Understanding the Efficacy of Deployed Internet Source Address Validation Filtering

Robert Beverly, Arthur Berger (MIT), Young Hyun, k claffy (UCSD/CAIDA)

ACM Internet Measurement Conference 2009
Spoofer Project

• Background
• Recent Relevance
• Project Methodology
• Results
• Parting Thoughts
Spoofed-Source IP Packets

• Circumvent host network stack to forge or “spoof” source address of an IP packet

• Lack of source address accountability a basic Internet weakness:
  – Anonymity, indirection [VP01], amplification

• Security issue for more than two-decades [RTM85, SB89]

• Still an attack vector?
Circa 2004...

IP source spoofing doesn’t matter!

a) All providers filter
b) All modern attacks use botnets
c) Compromised hosts are behind NATs

• Strong opinions from many:
  – Academic
  – Operational
  – Regulatory

• …but only anecdotal data
Internet-wide active measurement effort:

- Quantify the **extent** and **nature** of Internet source address filtering
- Understand real-world efficacy of available best-practice defenses
- Validate common assumption of edge filtering

Began Feb. 2005

- Understand how filtering has evolved
- Basis for driving design of more secure architectures
Spoofer Project

• Background
• Recent Relevance
• Project Methodology
• Results
• Parting Thoughts
Prediction: spoofing increasingly a problem in the future

- **Spoofed traffic complicates a defenders job**
- Tracking spoofs is operationally difficult:
  - [Greene, Morrow, Gemberling NANOG 23]
  - Hash-based IP traceback [Snoeren01]
  - ICMP traceback [Bellovin00]
- Consider:
  - Today (worst case scenario): When spoofing zombies are widely distributed, a network operator must defend against attack packets from 5% of routeable netblocks.
  - Future: if 25% of zombies capable of spoofing significant volume of the traffic could appear to come any part of the IPv4 address space
- Adaptive programs that make use of all local host capabilities to amplify their attacks
The Spoofing Problem (2009)

- DNS Amplifier Attacks
- DNS Cache Poisoning
- In-Window TCP Reset Attacks
- Bots that probe for ability to spoof
- Spam Filter Circumvention
- UW reverse traceroute
- etc, etc…

Can’t anticipate next attack employing IP spoofing
The Operational Side

• Arbor 2008 Infrastructure Survey:
  – “Reflective amplification attacks responsible for the largest attacks (40Gbps) exploit IP spoofing”
  – “No bots were used in this attack. The attacker had a small number of compromised Linux boxes from which he’d launch the spoofed source DNS query”

• What’s an operator to do?
Operational View

- IETF BCP38 best filtering practice
- But, not all sources created equal:

<table>
<thead>
<tr>
<th>Example Source IP</th>
<th>Description</th>
<th>Possible Defense</th>
</tr>
</thead>
<tbody>
<tr>
<td>192.168.1.1</td>
<td>RFC1918 private</td>
<td>Static ACL</td>
</tr>
<tr>
<td>1.2.3.4</td>
<td>Unallocated</td>
<td>Bogon Filters</td>
</tr>
<tr>
<td>6.1.2.3</td>
<td>Valid (In BGP table)</td>
<td>uRPF (loose/strict)</td>
</tr>
<tr>
<td>Client IP ⊕ (2^N)</td>
<td>Neighbor Spoof</td>
<td>Switch, DOCSIS</td>
</tr>
</tbody>
</table>
Operational View

• We have defenses, what’s the problem?
• BCP38 suffers from:
  – Lack of hardware support (see NANOG)
  – Global participation requirement
  – Management nightmare (edge filters)
  – Multi-homing, asymmetry, etc implies loose uRPF, implies little protection

• This work: understand the real-world efficacy of these best practices
Spoofer Project

• Background
• Recent Relevance
• Project Methodology
• Results
• Parting Thoughts
Spoofer Test

- Willing participants run “spoofer” client to test policy, perform inference, etc.
  - Binaries, source publicly available
  - Useful diagnostic tool for many
  - Runs once, not an agent

- Clients self-selecting
  - Understand population and potential bias
Spoofer Test

• Testing vulnerability of Internet to source spoofing, not prevalence of source spoofing (e.g. backscatter analysis)

• Uses CAIDA’s Ark infrastructure to test many paths

• Aggregate results, tomography, etc to form global picture of best-practices (BCP38) efficacy
Archipelagio

- Tied into CAIDA’s distributed measurement infrastructure (Ark)
- ~40 nodes, globally distributed
- Ark nodes act as IPv4/v6 spoof probe receivers
Spoofer Operation

- Client confers with control server, receives test
- \((SRC, DST, HMAC, SEQ)\) probe tuples
- Use TCP destination port 80 to avoid secondary filtering
Distributed Probing

- Client sends HMAC keyed spoof probes to ark nodes
- Includes ground-truth validation (non-spoofed) probes
- UDP port 53 + random delay to avoid secondary filtering
- Client runs traceroute to each ark node in parallel
Distributed Probing

• Ark nodes publish to tuple space
• Server asynchronously picks up results
• Run tracefilter (described next)

Return results to user via web report
Outcome of a Probe

• Blocked by OS:
  – Detect, revert to raw Ethernet

• Hits a NAT along path:
  – Detect, exclude from results

• Other blocking (proxy, congestion):
  – Detect, exclude from results

• Blocked by source validation filter

• Successfully received at Ark node
Ark Enables Better Inferences
Multiple Destinations

- Blue line is bogon traffic, Red Valid, Green private
- Greater inference power
- Detect bogon filtering at multiple ASes
- Single server finds valid filtered; too coarse!

Client

1909

195

2153

2152

R&E

22388

101

668

19401

6461

.mil

7660

11537

6509

3

20965

8218

MIT

24324

681

5408

2603

1764

9821

15496

Univ NZ

Univ GR

Commercial

Univ FI
Multiple Destinations

- Metric of spoofability a path rather than a client
- Allows inference on the complete AS graph
- Better understanding of where to employ spoofing defenses
tracefilter

• A tool for *locating* source address validation (anti-spoofing) filters along path
• “traceroute for BCP38”
• Better understand at finer granularity (router) who is/is not filtering
tracefilter

• Client c works in conjunction with our server S
- c sends spoofed packet with:
  - ttl=x, src=S, dst=S+1 for 0<x<pathlen
tracefilter

- S receives ICMP expiration messages from routers along path
- For each decoded TTL, S records which spoofed packets are received
tracefilter

- Increase TTL, repeat
- Largest TTL indicates filtering point
tracefilter

- How can S determine *originating* TTL of c’s packets?
- ICMP echo includes only 28 bytes of expired packet
- c encodes TTL by padding payload with zeros

<table>
<thead>
<tr>
<th>Probe</th>
<th>IP</th>
<th>UDP</th>
<th>Payload</th>
</tr>
</thead>
<tbody>
<tr>
<td>SRC: S</td>
<td>DST: S+1</td>
<td>TTL: x</td>
<td>SRC: SessID</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ICMP</th>
<th>IP</th>
<th>UDP Echo</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type: TTL Exceeded</td>
<td>SRC: S</td>
<td>DST: S+1</td>
</tr>
</tbody>
</table>
Spoofer Project

• Background
• Recent Relevance
• Project Methodology
• Results
• Parting Thoughts
Client Population

Advertised to NANOG, dshield, etc. mailing lists

Slashdot!
Sample Bias

- Obtain general population using 20.8M unique IPs from random topology traces
- Use NetAcuity for geolocation, descriptive statistics
- Aggregate general population into /24s to eliminate non-homogenous properties
Comparing Populations

• Evaluate Bias:
  – Country, speed, organization type, etc.

• Continent Analysis

<table>
<thead>
<tr>
<th>Continent</th>
<th>Population</th>
<th>Measurement Set</th>
</tr>
</thead>
<tbody>
<tr>
<td>N. America</td>
<td>37%</td>
<td>36%</td>
</tr>
<tr>
<td>Europe</td>
<td>29%</td>
<td>33%</td>
</tr>
<tr>
<td>Asia</td>
<td>28%</td>
<td>17%</td>
</tr>
<tr>
<td>S. America</td>
<td>4%</td>
<td>4%</td>
</tr>
<tr>
<td>Oceania</td>
<td>1%</td>
<td>2%</td>
</tr>
<tr>
<td>Africa</td>
<td>0.5%</td>
<td>6%</td>
</tr>
</tbody>
</table>
Client Population Distribution

- ~12,000 unique tests
- 132 countries present in data set
- Don’t claim zero bias
- Do claim diverse and representative
Questions

• Are there filtering variations among paths?
• What filtering methods are used?
• Where in network is source validation?
• Granularity of filtering?
• How vulnerable is the Internet?
• How has filtering evolved over >4 years?
Path-based Filtering Variation?
Valid source probes reach either none (~67%) or all receivers: edge filtering

Surprising variation among bogon and private sources: filtering deeper in-network
Where is source validation?

• tracefilter results:

- 70% of filters at 1st hop; 81% within first two hops
tracefilter Results

- 70% of filters at 1\textsuperscript{st} hop; 81% within first two hops
- 97% of filters within first AS
tracefilter Results

- 70% of filters at 1st hop; 81% within first two hops
- 97% of filters within first AS

If a spoofed packet passes through first two hops, likely to travel unimpeded to destination
Filtering Granularity

- Clients test own IP $\oplus (2^n)$ for $0 < n < 24$
- Filtering on a /8 boundary enables a client within that network to spoof $\sim 16M$ addresses

~70% of clients unable to spoof test sources can spoof neighbors

* “Neighbor spoof” excluded from macro results
AS Degree

- Small or large providers filtering?
- Surprisingly, no clear trend
- Work required across the board (or a new solution)
Evolution of Spoofability

• Find two three-month periods with large and comparable sample sizes

<table>
<thead>
<tr>
<th>Metric</th>
<th>2005 (single dest)</th>
<th>2009 (single dest)</th>
<th>2009 (all dests)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sessions</td>
<td>18.8%</td>
<td>29.9%</td>
<td>31.2%</td>
</tr>
<tr>
<td>Netblocks</td>
<td>20.0%</td>
<td>30.2%</td>
<td>31.7%</td>
</tr>
<tr>
<td>Addresses</td>
<td>5.0%</td>
<td>11.0%</td>
<td>11.1%</td>
</tr>
<tr>
<td>ASes</td>
<td>23.4%</td>
<td>31.8%</td>
<td>34.1%</td>
</tr>
</tbody>
</table>
Evolution of Spoofability

• Find two three-month periods with large and comparable sample sizes

<table>
<thead>
<tr>
<th>Metric</th>
<th>2005 (single dest)</th>
<th>2009 (single dest)</th>
<th>2009 (all dests)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sessions</td>
<td>18.8%</td>
<td>29.9%</td>
<td>31.2%</td>
</tr>
<tr>
<td>Netblocks</td>
<td>20.0%</td>
<td>30.2%</td>
<td>31.7%</td>
</tr>
<tr>
<td>Addresses</td>
<td>5.0%</td>
<td>11.0%</td>
<td>11.1%</td>
</tr>
<tr>
<td>ASes</td>
<td>23.4%</td>
<td>31.8%</td>
<td>34.1%</td>
</tr>
</tbody>
</table>

Less filtering four years later

Change not attributable to increasing number of destinations
Spoofer Project

• Background
• Recent Relevance
• Project Methodology
• Results
• Parting Thoughts
Parting Thoughts

• Even after all these years, source spoofing problem not solved. It’s the incentives:
  – Provider can follow BCP38 and still receive anonymous, spoofed traffic
  – Others can spoof a provider’s address space
  – Disincentive in form of accidental blocking
• Single unfiltered ingress can compromise entire Internet system
  – Can we plug every hole?
  – Regulatory Response? … but multinational?
  – Spoofer page for public provider flogging?
Parting Thoughts

• Tracefilter exposes operational tension between filtering incentives and managing edge filters
• If a spoofed packet isn’t filtered at edge, will travel unimpeded to destination
• Needed?
  – Filtering in the core
  – Clean slate design
• Think (seriously) about alternate techniques?
  – StackPI [Yaar, Perrig, Song 2006]
  – Passport [Liu, Li, Yang, Wetherall 2008]
  – Others?
Parting Thoughts

- Tracefilter exposes operational tension between filtering incentives and managing edge filters.
- If a spoofed packet isn’t filtered at edge, will travel unimpeded to destination.
- Needed?
  - Filtering in the core
  - Clean slate design
- Think (seriously) about alternate techniques?
  - StackPI [Yaar, Perrig, Song 2006]
  - Passport [Liu, Li, Yang, Wetherall 2008]
  - Others?

Thanks!