SEARCHING THE VOID
Nothing said during this talk is endorsed, supported, or is the opinion of any past, current, or future employer, family member, or collaborator.
IPv6 Info

• 128 Bits for each address (v4 is 32)
• Represented with hex
  • 0123456789abcdef
• Address Truncation
  • 2001:0db8:0000:0000:0000:0000:0000:0001
IPv6 Info

• 128 Bits for each address (v4 is 32)
• Represented with hex
  • 0123456789abcdef
• Address Truncation
  • 2001:db8::1
VASTNESS
The Problem
$2^{128} \approx 3.4 \times 10^{38}$

340,282,366,920,938,463,463,374,607,431,768,211,456
Enough addresses for each atom in every person on earth
Bad Assumptions

• ‘While IPv6 hasn't exactly taken the world by storm, it gets significant use in some (usually Asian) countries’
OK but I don’t have to scan it all

• Most companies will have a /64
  • 18,446,744,073,709,551,616 Addresses
  • 58,000 years/1 server || 1 year/58,000 servers
  • $2,543,288,400 AWS - i3.16xlarge

• Some might have a /48
  • 1,208,925,819,614,629,174,706,176 Addresses
  • 3,833,478,626 years
Find Humility
Embrace the Impossible
Let's do some recon
SHINK THE SCAN SPACE
The Solution
Network Reconnaissance in IPv6 Networks

Abstract

IPv6 offers a much larger address space than that of its IPv4 counterpart. An IPv4 address space (32 bits) can accommodate approximately $1.844 \times 10^{38}$ hosts, thus resulting in a much lower host density ($\text{#hosts}/\text{#addresses}$) than is typical in IPv4 networks, where a site typically has 65,000 or fewer unique addresses. As a result, it is widely assumed that it would take a tremendous effort to perform address-scanning attacks against IPv6 networks; therefore, IPv6 address-scanning attacks have been considered infeasible. This document formally obsoletes RFC 5157, which first discussed this assumption, by providing further analysis on how traditional address-scanning techniques apply to IPv6 networks and exploring some additional techniques that can be employed for IPv6 network reconnaissance.
PATTERNS
Slaac Hacks and SYN/ACKS

- EUI-64 format the host bits
  - 00:00:0C:FA:DE:D1 -> 2001:DB8::200:CFF:FEFA:DED1
- If you know the OUI, Scan space is roughly $2^{24}$
- Autoconfiguration of server addresses is unlikely
- Systems are moving towards non-EUI64 SLAAC

bash-3.2$ wc -l uniq_all_addr.txt
  146014 uniq_all_addr.txt
bash-3.2$ grep -i "ff:fe" uniq_all_addr.txt | wc -l
  15447
Low Hanging Bytes

• Using low numeric host bits
• Most common on infrastructure
• Example: 2001:db8::

Traceroute is awesome again

```
apollo@Zeus ~ traceroute6 google.com
traceroute to google.com (2607:f8b0:4004:802::200e) from
  30 hops max, 24 byte packets
     75 ms  0.276 ms  0.469 ms
  2  2604:a880:ffff:0:2::2d (2604:a880:ffff:0:2::2d)  0.49 ms
  3  2604:a880:ffff:c::41 (2604:a880:ffff:c::41)  0.75 ms
  4  2604:a880::101 (2604:a880::101)  0.917 ms  0.913 ms
  5  2001:4860:0:1127::14 (2001:4860:0:1127::14)  1.91 ms
  6  2001:4860::8:4000:cd7f (2001:4860::8:4000:cd7f)  2.24 ms
  8  2001:4860::1:4000:c8a3 (2001:4860::1:4000:c8a3)  8.04 ms
  9  2001:4860::0:1099::1 (2001:4860::0:1099::1)  7.606 ms
 10  2001:4860::0:1::20d5 (2001:4860::0:1::20d5)  7.786 ms
 11  iad23s58-in-x0e.1e100.net (2607:f8b0:4004:802::200e)
```
<table>
<thead>
<tr>
<th>Address</th>
<th>Comment</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>2600:1406:27::1</td>
<td>Cisco IOS Software, IOS-XE Software, Catalyst 4500 L3 Switch Software (cat4500e-UNIVERSALK9-M), Version 12.2(42)</td>
<td>United States</td>
</tr>
<tr>
<td>2001:d18:1</td>
<td>SSH-2.0-Cisco-1.25</td>
<td>United States</td>
</tr>
<tr>
<td>2a05:91c0:501:1</td>
<td>Cisco IOS Software, Catalyst 4500 L3 Switch Software (cat4500e-ENTSERVICESK9-M), Version 12.2(42)</td>
<td>China, Shenyang</td>
</tr>
<tr>
<td>2600:1404:28::1</td>
<td>Cisco IOS Software, IOS-XE Software, Catalyst 4500 L3 Switch Software (cat4500e-UNIVERSALK9-M), Version 12.2(42)</td>
<td>United States</td>
</tr>
<tr>
<td>2401:f480:1</td>
<td>SSL Certificate</td>
<td>China</td>
</tr>
</tbody>
</table>

**Network Data**

- **2600:1406:27::1**
  - Comment: Cisco IOS Software, IOS-XE Software, Catalyst 4500 L3 Switch Software (cat4500e-UNIVERSALK9-M), Version 12.2(42)
  - Details: United States

- **2001:d18:1**
  - Comment: SSH-2.0-Cisco-1.25
  - Key type: ssh-rsa
  - Key: A98D3B323C728ABAAABAB93608AE3K28N86PEI9GIN6Y5J7VY6C84132326VR8D9GB46980A
  - Details: United States

- **2a05:91c0:501:1**
  - Comment: Cisco IOS Software, Catalyst 4500 L3 Switch Software (cat4500e-ENTSERVICESK9-M), Version 12.2(42)
  - Details: China, Shenyang

- **2600:1404:28::1**
  - Comment: Cisco IOS Software, IOS-XE Software, Catalyst 4500 L3 Switch Software (cat4500e-UNIVERSALK9-M), Version 12.2(42)
  - Details: United States

- **2401:f480:1**
  - Comment: SSL Certificate
  - Details: China

**Network Event**

- **2401:f480:1**
  - Comment: SSL Certificate
  - Details: China

**SSL Certificate**

- **2401:f480:1**
  - Issued By: [Issuer Name]
  - Date: Mon, 06 Oct 2017 14:11:48 GMT

**HTTP/1.1 401 Unauthorized**

- **2401:f480:1**
  - Comment: HTTP/1.1 401 Unauthorized
  - Details: SSL Certificate

**Technical Support**

- **2600:1406:27::1**

- **2001:d18:1**

- **2a05:91c0:501:1**

- **2600:1404:28::1**

**Copyright**

- **2600:1406:27::1**
  - Comment: Copyright (c) 1986-2016 by Cisco Systems, Inc.

- **2001:d18:1**
  - Comment: Copyright (c) 1986-2016 by Cisco Systems, Inc.

- **2a05:91c0:501:1**
  - Comment: Copyright (c) 1986-2016 by Cisco Systems, Inc.

- **2600:1404:28::1**
  - Comment: Copyright (c) 1986-2016 by Cisco Systems, Inc.
Seeking the Horror of Bovine Ghosts
Or, Scanning for ::DEAD:BEEF

- Essentially IPv6 dictionary Attack
- XXXX:XXXX:f113:8083:face:b00c:0:25de
- Might be hex OR subs
  - 2001:db8::face:b00c:0:25de
  - 2001:db8::a11:beef:7ac0
bash-3.2$ grep -i "beef" uniq_all_addr.txt | wc -l
117
bash-3.2$ grep -i "dead" uniq_all_addr.txt | wc -l
123
bash-3.2$ grep -i "face" uniq_all_addr.txt | wc -l
24
bash-3.2$ grep -i "7ac0" uniq_all_addr.txt | wc -l
3
IPv4 In my Host Bits?

- Keeps things simple
- I’ve seen this advice given in a few forums
- Examples
  - 2001:db8::192.0.2.1 -> 2001:db8::c000:201
  - 2001:db8::192:0:2:1
- Might also have port as the LSBs
Taking the easy Route

- Make sure your prefix is on the internet
- Useful for IPv4 as well
- Public Route Servers are easy to find
Routing entry for 2607:f8b0:4005::/48
- Known via "bgp 3549", distance 200, metric 100, type internal
  Route count is 1/1, share count 0
Routing paths:
  2001:450:2001:8018::1
  MPLS label: nolabel
  Last updated 7w0d ago

Routing entry for ::/0
- Known via "static", distance 1, metric 0
  Redistributing via bgp 3549
  Route count is 1/1, share count 0
Routing paths:
  2001:450:2001:8018::1
  Last updated 7w0d ago
DNS dumpster Diving

• DNS info is amazing for OSINT!
• Plenty of good resources for free
• Look for common subdomains to get a better picture of the network
Checking DNS records for: google.com

google.com A record: 216.58.195.78
google.com AAAA record: 2607:f8b0:4005:807::200e
www.google.com A record: 172.217.11.68
www.google.com AAAA record: 2607:f8b0:4005:802::2004
mail.google.com A record: 216.58.195.69
mail.google.com AAAA record: 2607:f8b0:4005:807::2005
remote.google.com A record:
blog.google.com A record: 216.58.195.73
blog.google.com AAAA record: 2607:f8b0:4005:807::2009
webmail.google.com A record:
service.google.com A record:
ns1.google.com A record: 216.239.32.10
ns1.google.com AAAA record:
n2.google.com A record: 216.239.34.10
ns2.google.com AAAA record:
smt1p.google.com A record:
secure.google.com A record:
vpn.google.com A record: 64.9.224.70
vpn.google.com A record: 64.9.224.68
vpn.google.com A record: 64.9.224.69
vpn.google.com AAAA record:
m.google.com A record: 216.58.195.75
m.google.com AAAA record: 2607:f8b0:4005:807::200b
ftp.google.com A record:
mail.google.com A record: 216.58.195.69
mail.google.com AAAA record: 2607:f8b0:4005:807::2005
CENSYng CERTificates

• Certificates often expose uncommon/rare subdomains
• Data leaks can come from here as well
• Censys, Cert.sh, Shodan are good places to get started

In [109]: !dig -t aaaa login.corp.... +short
       uberproxy.l....com.
   f8b0:400e:c04::81
RFC1035 causes this

- `ip6.arpa` is a big radix tree
- IPv6 PTRs are backwards and exploded
  - 2001:db8::1
  - 1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.8.b.d.0.1.0.0.2.ip6.arpa

- Hint: Python Does this for you!

```
ipaddress.IPv6Address("2001:db8::1").reverse_pointer
'1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.8.b.d.0.1.0.0.2.ip6.arpa'
```
Continued

2001:db8:4006:819::/64
2001:db8:4006:819::1XXX < Reply Code 3
2001:db8:4006:819::2XXX < Reply Code 0
2001:db8:4006:819::3XXX < Reply Code 3
[...]
2001:db8:4006:819::FXXX < Reply Code 3
Et TU, NMAP?

• Does not support range scanning
  • “IPv6 addresses can only be specified by their fully qualified IPv6 address or hostname. CIDR and octet ranges aren't supported for IPv6 because they are rarely useful.” - https://nmap.org/book/nping-man-target-specification.html

• Many features are considered experimental

• NSE to the rescue
  • https://svn.nmap.org/nmap/scripts/targets-ipv6-wordlist.nse
  • Some internal goodness as well
Choosy Hackers choose THC-IPv6

- Comes with Kali
- Great once you are in, but not our scope
- dnsdict6 - DeadBeef, etc
- trace6 - Fast IPv6 trace route
IPv6toolkit

• Created by SI Networks (Fernando Gont)
  • Fernando is the co-author of RFC7707
• scan6 - A scanner!!!
  • SLAAC, Low Byte, IPv4, et al
• path6 - Another Traceroute tool
Questions?