How to Securely Operate an IPv6 Network

Eric Vyncke, evyncke@cisco.com

@evyncke

OPSEC
Internet-Draft
Intended status: Informational
Expires: April 25, 2014

K. Chittimaneni
Google
M. Kaeo
Double Shot Security
E. Vyncke
Cisco Systems
October 22, 2013

Operational Security Considerations for IPv6 Networks
draft-ietf-opsec-v6-04
Foreword

- All topics common to IPv4/IPv6 are unchanged:
  - Physical security
  - Role Base Access Control
  - ....

- I took the liberty to include Cisco configuration (as it may be useful for you) but I will not detail them
Agenda

- Management Plane
- Control Plane
  - Routing Information
  - Neighbor Discovery
  - Control Plane Protection
- Data Plane
  - Anti-spoofing
  - Access Control List
  - Tunnel loops
- Telemetry
- Forensic
- Summary
Management Plane
Management over IPv6

- SSH, syslog, SNMP, NetFlow, RADIUS all work over IPv6
- Dual-stack management plane
  - More resilient: works even if one IP version is down
  - More exposed: can be attacked over IPv4 and IPv6
- As usual, infrastructure ACL is your friend (more to come) as well as out-of-band management
- So, protect all SNMP, SSH access from untrusted interfaces
Control Plane: Routing Protocols
Preventing IPv6 Routing Attacks
Protocol Authentication

- BGP, IS-IS, EIGRP no change:
  - An MD5 authentication of the routing update

- OSPFv3 originally has changed and pulled MD5 authentication from the protocol and instead rely on transport mode IPsec (for authentication and confidentiality)
  - But see RFC 6506 *(not yet widely implemented)*

- IPv6 routing attack best practices
  - Use traditional authentication mechanisms on BGP and IS-IS
  - **Use IPsec** to secure protocols such as OSPFv3
BGP Route Filters

- Pretty obvious for customer links
- For peering, a relaxed one

```
ipv6 prefix-list RELAX deny 3ffe::/16 le 128
ipv6 prefix-list RELAX deny 2001:db8::/32 le 128
ipv6 prefix-list RELAX permit 2001::/32
ipv6 prefix-list RELAX deny 2001::/32 le 128
ipv6 prefix-list RELAX permit 2002::/16
ipv6 prefix-list RELAX deny 2002::/16 le 128
ipv6 prefix-list RELAX deny 0000::/8 le 128
ipv6 prefix-list RELAX deny fe00::/9 le 128
ipv6 prefix-list RELAX deny ff00::/8 le 128
ipv6 prefix-list RELAX permit 2000::/3 le 48
ipv6 prefix-list RELAX deny 0::/0 le 128
```

Source: http://www.space.net/~gert/RIPE/ipv6-filters.html
Link-Local Addresses vs. Global Addresses

- Link-Local addresses, fe80::/10, (LLA) are isolated
  - Cannot reach outside of the link
  - **Cannot be reached from outside of the link** 😊
  - LLA can be configured statically (not the EUI-64 default) to avoid changing neighbor statements when changing MAC

```bash
interface FastEthernet 0/0
ipv6 address fe80::1/64 link-local
```

OPsec Working Group
Internet-Draft
Intended status: Informational
Expires: July 10, 2014

M. Behringer
E. Vyncke
Cisco
January 6, 2014

Using Only Link-Local Addressing Inside an IPv6 Network
draft-ietf-opsec-lla-only-06
LLA-Only Pros and Cons

Benefits:
- no remote attack against your infrastructure links: implicit infrastructure ACL*
- Smaller routing table (links do not appear)
- Simpler configuration
- Easier to renumber

Cons:
- need to provision loopback for:
  - ICMP for Traceroute
  - ICMP for PMTUD
  - SNMP/NetFlow/syslog/ ...  
- No interface ping

Special case for IXP:
- Usually a specific /64 which is not routed => uRPF will drop ICMP generated (PMTUd) by routers in the IXP
- LLA-only on the IXP interfaces => ICMP are generated from a non IXP interface

*: loopbacks are still routable/reachable
Control Plane: Neighbor Discovery
Scanning Made Bad for CPU

Remote Neighbor Cache Exhaustion RFC 6583

- Potential router CPU/memory attacks if aggressive scanning
  - Router will do Neighbor Discovery... And waste CPU and memory
- **Local router** DoS with NS/RS/…
Mitigating Remote Neighbor Cache Exhaustion

- Built-in rate limiter with options to tune it
  - Since 15.1(3)T: `ipv6 nd cache interface-limit`
  - Or IOS-XE 2.6: `ipv6 nd resolution data limit`
  - `Destination-guard` is part of First Hop Security phase 3
  - Priority given to refresh existing entries vs. discovering new ones (RFC 6583)

- Using a /64 on **point-to-point links** => a lot of addresses to scan!
  - Using /127 could help (RFC 6164)

- **Internet edge/presence**: a target of choice
  - Ingress ACL permitting traffic to specific statically configured (virtual) IPv6 addresses only

- Using infrastructure ACL prevents this scanning
  - iACL: edge ACL denying packets addressed to your routers
  - Easy with IPv6 because new addressing scheme can be done 😊

http://www.insinuator.net/2013/03/ipv6-neighbor-cache-exhaustion-attacks-risk-assessment-mitigation-strategies-part-1
Simple Fix for Remote Neighbor Cache Exhaustion

- Ingress ACL allowing only valid destination and dropping the rest
- NDP cache & process are safe
- Requires DHCP or static configuration of hosts
ARP Spoofing is now NDP Spoofing: Threats

- ARP is replaced by Neighbor Discovery Protocol
  - Nothing authenticated
  - Static entries overwritten by dynamic ones
- Stateless Address Autoconfiguration
  - rogue RA (malicious or not)
  - All nodes badly configured
    - DoS
    - Traffic interception (Man In the Middle Attack)
- Attack tools exist (from THC – The Hacker Choice)
  - Parasit6
  - Fakerouter6
  - ...
ARP Spoofing is now NDP Spoofing: Mitigation

- **GOOD NEWS**: dynamic ARP inspection for IPv6 is available
  - First phase (Port ACL & RA Guard) available since Summer 2010
  - Second phase (NDP & DHCP snooping) starting to be available since Summer 2011

- **(Kind of) GOOD NEWS**: Secure Neighbor Discovery
  - SeND = NDP + crypto
  - IOS 12.4(24)T But not in Windows Vista, 2008 and 7, Mac OS/X, iOS, Android

- **Other GOOD NEWS**:
  - Private VLAN works with IPv6
  - Port security works with IPv6
  - IEEE 801.X works with IPv6 (except downloadable ACL)
Mitigating Rogue RA: Host Isolation

- Prevent Node-Node Layer-2 communication by using:
  - Private VLANs (PVLAN) where nodes (isolated port) can only contact the official router (promiscuous port)
  - WLAN in ‘AP Isolation Mode’
  - 1 VLAN per host (SP access network with Broadband Network Gateway)

- Link-local multicast (RA, DHCP request, etc) sent only to the local official router: no harm

- Can break DAD
  - Advertise the SLAAC prefix without the on-link bit to force router to do ‘proxy-ND’
First Hop Security: RAguard since 2010
RFC 6105

- **Port ACL** blocks all ICMPv6 RA from hosts
  ```
  interface FastEthernet0/2
  ipv6 traffic-filter ACCESS_PORT in
  access-group mode prefer port
  ```

- **RA-guard lite** *(12.2(33)SXI4 & 12.2(54)SG)*: also dropping all RA received on this port
  ```
  interface FastEthernet0/2
  ipv6 nd raguard
  access-group mode prefer port
  ```

- **RA-guard** *(12.2(50)SY, 15.0(2)SE)*
  ```
  ipv6 nd raguard policy HOST device-role host
  ipv6 nd raguard policy ROUTER device-role router
  ipv6 nd raguard attach-policy HOST vlan 100
  interface FastEthernet0/0
  ipv6 nd raguard attach-policy ROUTER
  ```

Can also enforce MTU, prefix, ... In RA
Control Plane Protection
Control Plane Policing for IPv6
Protecting the Router CPU

- Against DoS with NDP, Hop-by-Hop, Hop Limit Expiration...
- See also RFC 6192
Data Plane
DoS Example
Ping-Pong over Physical Point-to-Point

- Same as in IPv4, on real P2P without NDP, if not for me, then send it on the other side... Could produce looping traffic
- Classic IOS and IOS-XE platforms implement RFC 4443 so this is not a threat
  - Except on 76xx see CSCtg00387 (tunnels) and few others
  - IOS-XR see CSCsu62728
  - Else use /127 on P2P link (see also RFC 6164)
  - Or use infrastructure ACL or only link-local addresses
IPv6 Bogon and Anti-Spoofing Filtering

- IPv6 nowadays has its bogons:

- Every network should implement two forms of anti-spoofing protections:
  - Prevent spoofed addresses from entering the network
  - Prevent the origination of packets containing spoofed source addresses

- Anti-spoofing in IPv6 same as IPv4
  - => Same technique for single-homed edge= uRPF

Diagram:

IPv6 Intranet
IPv6 Unallocated Source Address

Inter-Networking Device with uRPF Enabled

IPv6 Intranet/Internet

No Route to SrcAddr => Drop
Bogons Filtering

- Detailed & updated list at:
  - http://www.team-cymru.org/Services/Bogons/fullbogons-ipv6.txt
- Or simpler but more relaxed

```
ipv6 access-list NO_BOGONS
  remark Always permit ICMP unreachable (Path MTU Discovery & co)
  permit icmp any any unreachable
  remark Permit only large prefix blocks from IANA
  permit ip 2001::/16 any
  permit ip 2002::/16 any
  permit ip 2003::/18 any
  permit ip 2400::/12 any
  permit ip 2600::/10 any
  permit ip 2800::/12 any
  permit ip 2a00::/12 any
  permit ip 2c00::/12 any
  Remark implicit deny at the end (but see later)
```

Source: http://www.iana.org/assignments/ipv6-unicast-address-assignments/ipv6-unicast-address-assignments.xml
Remote Triggered Black Hole

- RFC 5635 RTBH is easy in IPv6 as in IPv4
- uRPF is also your friend for blackholing a source
- RFC 6666 has a specific discard prefix
  - 100::/64


Source: Wikipedia Commons
Parsing the Extension Header Chain

- Finding the layer 4 information is not trivial in IPv6
  - Skip all known extension header
  - Until either known layer 4 header found => MATCH
  - Or unknown extension header/layer 4 header found... => NO MATCH

IPv6 hdr  HopByHop  Routing  AH  TCP  data
IPv6 hdr  HopByHop  Routing  AH  Unknown L4  ???
IOS IPv6 Extended ACL

- Can match on
  - Upper layers: TCP, UDP, SCTP port numbers, ICMPv6 code and type
  - TCP flags SYN, ACK, FIN, PUSH, URG, RST
  - Traffic class (only six bits/8) = DSCP, Flow label (0-0xFFFFF)

- IPv6 extension header
  - **routing** matches any RH, **routing-type** matches specific RH
  - **mobility** matches any MH, **mobility-type** matches specific MH
  - **dest-option** matches any destination options
  - **auth** matches AH
  - **hbh** matches hop-by-hop (since 15.2(3)T)

- **fragments** keyword matches
  - Non-initial fragments

- **undetermined-transport** keyword does not match if
  - TCP/UDP/SCTP and ports are in the fragment
  - ICMP and type and code are in the fragment
  - Everything else matches (including OSPFv3, …)
  - Only for deny ACE

---

**Check your platform & release as your mileage can vary…**

© 2014 Cisco and/or its affiliates. All rights reserved. Cisco Public
Looping Attack Between 6to4 and ISATAP (RFC 6324)

- **Root cause**
  - Same IPv4 encapsulation (protocol 41)
  - Different ways to embed IPv4 address in the IPv6 address

- **ISATAP router**:
  - accepts 6to4 IPv4 packets
  - Can forward the inside IPv6 packet back to 6to4 relay

- Symmetric looping attack exists

**Mitigation:**
- Easy on ISATAP routers: deny packets whose IPv6 is its 6to4
- Less easy on 6to4 relay: block all ISATAP-like local address?
- Good news: not so many open ISATAP routers on the Internet
- Do not announce the 6to4 relay address outside of your AS and accepts protocol-41 packets only from your AS

---

1. Spoofed packet
   S: 2001:db8::200:5efe:c000:201
   D: 2002:c000:202::1

2. IPv4 Packet to 192.0.2.2 containing
   S: 2001:db8::200:5efe:c000:201
   D: 2002:c000:202::1

3. IPv6 packet
   S: 2001:db8::200:5efe:c000:201
   D: 2002:c000:202::1

*Repeat until Hop Limit == 0*
6rd Relay Security Issues

- 6rd is more constrained than 6to4, hence more secure
- IPv4 ACL (or IPv4 routing) can limit the 6rd packets to the 6rd domain within the ISP
  - No more open relay
  - No more looping attacks
IPv6 security is similar to IPv4 security
No excuse to operate an insecure IPv6 network
Available Tools

- Usually IPv4 telemetry is available
- **SNMP MIB**
  - Not always available yet on Cisco gears
- **Flexible Netflow for IPv6**
  - Available in: 12.4(20)T, 12.2(33)SRE
  - Public domain tools: nfsen, nfdump, nfcpad...
# IPv6 MIB Implementation

<table>
<thead>
<tr>
<th>Protocol Version Independent (PVI)</th>
<th>IP FWD (ROUTES)</th>
<th>IP</th>
<th>ICMP</th>
<th>TCP</th>
<th>UDP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original IPv4 only</td>
<td></td>
<td>2096</td>
<td>2011</td>
<td>2012</td>
<td>2013</td>
</tr>
<tr>
<td>IPv6 only</td>
<td></td>
<td>2465</td>
<td>2466</td>
<td>2452</td>
<td>2454</td>
</tr>
</tbody>
</table>
| IPv4/IPv6 stats can be monitored from CLI “show interface accounting” on most platforms
| RFC 4292 and 4293 – Interface Stats table are added, also required HW support
| Tunnel MIB (RFC 4087)            |                 |     |      |      |      |
| ipMIB                            | rfc2096-update  | 4292 | rfc2011-update=4293 = IP-MIB |
|                                  | rfc2012-update  | 4022 | rfc2013-update=4113           |
### Using SNMP to Read Interfaces Traffic

```
evyncke@charly:~$ snmpwalk -c secret -v 1 udp6:[2001:db8::1] -Cw 70 -m IP-MIB
ipNetToPhysicalPhysAddress
```

**SNMP table: IP-MIB::ipIfStatsTable**

<table>
<thead>
<tr>
<th>index</th>
<th>ipIfStatsInReceives</th>
<th>ipIfStatsHCInReceives</th>
<th>ipIfStatsInOctets</th>
</tr>
</thead>
<tbody>
<tr>
<td>ipv4.1</td>
<td>683929</td>
<td>?</td>
<td>55054803</td>
</tr>
<tr>
<td>ipv4.2</td>
<td>1123281</td>
<td>?</td>
<td>107467461</td>
</tr>
<tr>
<td>ipv6.1</td>
<td>152612</td>
<td>?</td>
<td>17261398</td>
</tr>
<tr>
<td>ipv6.2</td>
<td>15083935</td>
<td>?</td>
<td>2131680450</td>
</tr>
</tbody>
</table>

```
evyncke@charly:~$ snmpwalk -c secret -v 1 udp6:[2001:db8::1] -Cw 70 ifTable
```

**SNMP table: IF-MIB::ifTable**

<table>
<thead>
<tr>
<th>index</th>
<th>ifIndex</th>
<th>ifDescr</th>
<th>ifType</th>
<th>ifMtu</th>
<th>ifSpeed</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>FastEthernet0/0</td>
<td>ethernetCsmacd</td>
<td>1500</td>
<td>1000000000</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>FastEthernet0/1</td>
<td>ethernetCsmacd</td>
<td>1500</td>
<td>1000000000</td>
</tr>
</tbody>
</table>
Using SNMP to Read IPv4/IPv6 Neighbors Cache

```
evyncke@charly:~$ snmpwalk -c secret -v 1 udp6:[2001:db8::1] -m IP-MIB
ipNetToPhysicalPhysAddress
IP-MIB::ipNetToPhysicalPhysAddress.1.ipv4."192.168.0.4" = STRING: 0:80:c8:e0:d4:be
...
IP-MIB::ipNetToPhysicalPhysAddress.2.ipv6."2a:02:05:78:85:00:01:01:02:07:e9:ff:fe:f2:a0:c6" = STRING: 0:7:e9:f2:a0:c6
IP-MIB::ipNetToPhysicalPhysAddress.2.ipv6."2a:02:05:78:85:00:01:01:02:20:4a:ff:fe:bf:ff:5f" = STRING: 0:20:4a:bf:ff:5f
...
```

```
evyncke@charly:~$ snmptable -c secret -v 1 udp6:[2001:db8::1] -Ci -m IP-MIB
ipNetToPhysicalTable
```

© 2014 Cisco and/or its affiliates. All rights reserved.
# Flexible Flow Record: IPv6 Key Fields

<table>
<thead>
<tr>
<th>IPv6</th>
<th>Payload Size</th>
<th>Routing</th>
<th>Transport</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP (Source or Destination)</td>
<td></td>
<td>Destination AS</td>
<td>TCP Flag: ACK</td>
</tr>
<tr>
<td>Prefix (Source or Destination)</td>
<td></td>
<td>Peer AS</td>
<td>TCP Flag: CWR</td>
</tr>
<tr>
<td>Mask (Source or Destination)</td>
<td></td>
<td>Traffic Index</td>
<td>TCP Flag: ECE</td>
</tr>
<tr>
<td>Minimum-Mask</td>
<td></td>
<td>Forwarding Status</td>
<td>TCP Flag: FIN</td>
</tr>
<tr>
<td>Protocol</td>
<td></td>
<td>Is-Multicast</td>
<td>TCP Flag: FIN</td>
</tr>
<tr>
<td>Traffic Class</td>
<td></td>
<td>IGP Next Hop</td>
<td>TCP Flag: PSH</td>
</tr>
<tr>
<td>Flow Label</td>
<td></td>
<td>BGP Next Hop</td>
<td>TCP Flag: RST</td>
</tr>
<tr>
<td>Option Header</td>
<td></td>
<td>Flow</td>
<td>TCP Flag: SYN</td>
</tr>
<tr>
<td>Header Length</td>
<td></td>
<td>Sampler ID</td>
<td>TCP Flag: URG</td>
</tr>
<tr>
<td>Payload Length</td>
<td></td>
<td>Direction</td>
<td>TCP Window-Size</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>UDP Message Length</td>
</tr>
</tbody>
</table>

## Transport

<table>
<thead>
<tr>
<th>Destination Port</th>
<th>Source Port</th>
<th>TCP Flag: ACK</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>TCP Flag: CWR</td>
</tr>
<tr>
<td>ICMP Code</td>
<td></td>
<td>TCP Flag: ECE</td>
</tr>
<tr>
<td>ICMP Type</td>
<td></td>
<td>TCP Flag: FIN</td>
</tr>
<tr>
<td>IGMP Type</td>
<td></td>
<td>TCP Flag: PSH</td>
</tr>
<tr>
<td>TCP ACK Number</td>
<td>TCP Flag: RST</td>
<td></td>
</tr>
<tr>
<td>TCP Header Length</td>
<td>TCP Flag: SYN</td>
<td></td>
</tr>
<tr>
<td>TCP Sequence Number</td>
<td>TCP Flag: URG</td>
<td></td>
</tr>
<tr>
<td>TCP Window-Size</td>
<td>UDP Message Length</td>
<td></td>
</tr>
<tr>
<td>TCP Source Port</td>
<td>UDP Source Port</td>
<td></td>
</tr>
<tr>
<td>TCP Destination Port</td>
<td>UDP Destination Port</td>
<td></td>
</tr>
<tr>
<td>TCP Urgent Pointer</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Flexible Flow Record: IPv6 Extension Header Map

<table>
<thead>
<tr>
<th>Bits 11-31</th>
<th>Bit 10</th>
<th>Bit 9</th>
<th>Bit 8</th>
<th>Bit 7</th>
<th>Bit 6</th>
<th>Bit 5</th>
<th>Bit 4</th>
<th>Bit 3</th>
<th>Bit 2</th>
<th>Bit 1</th>
<th>Bit 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Res</td>
<td>ESP</td>
<td>AH</td>
<td>PAY</td>
<td>DST</td>
<td>HOP</td>
<td>Res</td>
<td>UNK</td>
<td>FRA0</td>
<td>RH</td>
<td>FRA1</td>
<td>Res</td>
</tr>
</tbody>
</table>

- FRA1: Fragment header – not first fragment
- RH: Routing header
- FRA0: Fragment header – First fragment
- UNK: Unknown Layer 4 header (compressed, encrypted, not supported)
- HOP: Hop-by-hop extension header
- DST: Destination Options extension header
- PAY: Payload compression header
- AH: Authentication header
- ESP: Encapsulating Security Payload header
- Res: Reserved
Netflow Reverse Usage

- Scanning an IPv6 network is impossible (address space too large)
- How can we run a security audit?
- Easy
  - Get all IPv6 addresses from Netflow
  - Note: scanning link-local addresses requires layer-2 adjacency, i.e.
    - Ping6 ff02::1
Vulnerability Scanning in a Dual-Stack World

- Finding all hosts:
  - Address enumeration does not work for IPv6
  - Need to rely on DNS or NDP caches or NetFlow

- Vulnerability scanning
  - IPv4 global address, IPv6 global address(es) (if any), IPv6 link-local address
  - Some services are single stack only (currently mostly IPv4 but who knows...)
  - Personal firewall rules could be different between IPv4/IPv6

- **IPv6 vulnerability scanning MUST be done for IPv4 & IPv6 even in an IPv4-only network**
  - IPv6 link-local addresses are active by default
Forensic
Multiple Facets to IPv6 Addresses

- Every host can have multiple IPv6 addresses simultaneously
  - Need to do correlation!
  - Alas, no Security Information and Event Management (SIEM) supports IPv6
  - Usually, a customer is identified by its /48 😃

- Every IPv6 address can be written in multiple ways
  - 2001:0DB8:0BAD::0DAD
  - 2001:DB8:BAD:0:0:0:DAD
  - 2001:db8:bad::dad (this is the canonical RFC 5952 format)
  - => Grep cannot be used anymore to sieve log files…
#!/usr/bin/perl -w
use strict;
use Socket;
use Socket6;

my (@words, $word, $binary_address, $address);

$address = inet_pton AF_INET6, $ARGV[0];
if (! $address) { die "Wrong IPv6 address passed as argument" ; }

## go through the file one line at a time
while (my $line = <STDIN>) {
    @words = split /
\n\n\[/, $line;
    foreach $word (@words) {
        $binary_address = inet_pton AF_INET6, $word;
        if ($binary_address and $binary_address eq $address) {
            print $line;
        next;
        }
    }
}
How to Find the MAC Address of an IPv6 Address?

- Easy if EUI-64 format as MAC is embedded
  - 2001:db8::0226:bbff:fe4e:9434
    - (need to toggle bit 0x20 in the first MAC byte = U/L)
  - Is 00:26:bb:4e:94:34
How to Find the MAC Address of an IPv6 Address?

- DHCPv6 address or prefix… the client DHCP Unique ID (DUID) can be
  - MAC address: trivial
  - Time + MAC address: simply take the last 6 bytes
  - Vendor number + any number: no luck… next slide can help
  - No guarantee of course that DUID includes the real MAC address.

# show ipv6 dhcp binding
Client: FE80::225:9CFF:FEDC:7548
  DUID: 000100010000000A00259CDC7548
Username: unassigned
Interface: FastEthernet0/0
IA PD: IA ID 0x0000007B, T1 302400, T2 483840
  Prefix: 2001:DB8:612::/48
    preferred lifetime 3600, valid lifetime 3600
    expires at Nov 26 2010 01:22 PM (369)
DHCPv6 in Real Live...

- Not so attractive 😞
- Only supported in Windows Vista, and Windows 7, Max OS/X Lion
  - Not in Linux (default installation), …
- Windows Vista does not place the used MAC address in DUID but any MAC address of the PC

```
# show ipv6 dhcp binding
Client: FE80::FDFA:CB28:10A9:6DD0
  DUID: 0001000110DB0EA6001E33814DEE
  Username : unassigned
  IA NA: IA ID 0x1000225F, T1 300, T2 480
  Address: 2001:DB8::D09A:95CA:6918:967
  preferred lifetime 600, valid lifetime 600
  expires at Oct 27 2010 05:02 PM (554 seconds)
```

Actual MAC address: 0022.5f43.6522
How to Find the MAC Address of an IPv6 Address?

- Last resort… look in the live NDP cache (CLI or SNMP)
  
  ```
  #show ipv6 neighbors 2001:DB8::6DD0
  IPv6 Address      Age Link-layer Addr State Interface
  2001:DB8::6DD0      8 0022.5f43.6522 STALE Fa0/1
  ```

- If no more in cache, then you should have scanned and saved the cache…
- EEM can be your friend
- First-Hop Security phase II can generate a syslog event on each new binding
  - `ipv6 neighbor binding logging`
Summary
Our journey...

- Management Plane
- Control Plane
  - Routing Information
  - Neighbor Discovery
  - Control Plane Protection
- Data Plane
  - Anti-spoofing
  - Access Control List
  - Tunnel loops
- Telemetry
- Forensic
- Summary
Key Takeaway /1

- **Management plane**
  - Protect management plane with access-class

- **Control plane**
  - Authenticate IGP
  - Consider the use of link-local on P-P links?
  - Mitigate rogue-RA with RA-guard
  - Configure control plane policing

- **Data plane**
  - Beware of ping-pong on not /127 real P2P link
  - Apply anti-spoofing, anti-bogons
  - Disable source routing
  - Use ACL where applicable
    - ACL must permit NDP
Key Takeaway /2

- **Telemetry**
  - SNMP MIB and Netflow v9 are your friends
  - Netflow can be used for inventory

- **Forensic**
  - Multiple addresses per node, multiple ways to write an IPV6 address
  - Finding MAC address from IPv6:
    - EUI-64,
    - DHCPv6 (not so trivial)
    - else periodic NDP cache dumps...

- **Lawful Interception**
  - implemented, missing mediation device
Questions and Answers?