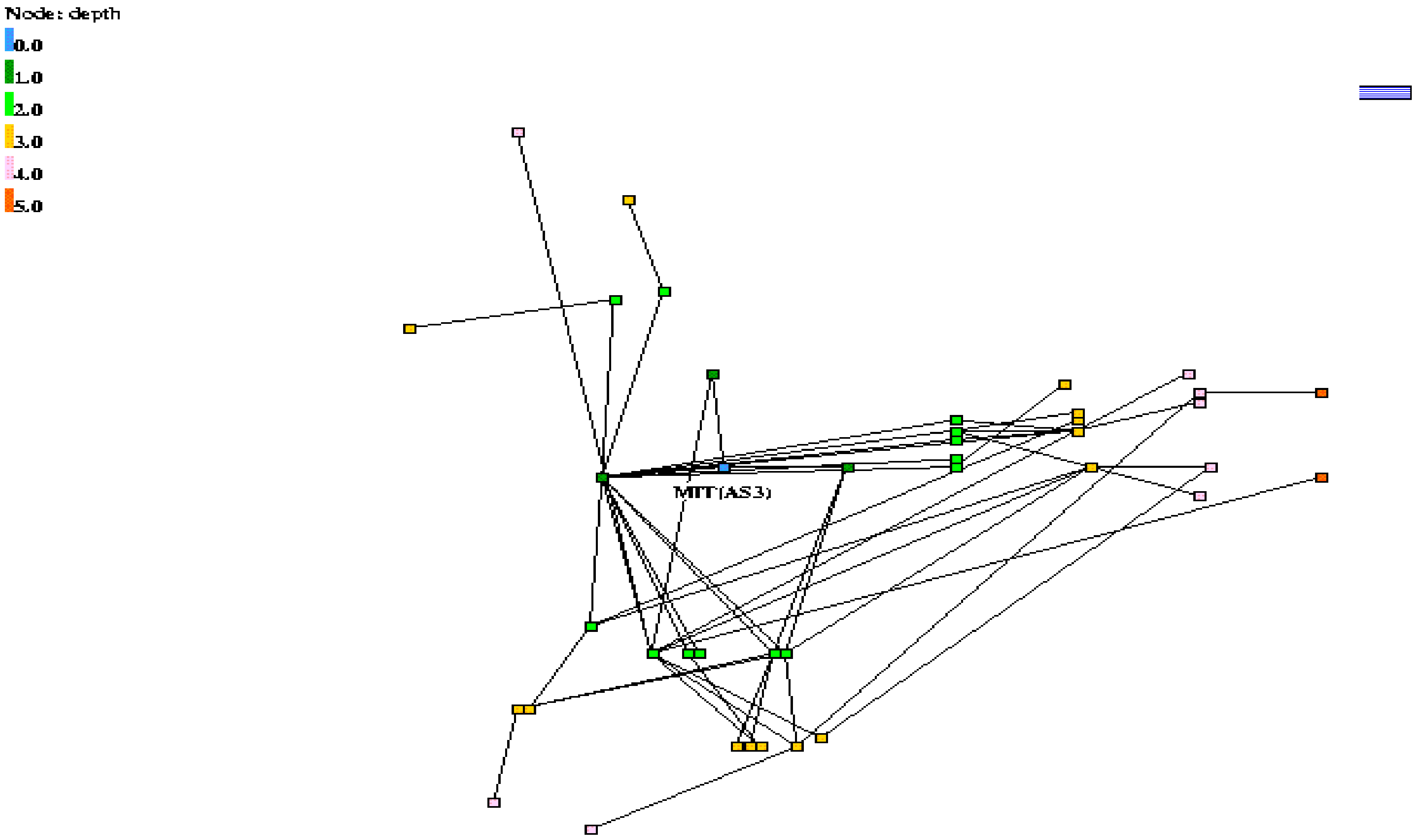
Visualizing Spoofing Perimeter

- Want to visualize:
  - Extent of spoofing
  - Spoofable paths vs. All observed paths
  - Geographic distribution of paths
- Using CAIDA’s otter tool [Huffaker99] to build AS graph
- For successful spoofs, define either AS path to our server as spoofable

Visualizing Spoofing Perimeter

- Nodes: Map each client to its AS, ASes along path to our server
- Edges: defined by AS path
- Semi-geographic coordinate system:
  - Similar to Skitter AS topology graphs
  - Our server at graph center (root)
  - Node radius: AS hop distance
  - Node degree: longitude of AS organization


Node: depth
0.0
1.0
2.0
3.0
4.0
5.0
6.0
MIT(AS3)


Node: depth
0.0
1.0
2.0
3.0
4.0
5.0
MIT(AS3)

Thank you!

http://momo.lcs.mit.edu/spoofer