Recent Advances in IPv6 Security

Fernando Gont

Hackito Ergo Sum 2012
Paris, France. April 12-14, 2012
About...

- Security researcher and consultant for SI6 Networks
- Have worked on security assessment on communications protocols for:
  - UK NISCC (National Infrastructure Security Co-ordination Centre)
  - UK CPNI (Centre for the Protection of National Infrastructure)
- Active participant at the IETF (Internet Engineering Task Force)
Agenda

• Disclaimer
• Motivation for this presentation
• Recent Advances in IPv6 Security
  • IPv6 Addressing
  • IPv6 Fragmentation & Reassembly
  • IPv6 First Hop Security
  • IPv6 Firewalling
  • Mitigation to some Denial of Service attacks
• Conclusions
• Questions and Answers
Disclaimer

- This talks assumes:
  - You know the basics of IPv4 security
  - You now the basics about IPv6 security
  - (i.e. I'm not doing an “IPv6 primer” in this presentation, sorry)
- Much of this is “work in progress” → your input is welcome!
Motivation for this presentation
Motivation for this presentation

- Sooner or later you will need to deploy IPv6
  - In fact, you have (at least) partially deployed it, already
- IPv6 represents a number of challenges: What can we do about them?

Option #1

Option #2

Option #3

Suicide is always an option
Motivation for this presentation (II)

- We have been doing a fair share of IPv6 security research
  - Identification of problems
  - Proposals to mitigate those problems
- Part of our research has been taken to the IETF
- This talk is about our ongoing work to improve IPv6 security
Advances in IPv6 Addressing
IPv6 Global Addresses format

<table>
<thead>
<tr>
<th>n bits</th>
<th>m bits</th>
<th>128-n-m bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>Global Routing Prefix</td>
<td>Subnet ID</td>
<td>Interface ID</td>
</tr>
</tbody>
</table>

• Traditional auto-configuration (SLAAC) addresses embed the MAC address in the Interface ID
• Originally considered convenient for auto-configuration
• But turned out to be a bad idea
Problem #1: Host-scanning attacks

- Search space for host-scanning considered to be $2^{64}$ bits and IPv6 host-scanning deemed infeasible – **really**?
- Modified EUI-64 format identifiers are created as:

<table>
<thead>
<tr>
<th>24 bits</th>
<th>16 bits</th>
<th>24 bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>IEEE OUI</td>
<td>FF FE</td>
<td>Lower 24 bits of MAC</td>
</tr>
</tbody>
</table>

  - Guessable or known
  - Known
  - Unknown

- In practice, the search space is ~$2^{24}$ bits – **feasible**!
Problem #2: Host-tracking attacks

• Modified EUI-64 IIDs are constant for each interface
• As the host moves, the prefix changes, but the IID doesn't
  • the 64-bit IID results in a super-cookie!
• This introduces a problem not present in IPv4: host-tracking

• Example:
  • In net #1, host configures address: 2001:db8:1::1111:2222:3333:4444
  • In net #2, host configures address: 2001:db8:2::1111:2222:3333:4444
  • The IID “1111:2222:3333:4444” leaks out host “identity”.
“Mitigation” to host-tracking

- RFC 4941: privacy/temporary addresses
  - Random IID that change over time
  - Generated in addition to traditional SLAAC addresses
  - Traditional addresses used for server-like communications, temporary addresses for client-like communications

- Operational problems:
  - Difficult to manage!

- Security problems:
  - They mitigate host-tracking only partially
  - They do not mitigate host-scanning attacks
Industry mitigations for scanning attacks

- Microsoft replaced the MAC-address-based identifiers with (non-standard) randomized IIDs
  - Essentially RFC 4941, but they don't vary over time
- Certainly better than MAC-address-based IIDs, but still not “good enough”
- They mitigate host-scanning, but not host tracking – constant IIDs are still present!
Auto-configuration address types

<table>
<thead>
<tr>
<th>Predictable</th>
<th>Stable</th>
<th>Temporary</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Mod. EUI-64 IIDs</td>
<td>None</td>
</tr>
<tr>
<td>Unpredictable</td>
<td>NONE</td>
<td>RFC 4941</td>
</tr>
</tbody>
</table>

- We lack stable privacy-enhanced IPv6 addresses
  - Used to replace MAC-derived addresses
  - Pretty much orthogonal to privacy addresses
  - Probably “good enough” in most cases even without RFC 4941
Stable privacy-enhanced addresses

- draft-gont-6man-stable-privacy-addresses proposes to generate Interface IDs as:
  \[ F(Prefix, Modified_{EUI64}, Network_{ID}, secret_{key}) \]

  Where:
  - \( F() \) is a PRF (e.g., a hash function)
  - Network_{ID} could be e.g. the SSID of a wireless network
  - the rest should be obvious ;-) 

  This function results in addresses that:
  - Are stable within the same subnet
  - Have different Interface-IDs when moving across networks
  - For the most part, they have “the best of both worlds”
Work in progress

- Proposal presented at IETF 83 (Paris, March 2012)
- 6man wg currently being polled about adoption of this document
- Hopefully, host-scanning attacks will become unfeasible, and host tracking less trivial ;-)
IPv6 Fragmentation and Reassembly
IPv6 fragmentation

- IPv6 fragmentation performed only by hosts (never by routers)
- Fragmentation support implemented in “Fragmentation Header”
- Fragmentation Header syntax:

<table>
<thead>
<tr>
<th>8 bits</th>
<th>8 bits</th>
<th>13 bits</th>
<th>2b</th>
<th>1b</th>
</tr>
</thead>
<tbody>
<tr>
<td>Next Header</td>
<td>Reserved</td>
<td>Fragment Offset</td>
<td>Res</td>
<td>M</td>
</tr>
</tbody>
</table>

Identification
Fragment Identification

- Security Implications of predictable Fragment IDs well-known from the IPv4 world
  - idle-scanning, DoS attacks, etc.
- Situation exacerbated by larger payloads resulting from:
  - Larger addresses
  - DNSSEC
- But no worries, since we learned the lesson from the IPv4 world... – right?
### Fragment ID generation policies

<table>
<thead>
<tr>
<th>Operating System</th>
<th>Algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>FreeBSD 9.0</td>
<td>Randomized</td>
</tr>
<tr>
<td>NetBSD 5.1</td>
<td>Randomized</td>
</tr>
<tr>
<td>OpenBSD-current</td>
<td>Randomized (based on SKIPJACK)</td>
</tr>
<tr>
<td>Linux 3.0.0-15</td>
<td>Predictable (GC init. to 0, incr. by +1)</td>
</tr>
<tr>
<td>Linux-current</td>
<td>Unpredictable (PDC init. to random value)</td>
</tr>
<tr>
<td>Solaris 10</td>
<td>Predictable (PDC, init. to 0)</td>
</tr>
<tr>
<td>Windows 7 Home Prem.</td>
<td>Predictable (GC, init. to 0, incr. by +2)</td>
</tr>
</tbody>
</table>

GC: Global Counter  
PDC: Per-Destination Counter

At least Solaris and Linux patched in response to our IETF I-D – more patches expected!
IPv6 Fragment Reassembly

- Security implications of overlapping fragments well-known (think Ptacek & Newsham, etc.,)
- Nonsensical for IPv6, but originally allowed in the specs
- Different implementations allow them, with different results
- RFC 5722 updated the specs, forbidding overlapping fragments
- Most current implementations reflect the updated standard
- See http://blog.si6networks.com
IPv6 Fragment reassembly (II)

- ICMPv6 PTB < 1280 triggers inclusion of a FH in all packets to that destination (not actual fragmentation)
- Result: IPv6 atomic fragments (Frag. Offset=0, More Frag.=0)
- Some implementations mixed these packets with “normal” fragmented traffic
- draft-ietf-6man-ipv6-atomic-fragments fixes that:
  - IPv6 atomic fragments required to be processed as non-fragmented traffic
  - Document ready for WGLC
## Handling of IPv6 atomic fragments

<table>
<thead>
<tr>
<th>Operating System</th>
<th>Atomic Frag. Support</th>
<th>Improved processing</th>
</tr>
</thead>
<tbody>
<tr>
<td>FreeBSD 8.0</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>FreeBSD 8.2</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>FreeBSD 9.0</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Linux 3.0.0-15</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>NetBSD 5.1</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>OpenBSD-current</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Solaris 11</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Windows Vista (build 6000)</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Windows 7 Home Premium</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>

At least OpenBSD patched in response to our IETF I-D – more patches expected!
IPv6 First Hop Security
IPv6 First Hop Security

- Security mechanisms/policies employed/enforced at the first hop (local network)
- Fundamental problem: lack of feature-parity with IPv4
  - arpwatch-like Neighbor Discovery monitoring virtually impossible
  - DHCP-snooping-like RA blocking trivial to circumvent
IPv6 First-Hop Security (II)

- Fundamental problem: complexity of traffic to be “processed at layer-2”
- Example:
Bringing “sanity” to ND traffic

- draft-gont-6man-nd-extension-headers forbids use of fragmentation with Neighbor Discovery
  - It makes ND monitoring feasible
  - Turns out it is vital for SEND (or SEND could be DoS'ed with fragments)

- Work in progress:
  - Discussed last year
  - Presented at IETF 83 (Paris, March 2012)
  - 6man wg to be polled about adoption shortly
RA-Guard

- Meant to block RA packets on “unauthorized” switch ports
- Real implementations trivial to circumvent
- draft-gont-6man-ra-guard-implementation contains:
  - Discussion of RA-Guard evasion techniques
  - Advice to filter RAs, while avoiding false positives
- Can only be evaded with overlapping fragments
  - But most current OSes forbid them
  - And anyway there's nothing we can do about this :-)
- Work in progress: to be WGLC'ed soon.
IPv6 firewalling
First step away from “insanity”

• Specs-wise, state-less IPv6 packet filtering is impossible
• draft-gont-6man-oversized-header-chain tries to improve that:
  • The entire IPv6 header chain must be within the first PMTU bytes of the packet
  • i.e. packets with header chains that span more than one fragment may be blocked – don't send them!
• Work in progress:
  • Presented at IETF 83 (Paris, March 2012)
  • To be discussed on the 6man wg mailing-list
• There's an insanely large amount of work to be done in the area of IPv6 firewalling
Mitigation to some DoS attacks
IPv6 Smurf-like Attacks

- IPv6 is assumed to eliminate Smurf-like attacks
  - Hosts are assumed to not respond to global multicast addresses
- But,
  - Options of type 10xxxxxx require hosts to generate ICMPv6 errors
  - Even if the packet was destined to a multicast address
- Probably less important than the IPv4 case (since it requires multicast routing)
- But might be an issue if multicast routing is deployed
- draft-gont-6man-ipv6-smurf-amplifier addresses this issue:
  - Discusses the problem
  - Recommends that multicasted packets must not elicit ICMPv6 errors
Some conclusions
Some conclusions

- Many IPv4 vulnerabilities have been re-implemented in IPv6
  - We just didn't learn the lesson from IPv4, or,
  - Different people working in IPv6 than working in IPv4, or,
  - The specs could make implementation more straightforward, or,
  - All of the above? :-)

- Still lots of work to be done in IPv6 security
  - We all know that there is room for improvements
  - We need IPv6, and should work to improve it
Questions?
Thanks!

Fernando Gont
fgont@si6networks.com

IPv6 Hackers mailing-list
http://www.si6networks.com/community/

www.si6networks.com