Basic Attacks and Mitigation Strategies

Christopher Werny, cwerny@ernw.de
Rafael Schaefer, rschaefer@ernw.de

Who we are

- Network geeks, working as security researcher for ERNW GmbH
  - Independent
  - Deep technical knowledge
  - Structured (assessment) approach
  - Business reasonable recommendations
  - We understand corporate
- Blog: www.insinuator.net

TROOPERS

WELCOME

to the IPv6 Security Summit

TROOPERS

ERNW providing security.

Day1 - March 14, 2016

Time	Day 1 Track 1	Day 1 Track 2
09:30	Developing an Enterprise IPv6 Security Strategy – Enno Rey	Basic IPv6 Attacks & Defenses. Hands-On Workshop – Rafael Schaefer, Christopher Werny
11:00	Break	
11:15	The Impact of Extension Headers on IPv6 Access Control Lists - Real Life Use Cases – Antonios Atlasis	Basic IPv6 Attacks & Defenses. Hands-On Workshop Part 2 – Rafael Schaefer, Christopher Werny
12:00	Security Aspects of IPv6 Multi-Interface and Source/Destination Routing – Eric Vyncke	Basic IPv6 Attacks & Denfenses. Hands-On workshop Part 3 – Rafael Schaefer, Christopher Werny
12:45	Lunch	

TROOPERS

ERNW
providing security.

12:45	Lunch	
13:45	NATTED - A Field Report – Gabriel Müller	Advanced IPv6 Network Reconnaissance – Fernando Gont
15:15	Break	
15:30	IPv6 First Hop Security Features on HP Devices – Christopher Werny	Security Assessment of Microsoft DirectAccess – Ali Hardudi
16:15 17:00	IPv6 First Hop Security Features on HP Devices continued – Christopher Werny	Anonymization IPv6 in PCAPs - Challenges and Wins – Jasper Bongertz

TROOPERS

ERNW providing security.

Day2 - March 15, 2016

Time	Day 2 Track 1	Day 2 Track 2	Day 2 Track 3
09:30	Building a Reliable and Secure IPv6 WiFi Network – Christopher Werny	Automating IPv6 Deployments – Ivan Pepelnjak	IPv6 in Wireshark Workshop – Jeff Carrell
10:15	Building a Reliable and Secure IPv6 WiFi Network – Christopher Werny	Protecting Hosts in IPv6 Networks – Enno Rey	IPv6 in Wireshark Workshop – Jeff Carrell
11:00		Break	
11:15	Remote Access and Business Partner Connections – Enno Rey	Recent IPv6 Standardization Efforts – Fernando Gont	IPv6 in Wireshark Workshop – Jeff Carrell
12:00	Remote Access and Business Partner Connections continued – Enno Rey	Recent IPv6 Standardization Efforts continued – Fernando Gont	IPv6 in Wireshark Workshop – Jeff Carrell
12:45		Lunch	

TROOPERS

ERNW
providing security.

12:45	Lunch		
13:45	Advanced IPv6 Attacks Using Chiron Training – Antonios Atlasis, Rafael Schaefer	Tools for Troubleshooting and Monitoring IPv6 Networks – Gabriel Müller	Security Evaluation of Dual-Stack Systems – Patrik Fehrenbach
15:15	Break		
15:30 17:00	Advanced IPv6 Attacks Using Chiron Training continued – Antonios Atlasis, Rafael Schaefer	Tools for Troubleshooting and Monitoring IPv6 Networks continued – Gabriel Müller	

Shared IPv6 Dinner

V!

- 7:30 PM
- Restaurant "Hirschgasse"
  - 50 min walk from PMA, but a scenic one
  - Bus from PMA leaves at 6:30 PM
  - You'll have to get back on your own, but we might be able to take/share cabs...

Agenda

- Part1:
  - IPv6 Refresher
  - Why IPv6 Security is so hard
- Part2:
  - Local-link IPv6 Security and Defense Strategies
- Part3:
  - Perimeter Security and Defense Strategies

IPv6 Refresher

Christopher Werny, cwerny@ernw.de
Rafael Schaefer, rschaefer@ernw.de

Current state of affairs or "some misconception"

"IPv6 is a well-defined set of completed standards."

- It's not!
- Still quite some debates on major fundamental elements.
- Lots of RFCs, both "standard track" and informational, and IETF drafts floating around.
- Vendors may implement fundamental stuff quite differently
  - E.g. how to get host part of address.

Some IPv6 Design Paradigms

- End-to-end principle / Network Transparency
  - NAT was never planned and there‘s still a “big debate“.
  - Only the “Hop Limit“-field supposed be changed by L3 hops.
- IPv6 is supposed to be used on a _large_ scale.
  - Mobile phones, sensors, smart meters, cars, fridges...
- IPv6 is supposed to be used by devices “not running in well-managed networks“.
  - Sensors, smart meters, fridges...
- IPv6 devices may be limited as for their processing and configuration capabilities.
  - Sensors, smart meters, fridges...
- Keep this in mind! This will help to better understand some design principles...

IPv6 Header Format (RFC 2460)

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|Version| Traffic Class |           Flow Label                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Payload Length        |  Next Header  |   Hop Limit   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                                                               +
|                                                               |
+                         Source Address                        +
|                                                               |
+                                                               +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                                                               +
|                                                               |
+                      Destination Address                      +
|                                                               |
+                                                               +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

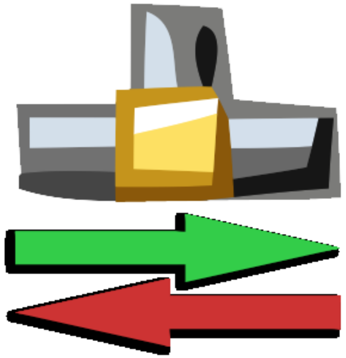
Extension Headers

```
+---------------+------------------------
|  IPv6 header  | TCP header + data
|               |
| Next Header = |
|      TCP      |
+---------------+------------------------


+---------------+----------------+------------------------
|  IPv6 header  | Routing header | TCP header + data
|               |                |
| Next Header = |  Next Header = |
|    Routing    |      TCP       |
+---------------+----------------+------------------------


+---------------+----------------+-----------------+----------------
|  IPv6 header  | Routing header | Fragment header | fragment of TCP
|               |                |                 |  header + data
| Next Header = |  Next Header = |  Next Header =  |
|    Routing    |    Fragment    |       TCP       |
+---------------+----------------+-----------------+----------------
```

Notation of IPv6 Addresses

- An IPv6 address is a 128 bit number. These 128 bits are used as eight 16-bit words and separated by colons. Each 16 bit word is represented by four hexadecimal digits:
  - fedc:ba98:7654:3210:0123:4567:89ab:cdef
- Prefixes are provided in the CIDR notation (Classless Inter-Domain Routing, RFC4632):
  - fe80:ba98:7600::/40 is a 40 bit long prefix.
- Some abbreviations are allowed. There's usually many zeroes:
  - 2001:0000:0000:0000:0008:0800:200c:417a

Notation of IPv6 Addresses



- A first simplification is to omit leading zeroes in each hex-combination
  - 2001:0:0:0:8:800:200c:417a
- The next consists of replacing consecutive zeros by using "::"
  - 2001::8:800:200c:417a
- This simplification can only be made once within an address.
- The following is the recommended way of including port numbers:
  - [2001:db8::1]:80
- See also: RFC 5952.
  But as well: http://labs.apnic.net/blabs/?p=309

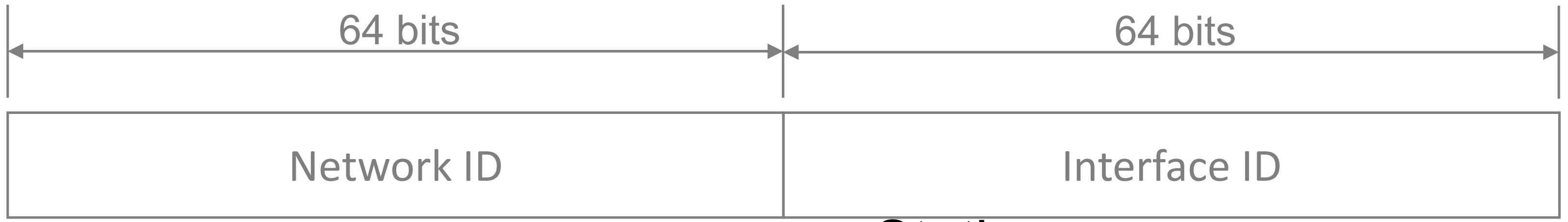
Address Space

- The IPv6 address space encompasses a total of 2 ^ 128 addresses (128-bit addresses).
- However, in IPv6 currently not all the addresses are “released by IANA”. As of 2014 the following areas are:

2000::/3	Global Unicast
FC00::/7	Unique Local Unicast
FE80::/10	Link Local Unicast
FF00::/8	Multicast

- **Also see: www.iana.org/assignments/ipv6-address-space for the current address allocation.**

How an Address is Composed



- Unicast
  - Link Local
  - Global
  - (ULA)
- Multicast

- Static
- “Automatic”
  - EUI-64
  - DHCPv6
  - Privacy Extensions
    - The Microsoft way
    - The “RFC way”
  - RFC 7217 et.al.

IPv6 interface ID generation

- Extended Unique Identifier (EUI)-64 Address
  - Is generated from the IEEE 802 Address
  - Default behavior on Windows XP, Windows Server 2003, FreeBSD and Linux, Mac OSX
    - Some Linux derivates (e.g. Ubuntu) and MAC OS-X “have changed their mind in the interim” → they default to PrivExtensions
  - Cisco:
    - interface INTERFACENAME
      - ipv6 address PREFIX/PREFLEN eui-64

- Randomly generated value (“Privacy Extensions”, RFC 4941)
  - Meant to counter address scanning
  - Hiding the identity
  - Default on Windows Vista, Windows Server 2008 und Windows 7

ICMPv6 (Internet Control Message Protocol for IPv6)

- ICMPv6 is the new version of ICMP.
  It was first specified in RFC 2462, latest in RFC 4443.
- ICMPv6 includes "traditional" ICMP functions, functionalities of IGMP (RFC 1112), IGMPv2 (RFC 2236) and extensions of the type "Multicast Listener Discovery" (MLD) for IPv6.
- Additionally ICMPv6 includes the Neighbor Discovery Protocol (RFC 2461, updated by RFC 4861).
- ICMPv6 is an integral part of every IPv6 implementation; every IPv6 stack must include ICMPv6.
- ICMPv6 has the next-header value 58.

(Main) ICMPv6 Types

Type(Value)	Description
1	Destination Unreachable (with codes 0,1,2,4)
2	Packet too big (Code 0)
3	Time Exceeded (Code 0,1)
4	Parameter Problem (Code 0,1,2)
128	Echo Request (Code 0)
129	Echo Reply (Code 0)
130	Multicast Listener Query
131	Multicast Listener Report
132	Multicast Listener Done
133	Router Solicitation
134	Router Advertisement
135	Neighbor Solitication
136	Neighbor Advertisement
137	Redirect

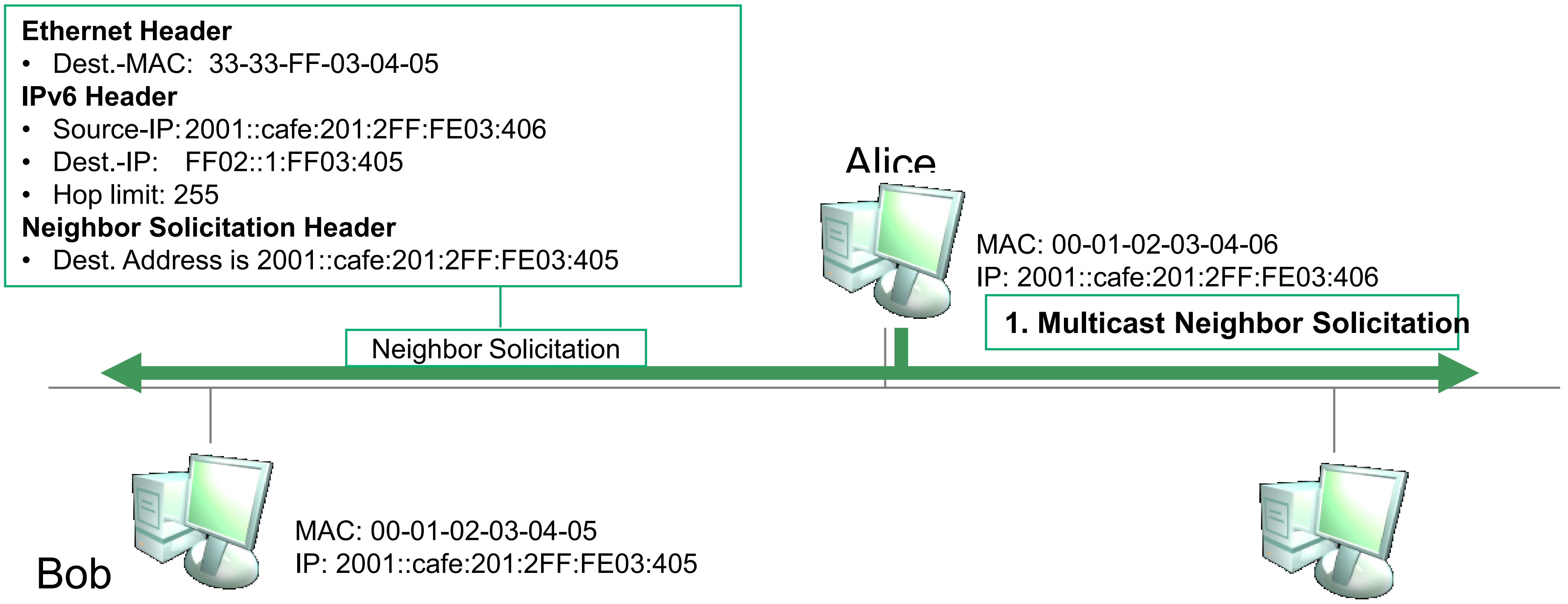
Neighbor Discovery Protocol RFC 4861

- *Neighbor Discovery* (ND) provides mechanisms for the following tasks:
  1. Neighbor Discovery / Address Resolution
  2. Router Discovery
  3. Prefix Discovery
  4. Parameter Discovery
  5. Address Autoconfiguration
  6. Next-Hop Determination
  7. Neighbor Unreachability Detection
  8. Duplicate Address Detection
  9. Redirect

Address resolution / Neighbor Discovery

- The address resolution is the exchange of *neighbor solicitation* and *neighbor advertisement* messages to the link-layer address, for example, to resolve the next hop.
  - Multicast Neighbor Solicitation Message
  - Unicast Neighbor Advertisement Message
- Both nodes involved update their *Neighbor Cache*.
- Once this is done successfully, the nodes can communicate with each other via unicast.
- Replaces the ARP (Address Resolution Protocol) in IPv4.

Neighbor Solicitation



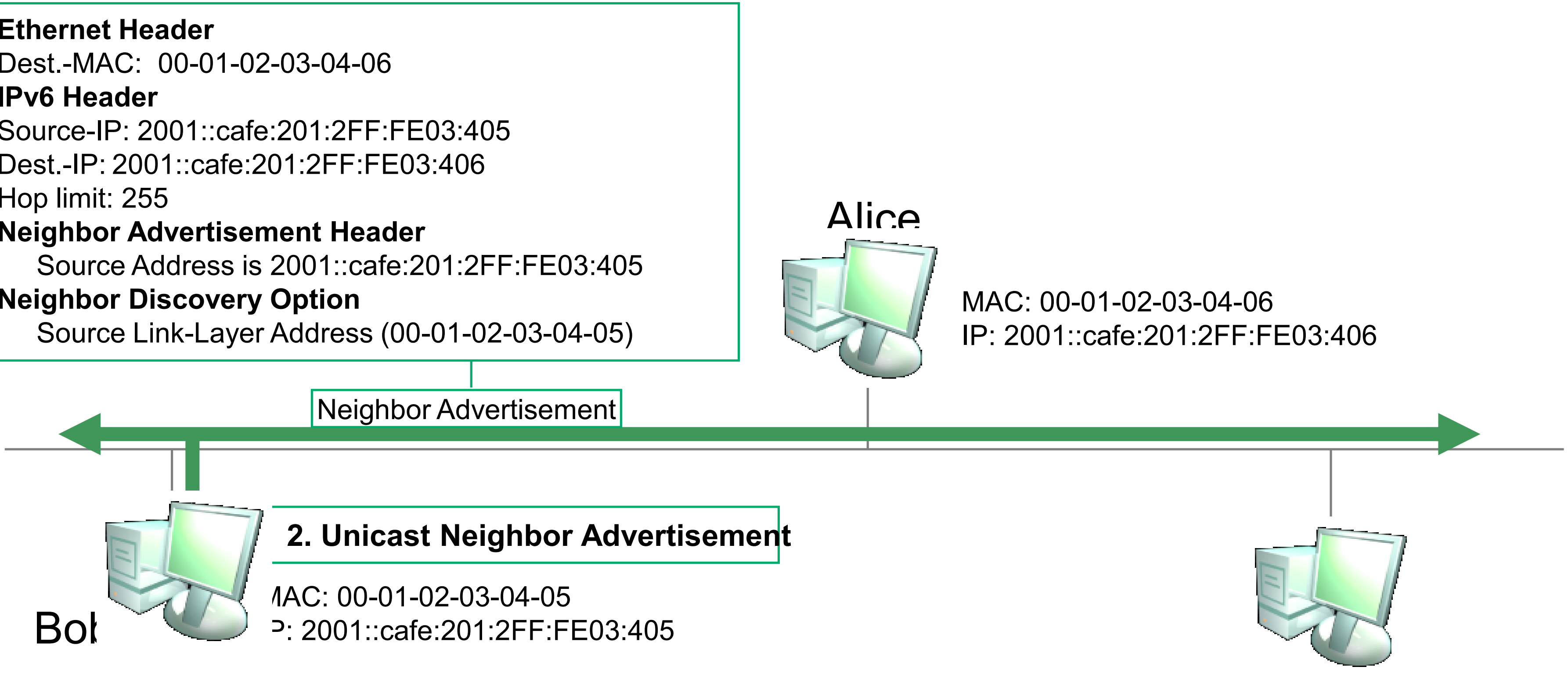
Ethernet Header
- Dest.-MAC: 33-33-FF-03-04-05

IPv6 Header
- Source-IP: 2001::cafe:201:2FF:FE03:406
- Dest.-IP: FF02::1:FF03:405
- Hop limit: 255

Neighbor Solicitation Header
- Dest. Address is 2001::cafe:201:2FF:FE03:405

Alice
MAC: 00-01-02-03-04-06
IP: 2001::cafe:201:2FF:FE03:406
1. Multicast Neighbor Solicitation
Neighbor Solicitation
Bob
MAC: 00-01-02-03-04-05
IP: 2001::cafe:201:2FF:FE03:405

Neighbor Advertisement


Ethernet Header
Dest.-MAC: 00-01-02-03-04-06
IPv6 Header
Source-IP: 2001::cafe:201:2FF:FE03:405
Dest.-IP: 2001::cafe:201:2FF:FE03:406
Hop limit: 255
Neighbor Advertisement Header
Source Address is 2001::cafe:201:2FF:FE03:405
Neighbor Discovery Option
Source Link-Layer Address (00-01-02-03-04-05)
Neighbor Advertisement
Alice
MAC: 00-01-02-03-04-06
IP: 2001::cafe:201:2FF:FE03:406
2. Unicast Neighbor Advertisement
MAC: 00-01-02-03-04-05
P: 2001::cafe:201:2FF:FE03:405
Bol

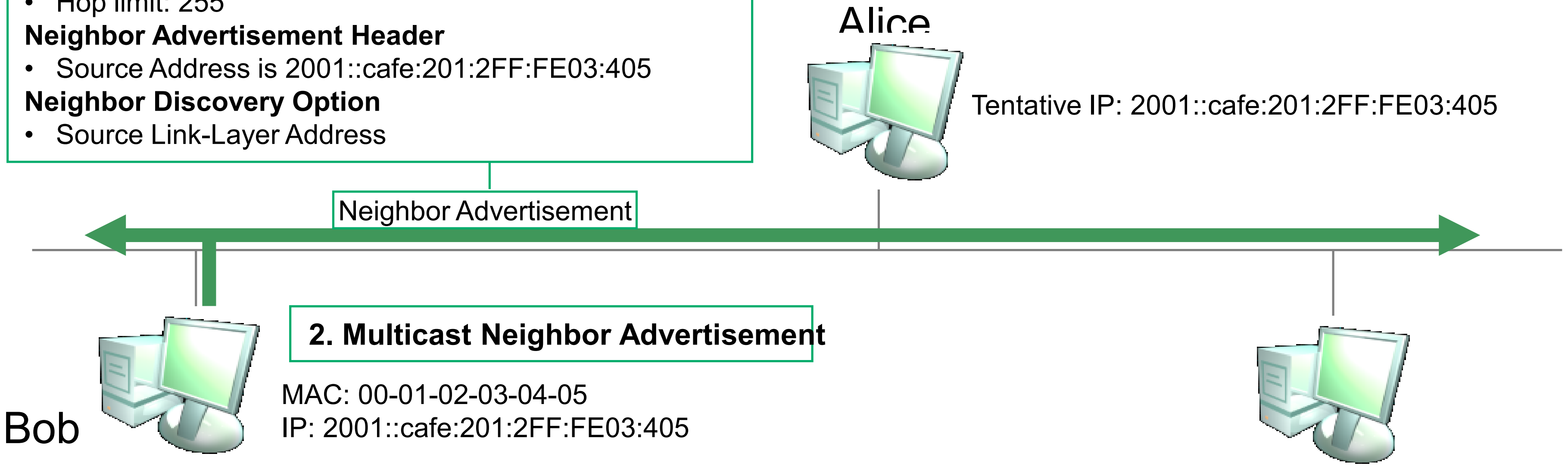
Multicast Neighbor Advertisement for Duplicate Address Detection

Ethernet Header
- Dest.-MAC: 33-33-00-00-00-01

IPv6 Header
- Source.-IP: 2001::cafe:201:2FF:FE03:405
- Dest.-IP: FF02::1
- Hop limit: 255

Neighbor Advertisement Header
- Source Address is 2001::cafe:201:2FF:FE03:405

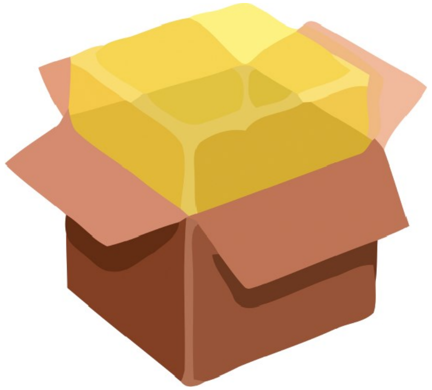
Neighbor Discovery Option
- Source Link-Layer Address



Alice
Tentative IP: 2001::cafe:201:2FF:FE03:405
Neighbor Advertisement
2. Multicast Neighbor Advertisement
MAC: 00-01-02-03-04-05
IP: 2001::cafe:201:2FF:FE03:405
Bob

Neighbor Cache

- Caching neighbor information / information delivered by NDP.
- Caching:
  - IPv6-Address → Link-Layer-Address
  - Further information, like
    - Pointer to packets, waiting for address resolution
    - Informations about reachability; is address a router?



Neighbor Cache entries

State	Description
INCOMPLETE	Neighbor Solicitation has been sent, but no Neighbor Advertisement has been retrieved.
REACHABLE	Positive confirmation was received within the last *ReachableTime* milliseconds, no special actions necessary
STALE	ReachableTime milliseconds have elapsed, no action takes place. This is entered upon receiving an unsolicited Neighbor Discovery message → entry must actually be used
DELAY	ReachableTime milliseconds have elapsed and a packet was sent within the last *DELAY_FIRST_PROBE_TIME* seconds. If no message was sent → change state to PROBE
PROBE	A reachability confirmation is actively sought by retransmitting Neighbor Solicitations every *RetransTimer* milliseconds until reachability confirmation is received

Router Discovery

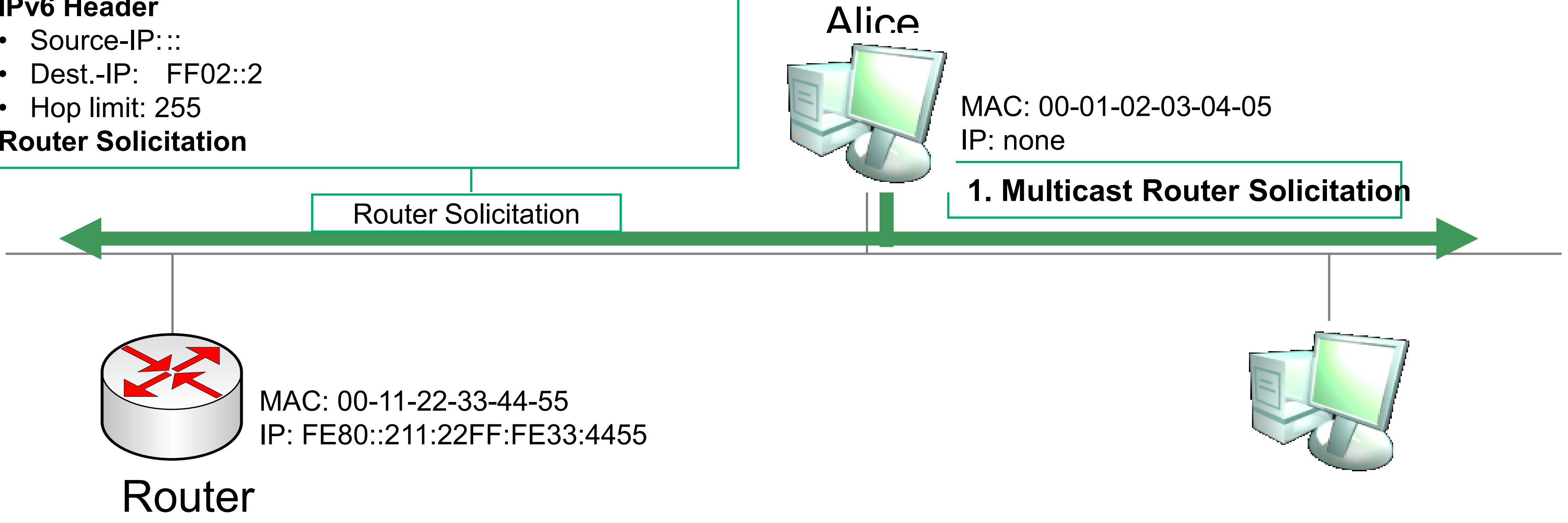
- Used to detect routers that are connected to the local network.
- IPv6 router discovery also provides the following information:
  - Default value for the "Hop Limit" field
  - Whether any "stateful address protocol" (DHCPv6) should be used.
  - Settings for the "Retransmission Timer"
  - The network prefix for the local network
  - The MTU of the network
  - Mobile IPv6 Information
  - Routing Information

Multicast Router Solicitation Message

Ethernet Header
- Dest.-MAC: 33-33-00-00-00-02

IPv6 Header
- Source-IP: ::
- Dest.-IP: FF02::2
- Hop limit: 255

Router Solicitation



Router Advertisement Message

Ethernet Header

- Dest.-MAC: 33-33-00-00-00-01

IPv6 Header

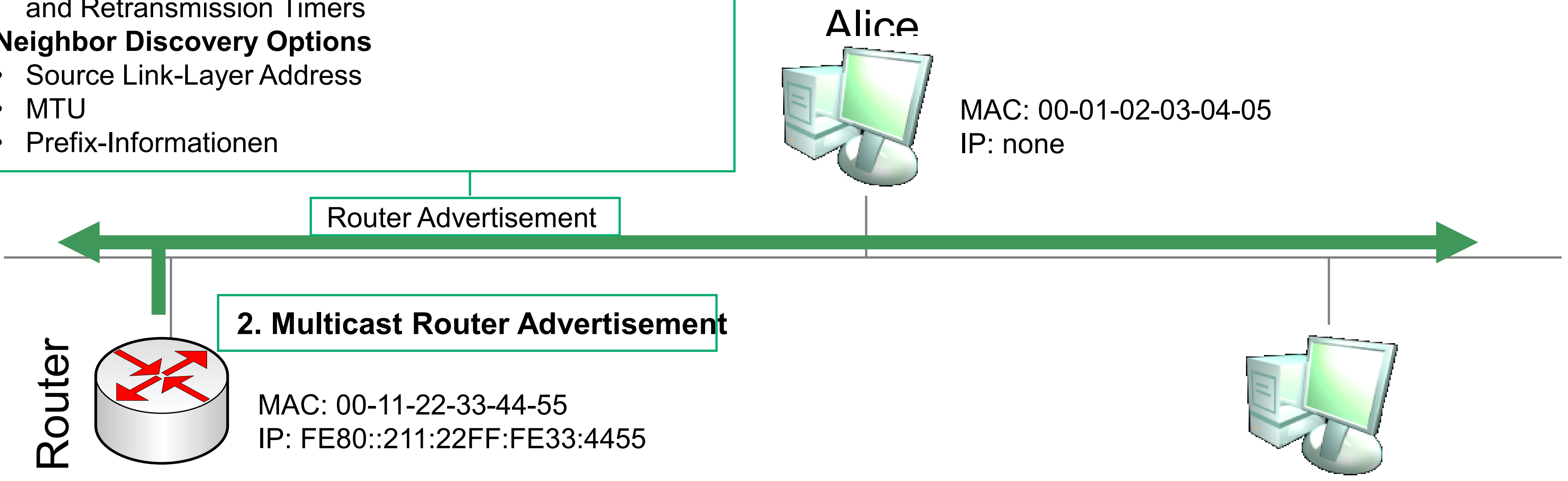
- Source-IP: FE80::211:22FF:FE33:4455
- Dest.-IP: FF02::1
- Hop limit: 255

Router Advertisement Header

- Current Hop Limit, Flags, Router Lifetime, Reachable and Retransmission Timers

Neighbor Discovery Options

- Source Link-Layer Address
- MTU
- Prefix-Informationen



Path MTU Discovery (RFC 1981)

- To discover the minimum MTU on a path, the following steps are performed
  - The IPv6 packet will be sent with the MTU of the local link.
  - If a router in the transit path cannot forward the packet (because of MTU issues), it will discard the packet and send an ICMPv6 ”Too Big“ packet back to the source, incl. the MTU which the source must use so that the router can forward the packet.
  - The source will transmit the packet again with the MTU specified in the ICMPv6 message.

Address Autoconfiguration Overview

- IPv6 interfaces are meant to configure themselves automatically, in terms of "basic IP parameters".
  - Even without DHCPv6.
  - In particular without DHCPv6!
    - Remember: IPv6 = consumer technology.
- Link-local addresses are always configured, for each interface.
- Using the *router discovery* process, other addresses, router addresses and other configuration parameters are selected.

Types of Autoconfiguration

- Stateless
  - Via *Router Advertisement Messages* (with one or more prefix)
  - Can (theoretically!) also distribute "other parameters", see RFC 6106.
  - SLAAC: “stateless address autoconfiguration“
- Stateful
  - Usage of a *Stateful Address Protocol* (e.g. DHCPv6).
- Stateless with DHCP
  - Use of Router Advertisement messages for allocation of prefixes
  - In addition, DHCP for "other parameters” (e.g. DNS Server, Domain Search List).

(In all cases there is always at least one link-local address anyway!)

Basic IP Config

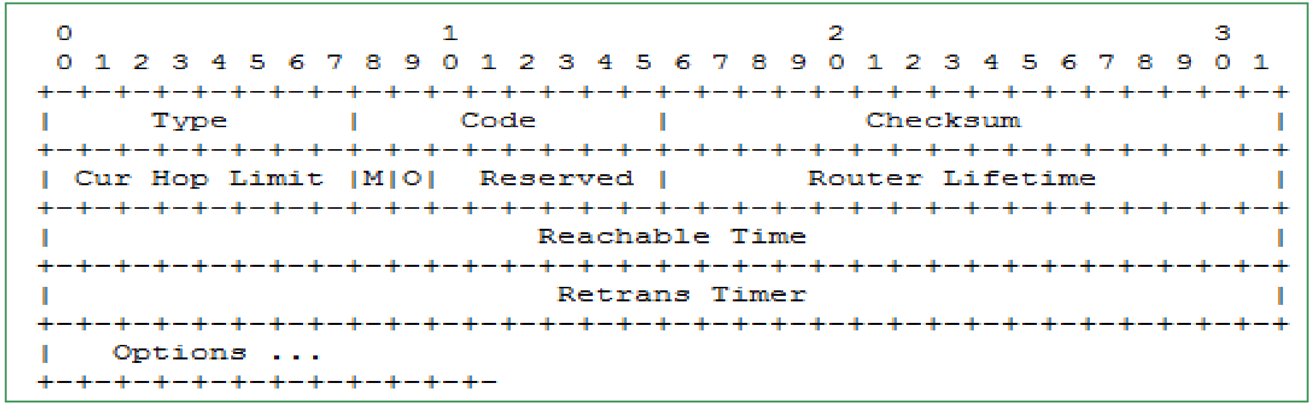
	Router Advertisements	DHCPv6
Address	P	P 🟥
Default Route	P	X
DNS Resolver	(RFC 6106)	P 🟩 🟥
All other options	X	P 🟥

🟩	O-Flag
🟥	M-Flag

Router Advertisements, Flags

- Routers can inform adjacent hosts (neighbors on the local link) that additional configuration parameters (like a DNS server) are available over a stateful configuration protocol (DHCPv6).

- In the router advertisement header two flags (M and O) can be included which can be set to inform the clients that additional configuration parameters are available.



```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |     Type      |     Code      |          Checksum             |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   | Cur Hop Limit |M|O|  Reserved |       Router Lifetime         |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                         Reachable Time                        |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                          Retrans Timer                        |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |   Options ...
   +-+-+-+-+-+-+-+-+-+-+-+-
```

Summary

- Different mode of operation
- Different design goals
- Lots of flexibility introduced into IPv6
  - I let you decide whether this is a good or bad thing in terms of security ;)

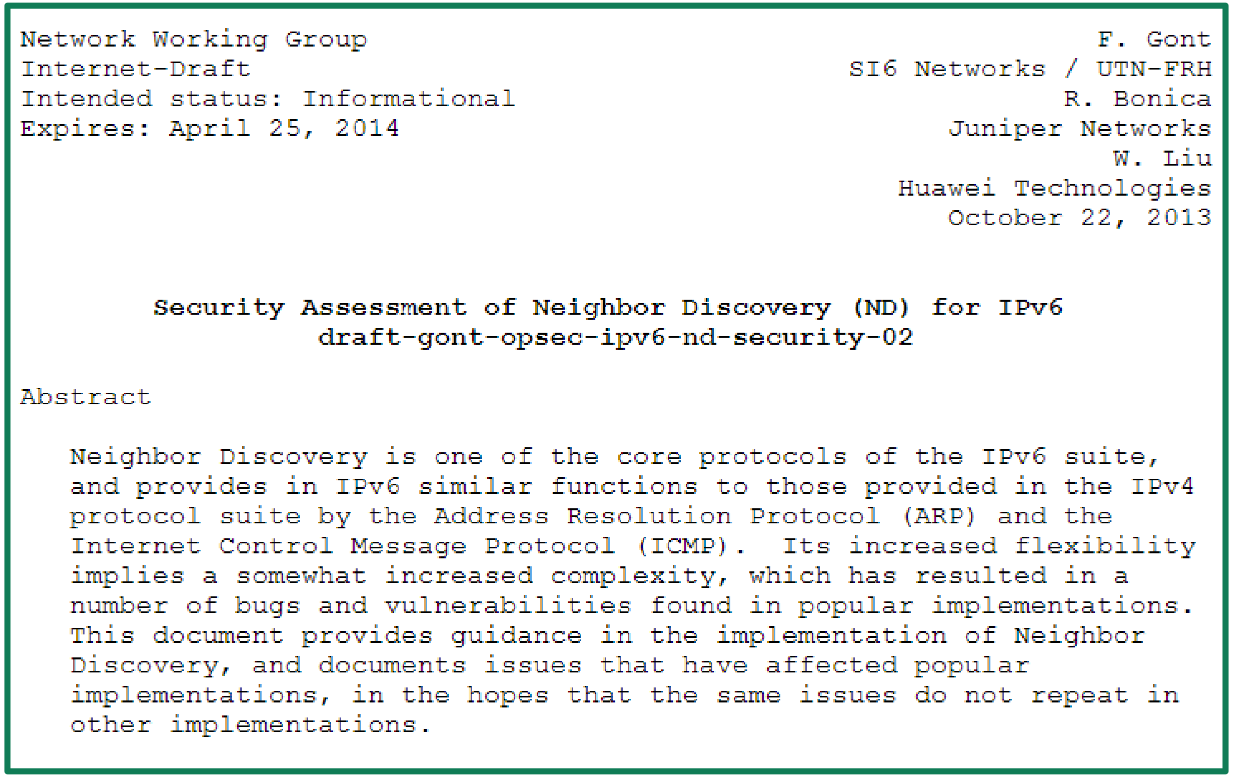
Why IPv6 Security Is So Hard

Why IPv6 Security Is So Hard

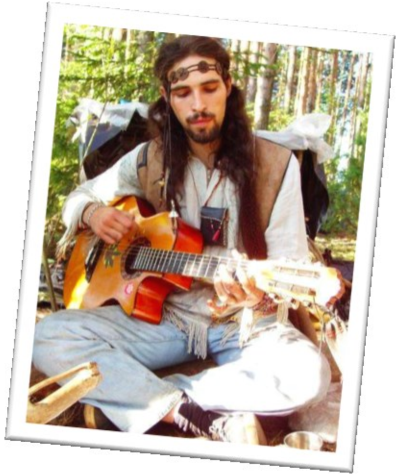
- Trust Model & Provisioning
- Crypto-Optimism
- Complexity
- The State Problem
- Stack Heterogeneity
- Attack / Defense Asymmetry

IPv6’s Trust Model

On the *local link* we’re all brothers.



```
Network Working Group                                            F. Gont
Internet-Draft                                   SI6 Networks / UTN-FRH
Intended status: Informational                                R. Bonica
Expires: April 25, 2014                                Juniper Networks
                                                                  W. Liu
                                                    Huawei Technologies
                                                       October 22, 2013


        Security Assessment of Neighbor Discovery (ND) for IPv6
                    draft-gont-opsec-ipv6-nd-security-02

Abstract

   Neighbor Discovery is one of the core protocols of the IPv6 suite,
   and provides in IPv6 similar functions to those provided in the IPv4
   protocol suite by the Address Resolution Protocol (ARP) and the
   Internet Control Message Protocol (ICMP).  Its increased flexibility
   implies a somewhat increased complexity, which has resulted in a
   number of bugs and vulnerabilities found in popular implementations.
   This document provides guidance in the implementation of Neighbor
   Discovery, and documents issues that have affected popular
   implementations, in the hopes that the same issues do not repeat in
   other implementations.
```

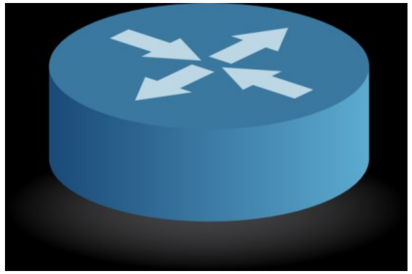


We’re All Brothers

We like the idea. Really.

As much as we like the concept of eternal happiness & peace.

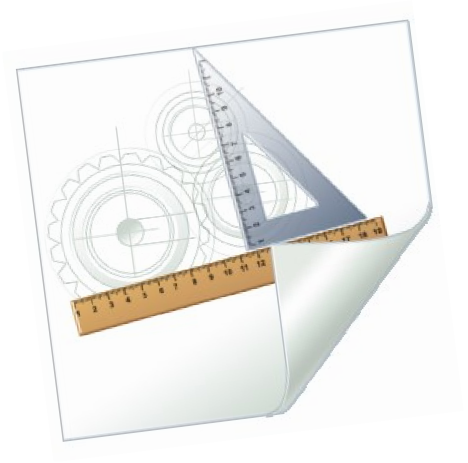
What's a *Router*?



- Wikipedia:
  - router = "a **router** is a device that forwards *data packets* between *computer networks*"
- RFC 2460:
  - router: "router - a node that forwards IPv6 packets not explicitly addressed to itself."
- Is there any issue then?

What's a *Router,* in IPv6?

Looking Closer

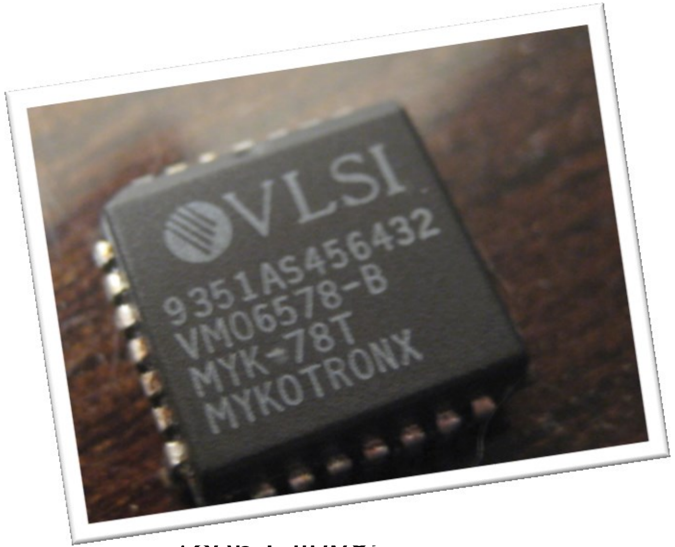


- RFC 2461: "Routers advertise their presence together with various link and Internet parameters either periodically, or in response to a Router Solicitation message".
- In the end of the day, in IPv6 a router is not just a forwarding device but a provisioning system as well.
  - As many other IPv6 guys we generally like the idea.
  - Still, having an operations background in large scale enterprise networks we can tell you quite some of our colleagues have a hard time with this.
  - While we're at it: MANY THANKS TO YOU GUYS OVER THERE AT IETF FOR THE BRILLIANT STATE OF RA & DHCPv6 "INTERACTION".
    - This really helps a lot with widespread IPv6 adoption. Rly!
  - That said we won't further open this can of worms here…

The 90’s “Crypto-Optimism”

- Every network security problem considered to be solvable by means of math & some algorithms.

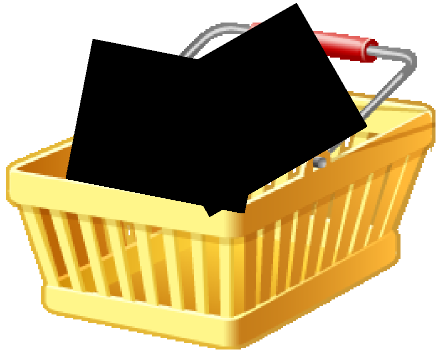
- This thinking shaped IPv6
  – RFC 3315 (DHCPv6) complemented by RFC 3318.
    – Which pretty much no DHCPv6 server supports…
  – RFC 2461 (ND, initial spec) by RFC 3971 (SeND).
    – Which pretty much no common desktop OS supports…
  – etc.



In 1995 Clipp

Complexity

Want some samples?



“ND overspecified”

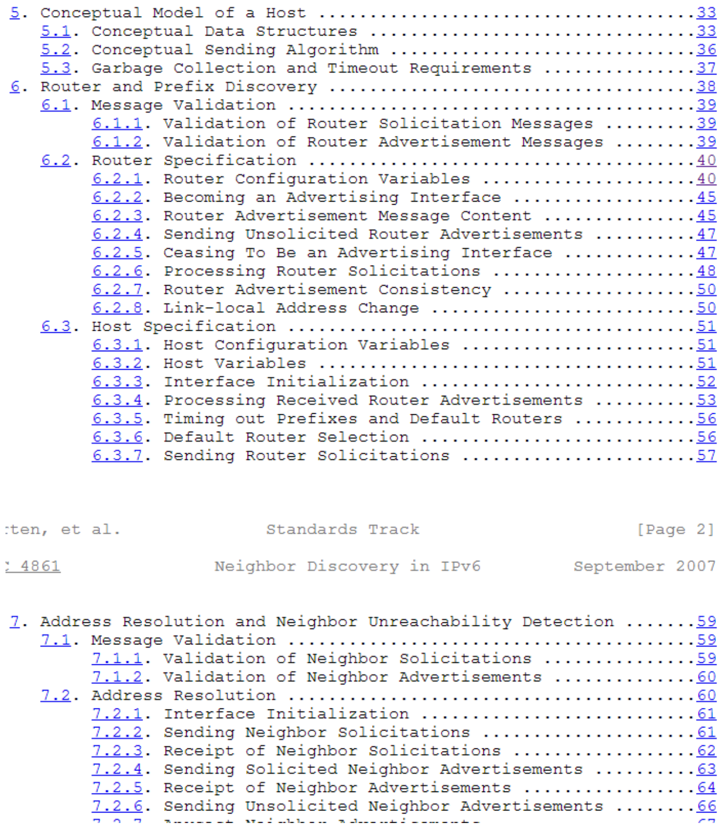
(one of the first statements in 6man at IETF 89 in London)

Neighbor Discovery

- Initial specification in RFC 1970 (Aug 1996, 82 pages), obsoleted by
- RFC 2461 (Dec 1998, 93 pages), obsoleted (after update via 4311) by
- RFC 4861 (Sep 2007, 97 pages)
  - This is mainly considered “the latest, stable one”, cited in most textbooks and – if existent – stack documentation.

RFC 4861

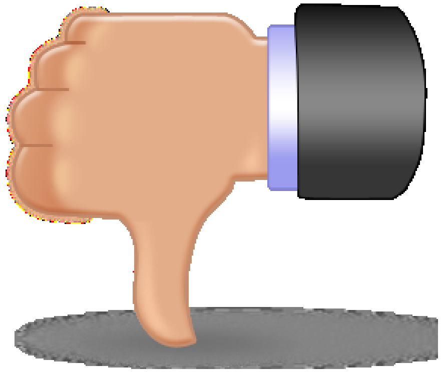
Small excerpt



:ten, et al.   Standards Track   [Page 2]

: 4861   Neighbor Discovery in IPv6   September 2007

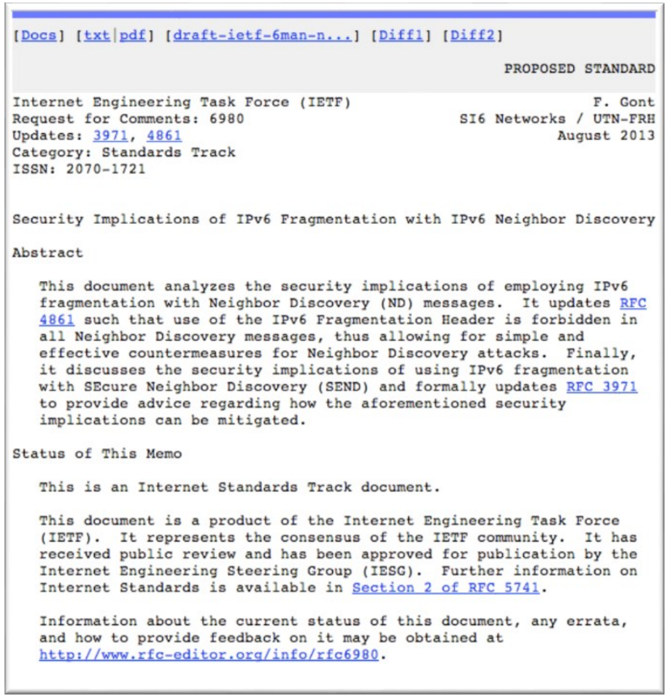
So We've Reached a kind-of stable State as for the Core of IPv6?



- Well… unfortunately… no.
- RFC 4861 updated by
  - RFC 5942
  - RFC 6980 *Security Implications of IPv6 Fragmentation with IPv6 Neighbor Discovery*
  - *RFC 7048*
  - *yadda yadda yadda*
- In Mar 2014, at IETF 89, in *6man* (IPv6 Maintenance) and *v6ops* (IPv6 Operations) significant time spent on…

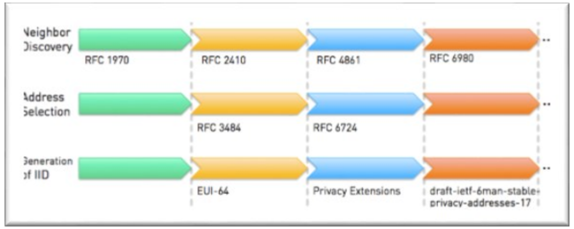
  … modifications of ND!

Let’s Have a Quick Look At RFC 6980



```
[Docs] [txt|pdf] [draft-ietf-6man-n...] [Diff1] [Diff2]

                                                     PROPOSED STANDARD

Internet Engineering Task Force (IETF)                           F. Gont
Request for Comments: 6980                        SI6 Networks / UTN-FRH
Updates: 3971, 4861                                          August 2013
Category: Standards Track
ISSN: 2070-1721


Security Implications of IPv6 Fragmentation with IPv6 Neighbor Discovery

Abstract

   This document analyzes the security implications of employing IPv6
   fragmentation with Neighbor Discovery (ND) messages.  It updates RFC
   4861 such that use of the IPv6 Fragmentation Header is forbidden in
   all Neighbor Discovery messages, thus allowing for simple and
   effective countermeasures for Neighbor Discovery attacks.  Finally,
   it discusses the security implications of using IPv6 fragmentation
   with SEcure Neighbor Discovery (SEND) and formally updates RFC 3971
   to provide advice regarding how the aforementioned security
   implications can be mitigated.

Status of This Memo

   This is an Internet Standards Track document.

   This document is a product of the Internet Engineering Task Force
   (IETF).  It represents the consensus of the IETF community.  It has
   received public review and has been approved for publication by the
   Internet Engineering Steering Group (IESG).  Further information on
   Internet Standards is available in Section 2 of RFC 5741.

   Information about the current status of this document, any errata,
   and how to provide feedback on it may be obtained at
   http://www.rfc-editor.org/info/rfc6980.
```

- From a security perspective this can be considered long over-due
  - Think attack/defense asymmetry (see below)
- Still, it adds complexity to decision taking and, subsequently, stack code.
  - And yet another sector on the time-bar.



- It doesn’t end here…
  - There’s `draft-gont-6man-lla-opt-validation-00 Validation of Neighbor Discovery Source Link-Layer Address (SLLA) and Target Link-layer Address (TLLA) options`
  - → ask Fernando for details.
  - → even more checks a stack might have to perform…

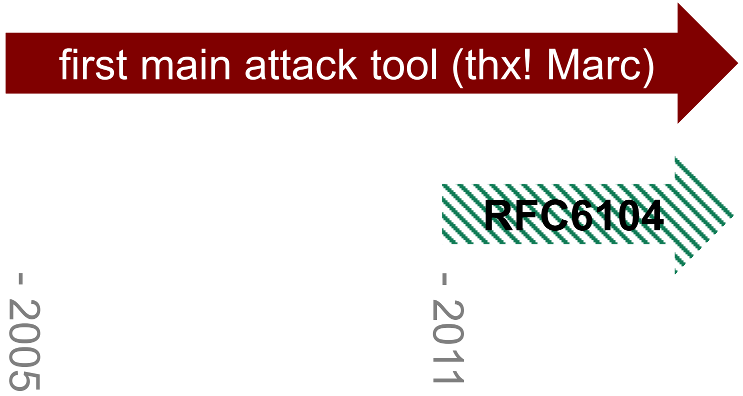
TROOPERS

ERNW providing security.

There's Different Generations of IPv6 Stacks

Neighbor Discovery	RFC 1970	RFC 2410	RFC 4861	RFC 6980	…
Address Selection		RFC 3484	RFC 6724		…
Generation of IID		EUI-64	Privacy Extensions	RFC 7217	…
et.al.		◂ RFC XXX	◂ RFC XXX	◂ RFC XXX	…

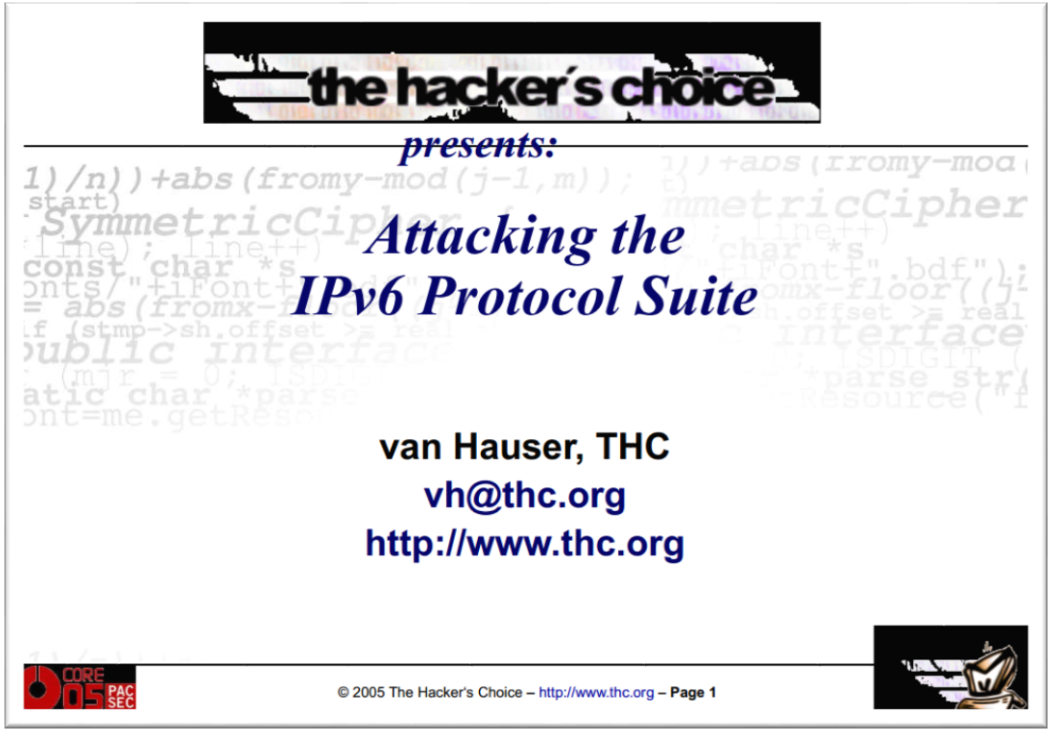
Attack / Defense Asymmetry



- Due to long IPv6 “warm up phase” there’s a huge asymmetry between attackers and defenders.
  - *THC-IPV6* was initially released in 2005.
  - RFC 6104 describing RA Guard is from February 2011!
    - And RA Guard still doesn’t work sufficiently. And probably never will.

Asymmetry

http://pacsec.jp/psj05/psj05-vanhauser-en.pdf



the hacker's choice

presents:

Attacking the IPv6 Protocol Suite

van Hauser, THC
vh@thc.org
http://www.thc.org

© 2005 The Hacker's Choice – http://www.thc.org – **Page 1**

Last but not Least

- IPv6 is very different from IPv4
  - So is IPv6 security.
- Don‘t rely on transforming v4 models 1:1 to v6. Do not!
- Think *feature suitability* instead.

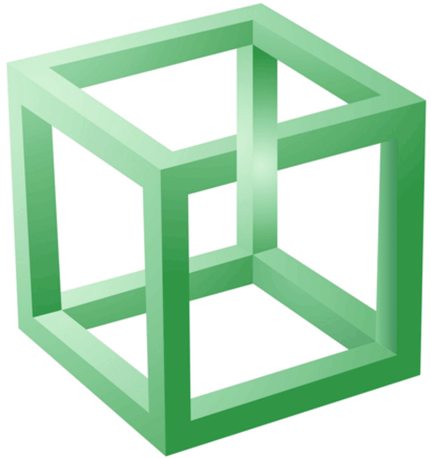
IPv6 Security Fundamentals

IPv6 Attacking Scene

- Reconnaissance
- Network Scanning
- Attacks at the Local Link
  - Neighbor Discovery Attacks
  - IPv6 Router-related attacks
  - MLD Attacks
- Routing Headers Attacks
- Covert Channels
- Remote DoS Attacks
- Fragmentation
- Abusing IPv6 Extension Headers

IPv6-Specific Attacking Frameworks

- “The Hackers Choice” ***thc-ipv6*** attacking framework
  https://www.thc.org/thc-ipv6/
- Si6 Networks ***ipv6-toolkit***
  http://www.si6networks.com/tools/ipv6toolkit/
- ***Chiron*** http://www.secfu.net/tools-scripts/
- Each of them supports plenty of other tools/options.
  - sometime with overlapping features/capabilities
  - but they are also complementary.



thc-ipv6 (sample) list of tools

- ***parasite6***: icmp neighbor solitication/advertisement spoofer, puts you as man-in-the-middle, same as ARP mitm (and parasite)
- ***alive6***: an effective alive scanning, which will detect all systems listening to this address
- ***fake_router6***: announce yourself as a router on the network, with the highest priority
- ***redir6***: redirect traffic to you intelligently (man-in-the-middle) with a clever icmp6 redirect spoofer
- ***toobig6***: mtu decreaser with the same intelligence as redir6
- ***flood_router6***: flood a target with random router advertisements
- ***flood_advertise6***: flood a target with random neighbor advertisements
- ***denial6***: a collection of denial-of-service tests againsts a target
- ***fake_mld6***: announce yourself in a multicast group of your choice on the net
- ***fake_mld26***: same but for MLDv2
- ***fake_mldrouter6***: fake MLD router messages
- ***fake_advertiser6***: announce yourself on the network
- ***smurf6***: local smurfer
- ***thcping6***: sends a hand crafted ping6 packet

IPv6-toolkit

- ***addr6***: An IPv6 address analysis and manipulation tool.
- ***flow6***: A tool to perform a security asseessment of the IPv6 Flow Label.
- **frag6**: A tool to perform IPv6 fragmentation-based attacks and to perform a security assessment of a number of fragmentation-related aspects.
- ***icmp6***: A tool to perform attacks based on ICMPv6 error messages.
- ***jumbo6***: A tool to assess potential flaws in the handling of IPv6 Jumbograms.
- ***na6***: A tool to send arbitrary Neighbor Advertisement messages.
- ***ni6***: A tool to send arbitrary ICMPv6 Node Information messages, and assess possible flaws in the processing of such packets.
- ***ns6***: A tool to send arbitrary Neighbor Solicitation messages.
- ***ra6***: A tool to send arbitrary Router Advertisement messages.
- ***rd6***: A tool to send arbitrary ICMPv6 Redirect messages.
- ***rs6***: A tool to send arbitrary Router Solicitation messages.
- ***scan6***: An IPv6 address scanning tool.
- ***tcp6***: A tool to send arbitrary TCP segments and perform a variety of TCP-based attacks.

Our Goal

- We will try to cover a wide range of IPv6 related attacks
  - Some very common and well known
- In order to get the “big picture”
- And to be prepared!

Attacks On The Local Link

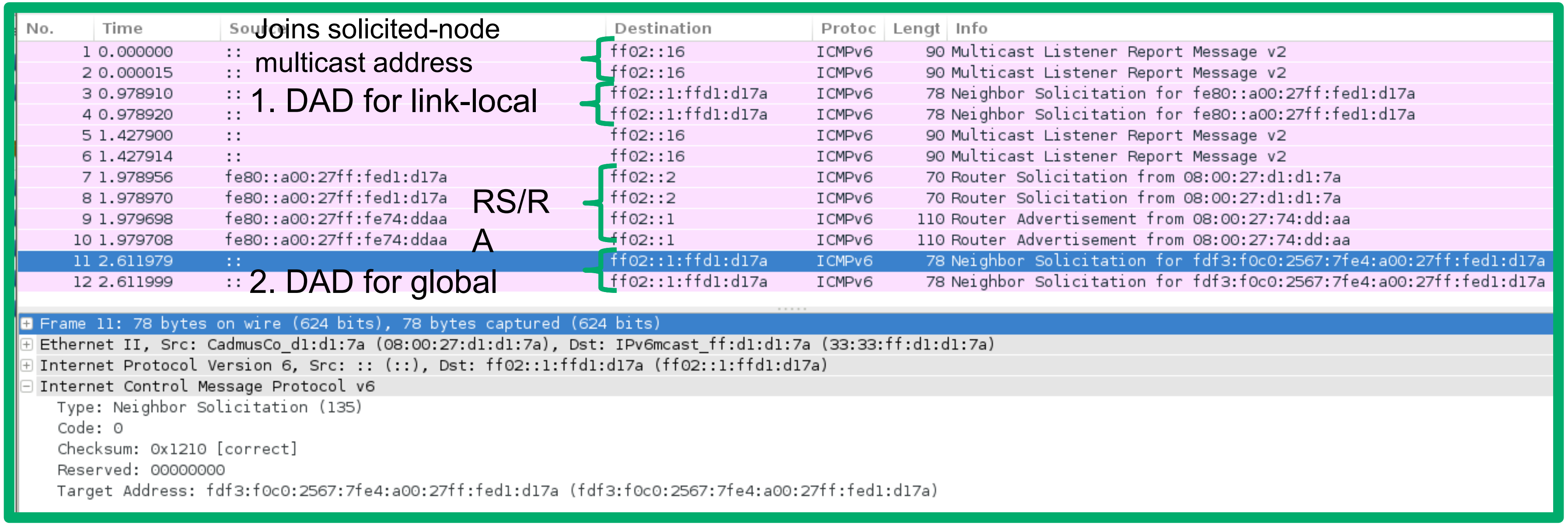
Neighbor Discovery

Attacks On The Local Link

- Two families of attacks
  - Attacks related with the Neighbor Discovery (ND) process
    - NS – NA messages
    - DAD
  - Attacks related with IPv6 Router
    - RA
  - Other attacks
    - With MLD etc. (Not covered in this Workshop)

Attacks Related with the Neighbor Discovery Process

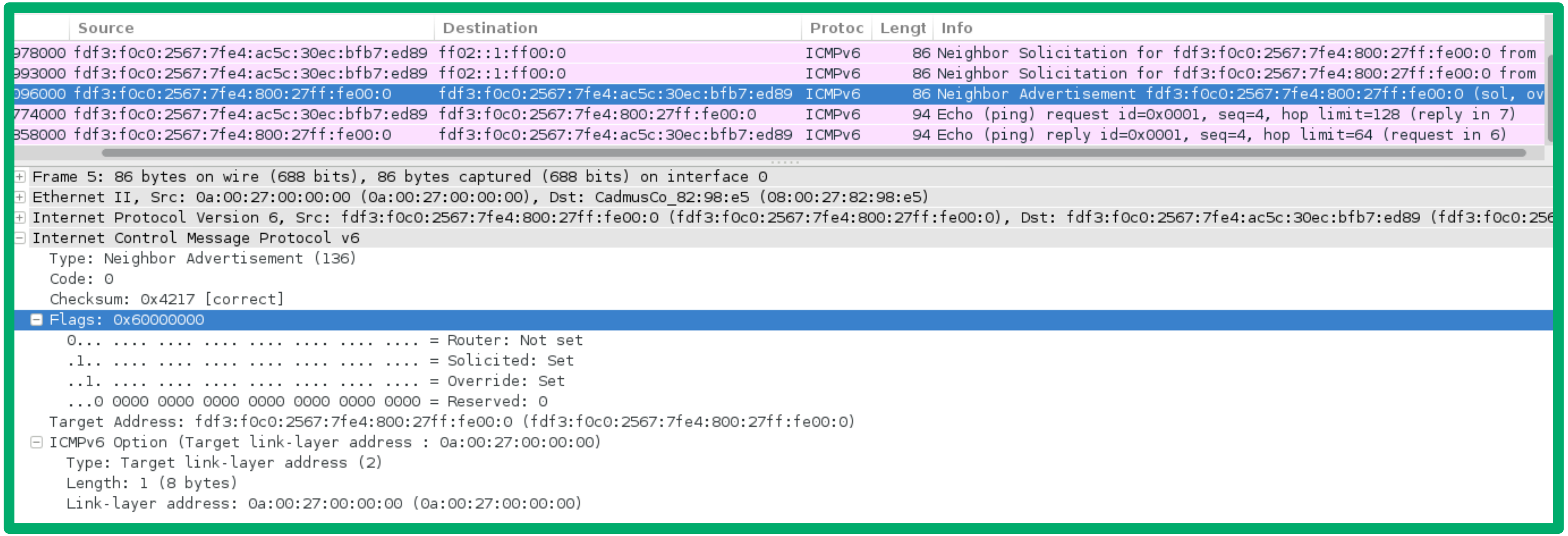
Duplicate Address Detection during SLAAC



```
No.  Time      Sou                              Destination           Protoc  Lengt  Info
  1  0.000000  ::                               ff02::16              ICMPv6     90  Multicast Listener Report Message v2
  2  0.000015  ::                               ff02::16              ICMPv6     90  Multicast Listener Report Message v2
  3  0.978910  ::                               ff02::1:ffd1:d17a     ICMPv6     78  Neighbor Solicitation for fe80::a00:27ff:fed1:d17a
  4  0.978920  ::                               ff02::1:ffd1:d17a     ICMPv6     78  Neighbor Solicitation for fe80::a00:27ff:fed1:d17a
  5  1.427900  ::                               ff02::16              ICMPv6     90  Multicast Listener Report Message v2
  6  1.427914  ::                               ff02::16              ICMPv6     90  Multicast Listener Report Message v2
  7  1.978956  fe80::a00:27ff:fed1:d17a         ff02::2               ICMPv6     70  Router Solicitation from 08:00:27:d1:d1:7a
  8  1.978970  fe80::a00:27ff:fed1:d17a         ff02::2               ICMPv6     70  Router Solicitation from 08:00:27:d1:d1:7a
  9  1.979698  fe80::a00:27ff:fe74:ddaa         ff02::1               ICMPv6    110  Router Advertisement from 08:00:27:74:dd:aa
 10  1.979708  fe80::a00:27ff:fe74:ddaa         ff02::1               ICMPv6    110  Router Advertisement from 08:00:27:74:dd:aa
 11  2.611979  ::                               ff02::1:ffd1:d17a     ICMPv6     78  Neighbor Solicitation for fdf3:f0c0:2567:7fe4:a00:27ff:fed1:d17a
 12  2.611999  ::                               ff02::1:ffd1:d17a     ICMPv6     78  Neighbor Solicitation for fdf3:f0c0:2567:7fe4:a00:27ff:fed1:d17a
```

```
⊞ Frame 11: 78 bytes on wire (624 bits), 78 bytes captured (624 bits)
⊞ Ethernet II, Src: CadmusCo_d1:d1:7a (08:00:27:d1:d1:7a), Dst: IPv6mcast_ff:d1:d1:7a (33:33:ff:d1:d1:7a)
⊞ Internet Protocol Version 6, Src: :: (::), Dst: ff02::1:ffd1:d17a (ff02::1:ffd1:d17a)
⊟ Internet Control Message Protocol v6
    Type: Neighbor Solicitation (135)
    Code: 0
    Checksum: 0x1210 [correct]
    Reserved: 00000000
    Target Address: fdf3:f0c0:2567:7fe4:a00:27ff:fed1:d17a (fdf3:f0c0:2567:7fe4:a00:27ff:fed1:d17a)
```

Neighbor Solicitation/Advertisement Process



```
        Source                                   Destination                              Protoc  Lengt  Info
978000 fdf3:f0c0:2567:7fe4:ac5c:30ec:bfb7:ed89  ff02::1:ff00:0                           ICMPv6      86 Neighbor Solicitation for fdf3:f0c0:2567:7fe4:800:27ff:fe00:0 from
993000 fdf3:f0c0:2567:7fe4:ac5c:30ec:bfb7:ed89  ff02::1:ff00:0                           ICMPv6      86 Neighbor Solicitation for fdf3:f0c0:2567:7fe4:800:27ff:fe00:0 from
096000 fdf3:f0c0:2567:7fe4:800:27ff:fe00:0      fdf3:f0c0:2567:7fe4:ac5c:30ec:bfb7:ed89  ICMPv6      86 Neighbor Advertisement fdf3:f0c0:2567:7fe4:800:27ff:fe00:0 (sol, ov
774000 fdf3:f0c0:2567:7fe4:ac5c:30ec:bfb7:ed89  fdf3:f0c0:2567:7fe4:800:27ff:fe00:0      ICMPv6      94 Echo (ping) request id=0x0001, seq=4, hop limit=128 (reply in 7)
858000 fdf3:f0c0:2567:7fe4:800:27ff:fe00:0      fdf3:f0c0:2567:7fe4:ac5c:30ec:bfb7:ed89  ICMPv6      94 Echo (ping) reply id=0x0001, seq=4, hop limit=64 (request in 6)

+ Frame 5: 86 bytes on wire (688 bits), 86 bytes captured (688 bits) on interface 0
+ Ethernet II, Src: 0a:00:27:00:00:00 (0a:00:27:00:00:00), Dst: CadmusCo_82:98:e5 (08:00:27:82:98:e5)
+ Internet Protocol Version 6, Src: fdf3:f0c0:2567:7fe4:800:27ff:fe00:0 (fdf3:f0c0:2567:7fe4:800:27ff:fe00:0), Dst: fdf3:f0c0:2567:7fe4:ac5c:30ec:bfb7:ed89 (fdf3:f0c0:256
- Internet Control Message Protocol v6
    Type: Neighbor Advertisement (136)
    Code: 0
    Checksum: 0x4217 [correct]
  - Flags: 0x60000000
      0... .... .... .... .... .... .... .... = Router: Not set
      .1.. .... .... .... .... .... .... .... = Solicited: Set
      ..1. .... .... .... .... .... .... .... = Override: Set
      ...0 0000 0000 0000 0000 0000 0000 0000 = Reserved: 0
    Target Address: fdf3:f0c0:2567:7fe4:800:27ff:fe00:0 (fdf3:f0c0:2567:7fe4:800:27ff:fe00:0)
  - ICMPv6 Option (Target link-layer address : 0a:00:27:00:00:00)
      Type: Target link-layer address (2)
      Length: 1 (8 bytes)
      Link-layer address: 0a:00:27:00:00:00 (0a:00:27:00:00:00)
```

Neighbor Discovery Related Attacks - DAD

- Attacks against Duplicate Address Detection – DAD (for DoS)
  - Against link-local address (phase 1) => needs intervention of the administrator
  - Against global unicast address (phase 3)
- DAD should be performed for all unicast addresses (obtained though SLAAC, DHCPv6 or static).

Neighbor Discovery Related Attacks - ND

- Attacks against Other Nodes (for DoS or MITM purposes)
  - Spoofed NS → populate victim's Neighbor Cache → DoS for legitimate hosts.
  - Reply with spoofed NA to NS (race condition with legitimate host) → DoS/ MiTM
  - Unsolicited Spoofed NAs → DoS or MiTM

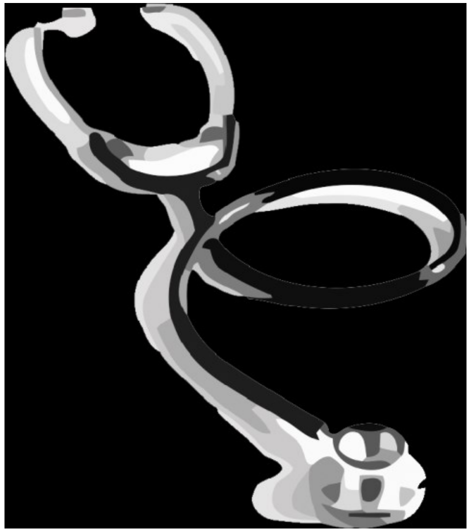
Spoofed NAs to NS

- You can use thc-ipv6 *parasite6*
- It can be used for DoS / MiTM attacks.
- NOTE: It will redirect ALL local traffic.

- ./parasite6 vboxnet0 0a:00:27:00:00:00 -l –R

```
Remember to enable routing (ip_forwarding), you will denial service otherwise!
 =>  echo 1 > /proc/sys/net/ipv6/conf/all/forwarding
Started ICMP6 Neighbor Solitication Interceptor (Press Control-C to end) ...
Spoofed packet to fdf3:f0c0:2567:7fe4:ac5c:30ec:bfb7:ed89 as fdf3:f0c0:2567:7fe4:a00:27ff:fed1:d17a
Spoofed packet to fdf3:f0c0:2567:7fe4:a00:27ff:fed1:d17a as fdf3:f0c0:2567:7fe4:ac5c:30ec:bfb7:ed89
Spoofed packet to fdf3:f0c0:2567:7fe4:ac5c:30ec:bfb7:ed89 as fdf3:f0c0:2567:7fe4:a00:27ff:fed1:d17a
Spoofed packet to fdf3:f0c0:2567:7fe4:a00:27ff:fed1:d17a as fdf3:f0c0:2567:7fe4:ac5c:30ec:bfb7:ed89
Spoofed packet to fe80::a511:624a:fcec:4377 as fe80::a00:27ff:fed1:d17a
Spoofed packet to fe80::a00:27ff:fed1:d17a as fe80::a511:624a:fcec:4377
Spoofed packet to fe80::a511:624a:fcec:4377 as fe80::a00:27ff:fed1:d17a
Spoofed packet to fe80::a00:27ff:fed1:d17a as fe80::a511:624a:fcec:4377
```

A MiTM Attack at the Local Link

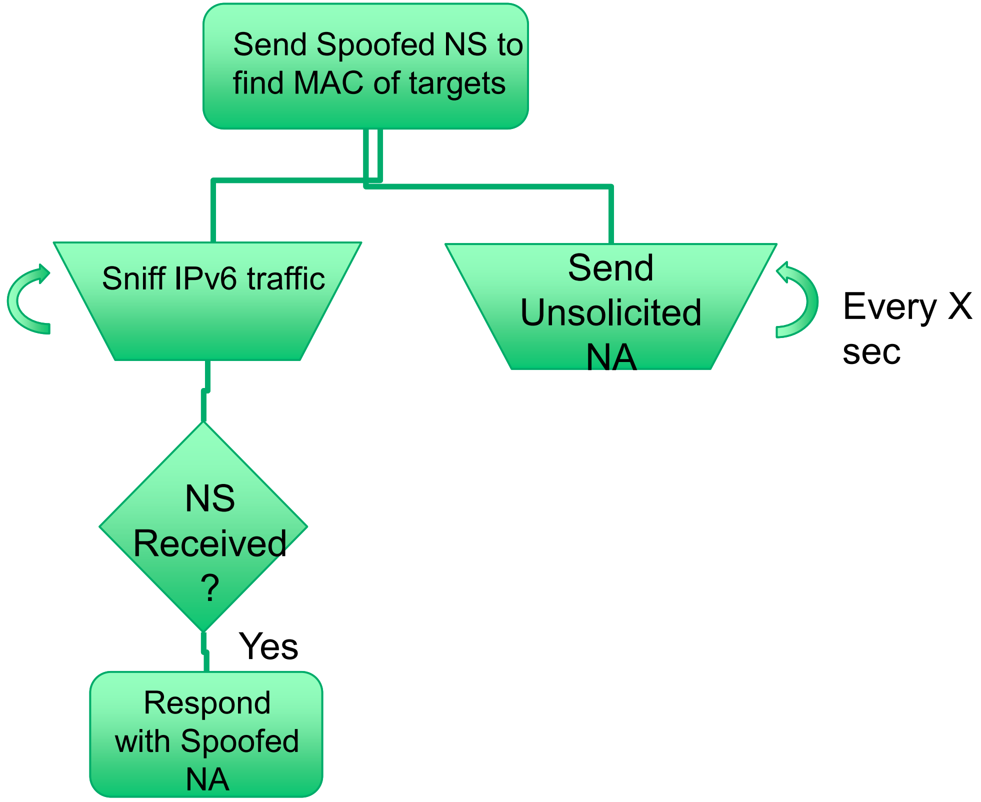


1. Send spoofed Neighbor Solicitations (NS) to find the MAC addresses of your target.

2. Respond to NS with spoofed Neighbor Advertisements (NA) with the "*Override* Flag" and the "*Solicited* Flag" set.

3. Send unsolicited NA with the "*Override* Flag" at regular time intervals (e.g. 2 to 5 sec).

A MiTM Attack at the Local Link



A MiTM Attack at the Local Link Using Scapy

- A selective (between two pairs) attack
  - Syntax: Usage mitm_attack.py <your_ipv6_address> <targets_comma_separated> <iface> <pcap_file_to_write_captured_traffic>
  - Use it as root:
  - Example:
    - ./mitm_attack.py fdf3:f0c0:2567:7fe4:800:27ff:fe00:0 fdf3:f0c0:2567:7fe4:a00:27ff:fe29:bfb0,fdf3:f0c0:2567:7fe4:2c9f:a8a1:7ac0:a8f1 vboxnet0 /tmp/mitm.pcap

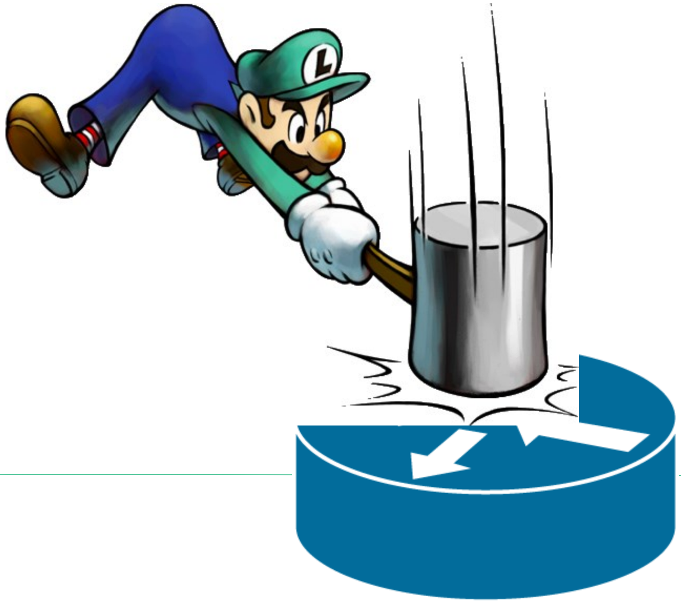
- Notes:
  - You must carefully choose the target's address (e.g. the private/temporary one for outgoing connections of the target).
  - It can also be a comma-separated list.

If You Need to Enable ipv6 forwarding

- Configure routing
- # echo 1 > /proc/sys/net/ipv6/conf/all/forwarding
- # sysctl -w net.ipv6.conf.all.forwarding=1

- To enable forwarding at boot, you'll need to edit /etc/sysctl.conf and add the following line.
- ## (If you will be using radvd, this step is unnecessary)
- net.ipv6.conf.default.forwarding=1

IPv6 Router Attacks at the Local Link



The Rogue Router Advertisement Problem Statement

- Router advertisements (as part of autoconfig approach) fundamental part of “IPv6 DNA”.
  - Modifying this behavior (e.g. by deactivating their processing on the host level) is a severe “deviation from default” and as such “operationally expensive”.
  - Such an approach might be hard to maintain through a system’s lifecycle as well.
    - Think service packs in MS world, kernel updates, installation of libs/tools/apps.
- By default, local link regarded trustworthy in IPv6 world (as we are all brothers on the local link) ;-)
  - All ND related stuff (which includes RAs) unauthenticated, by default.

Bad things that can happen

- Some RA-generating entity accidentally active in your network
  - IPv6 capable SOHO device connected by user.
  - Windows system with ICS enabled
    - No longer valid, see http://support.microsoft.com/kb/2750841/en-us.
  - Virtual machine running sth emitting RAs…

- Attacker interferes with router discovery
  - Denial-of-service by sending many bogus RAs
  - Traffic redirection by spoofed RAs

Get Router Info

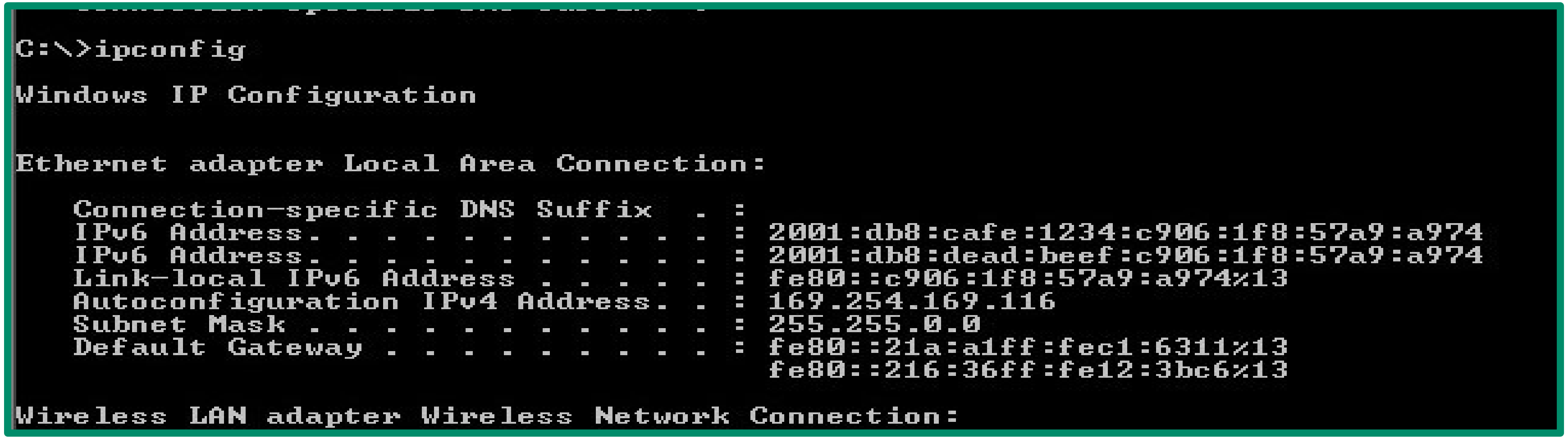
¬ [thc-ipv6-2.5]# ./*dump_router6* vboxnet0

```
Router: fe80::a00:27ff:fe74:ddaa (MAC: 08:00:27:74:dd:aa)
Priority: medium
Hop Count: 64
Lifetime: 300, Reachable: 0, Retrans: 0
Flags: NOTmanaged NOTother NOThome-agent NOTproxied
Options:
  Prefix: fdf3:f0c0:2567:7fe4::/64 (Valid: 86400, Preferred: 14400)
    Flags: On-Link Autoconfig RESERVED-BITS-SET-32
  MAC: 08:00:27:74:dd:aa
```

IPv6 Router Attacks

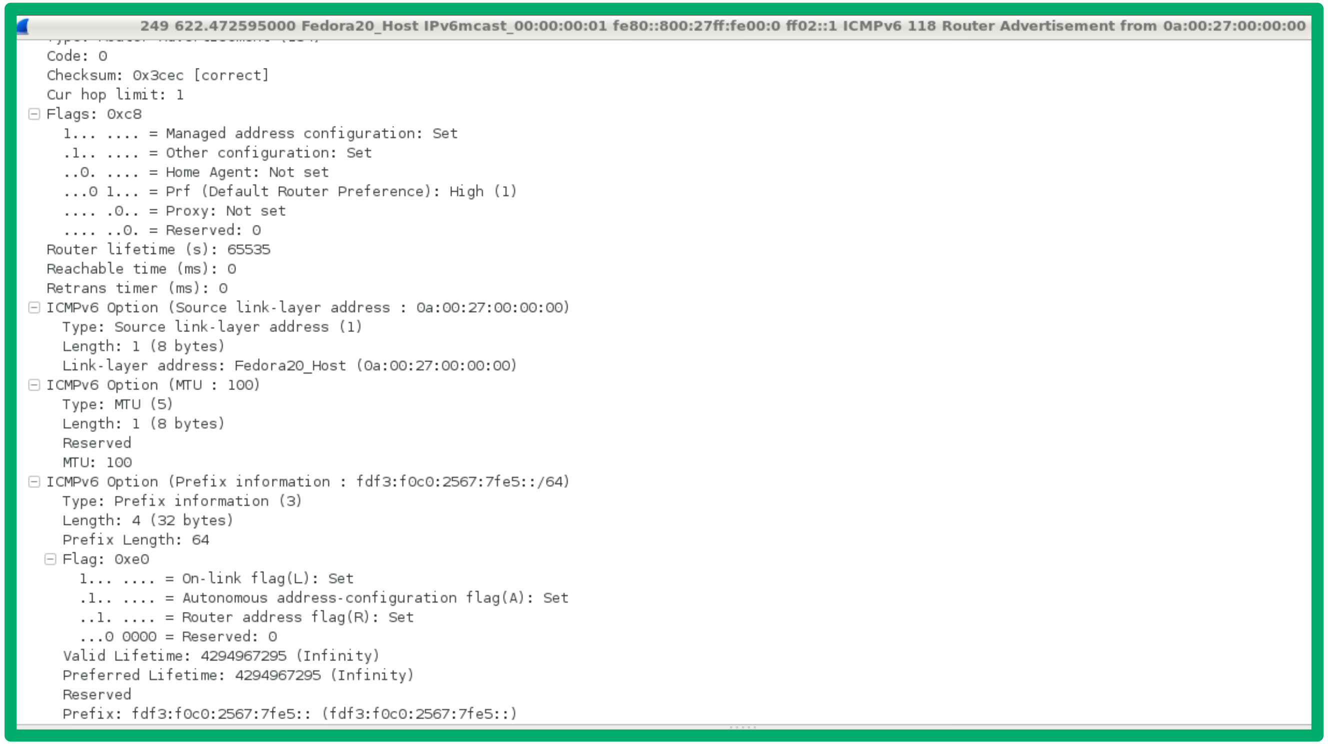
- Rogue RAs – periodic or in response to RS
  - Wrong gateway => DoS/MiTM
  - Router Lifetime = 0 => DoS – Can also help for MiTM
  - Router Priority => can help for DoS and MiTM
  - Set the L-bit for off-link prefixes => DoS
  - Provide invalid prefix for SLAAC => DoS
  - Wrong DHCP or DNS information => DoS/MitM (if the attacker sets up a bogus DHCPv6 server)
  - Small Current Hop Limit => Dos for large distances.
  - Empty default Router list (making the hosts believe that they are on-link); should have not been still effective.
- Router Redirection → DoS/MiTM
- Can be sent to multicast (all nodes) or unicast addresses (selective attack, more difficult to be detected).

fake_router26, Impact



```
C:\>ipconfig

Windows IP Configuration


Ethernet adapter Local Area Connection:

   Connection-specific DNS Suffix  . :
   IPv6 Address. . . . . . . . . . . : 2001:db8:cafe:1234:c906:1f8:57a9:a974
   IPv6 Address. . . . . . . . . . . : 2001:db8:dead:beef:c906:1f8:57a9:a974
   Link-local IPv6 Address . . . . . : fe80::c906:1f8:57a9:a974%13
   Autoconfiguration IPv4 Address. . : 169.254.169.116
   Subnet Mask . . . . . . . . . . . : 255.255.0.0
   Default Gateway . . . . . . . . . : fe80::21a:a1ff:fec1:6311%13
                                       fe80::216:36ff:fe12:3bc6%13

Wireless LAN adapter Wireless Network Connection:
```

Example of a Fake ICMPv6 RA

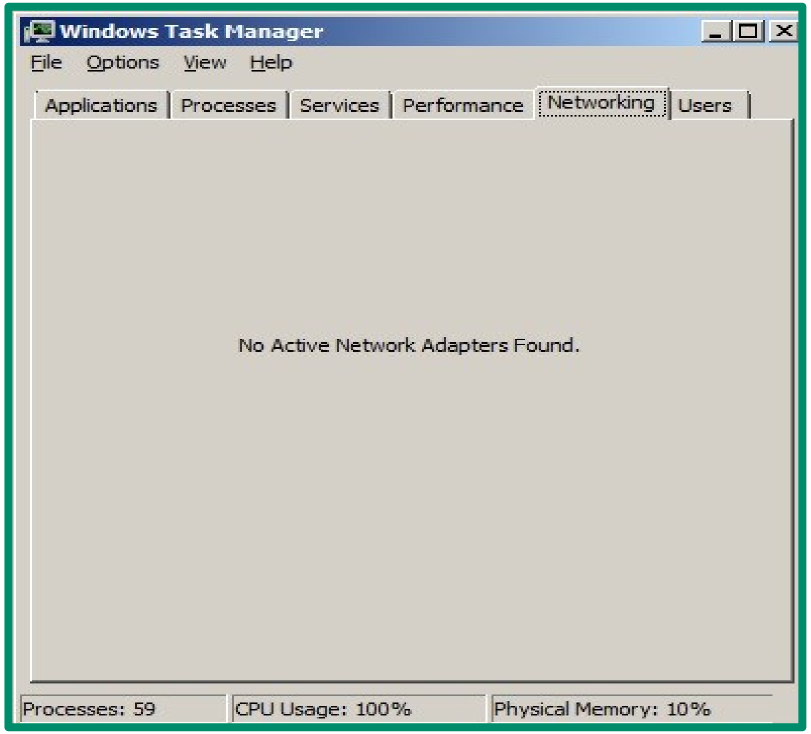


```
249 622.472595000 Fedora20_Host IPv6mcast_00:00:00:01 fe80::800:27ff:fe00:0 ff02::1 ICMPv6 118 Router Advertisement from 0a:00:27:00:00:00
  Code: 0
  Checksum: 0x3cec [correct]
  Cur hop limit: 1
⊟ Flags: 0xc8
    1... .... = Managed address configuration: Set
    .1.. .... = Other configuration: Set
    ..0. .... = Home Agent: Not set
    ...0 1... = Prf (Default Router Preference): High (1)
    .... .0.. = Proxy: Not set
    .... ..0. = Reserved: 0
  Router lifetime (s): 65535
  Reachable time (ms): 0
  Retrans timer (ms): 0
⊟ ICMPv6 Option (Source link-layer address : 0a:00:27:00:00:00)
    Type: Source link-layer address (1)
    Length: 1 (8 bytes)
    Link-layer address: Fedora20_Host (0a:00:27:00:00:00)
⊟ ICMPv6 Option (MTU : 100)
    Type: MTU (5)
    Length: 1 (8 bytes)
    Reserved
    MTU: 100
⊟ ICMPv6 Option (Prefix information : fdf3:f0c0:2567:7fe5::/64)
    Type: Prefix information (3)
    Length: 4 (32 bytes)
    Prefix Length: 64
  ⊟ Flag: 0xe0
      1... .... = On-link flag(L): Set
      .1.. .... = Autonomous address-configuration flag(A): Set
      ..1. .... = Router address flag(R): Set
      ...0 0000 = Reserved: 0
    Valid Lifetime: 4294967295 (Infinity)
    Preferred Lifetime: 4294967295 (Infinity)
    Reserved
    Prefix: fdf3:f0c0:2567:7fe5:: (fdf3:f0c0:2567:7fe5::)
```

Windows DoS by Randomising RA prefix

- CVE-2010-4669:
  - The Neighbor Discovery (ND) protocol implementation in the IPv6 stack in Microsoft Windows XP, Windows Server 2003, Windows Vista, Windows Server 2008, and Windows 7 allows remote attackers to cause a denial of service (CPU consumption and system hang) by sending many Router Advertisement (RA) messages with different source addresses

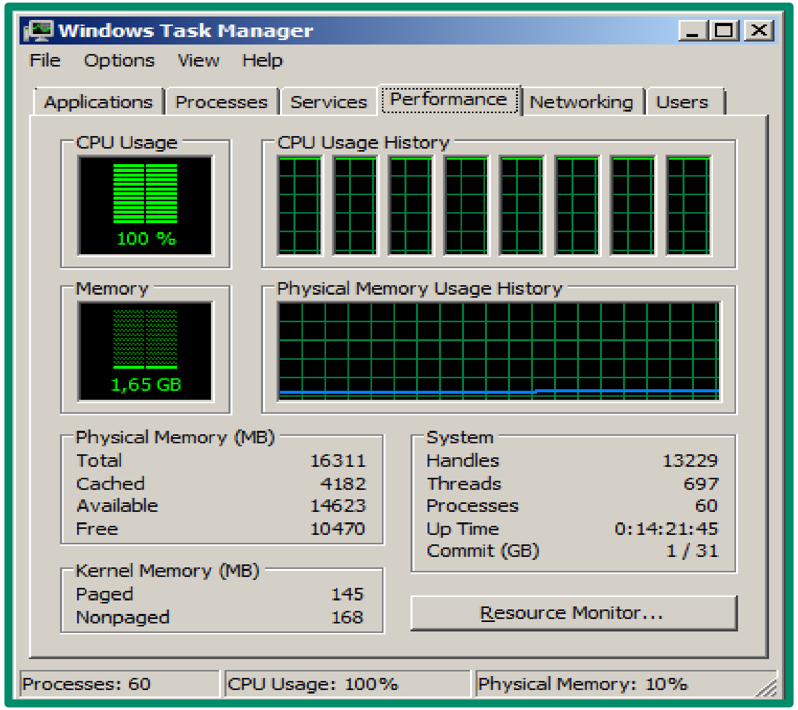
flood_router26, Impact


Windows Task Manager

File Options View Help
Applications Processes Services Performance Networking Users

No Active Network Adapters Found.

```
Processes: 59    CPU Usage: 100%    Physical Memory: 10%
```


Windows Task Manager

File Options View Help
Applications Processes Services Performance Networking Users
CPU Usage
100 %
CPU Usage History
Memory
1,65 GB
Physical Memory Usage History
Resource Monitor...

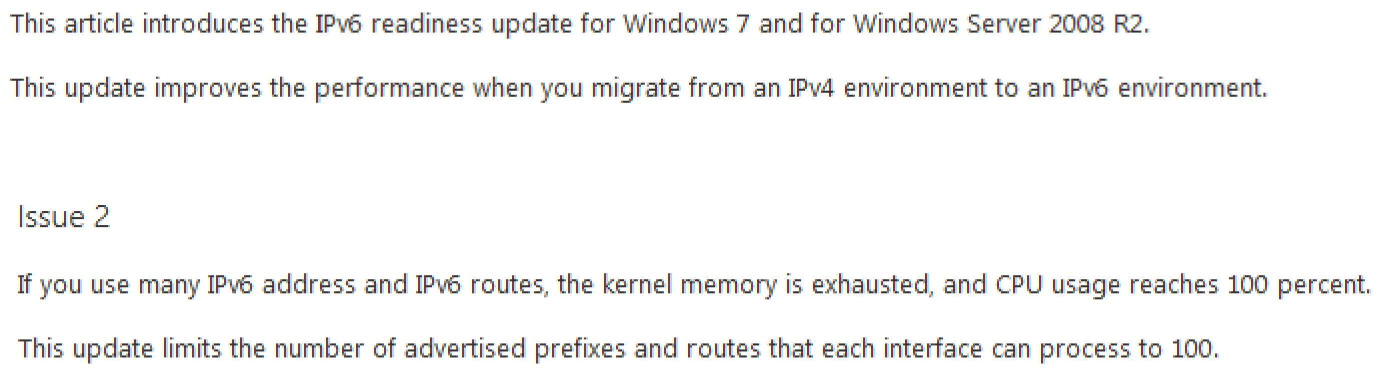
```
Physical Memory (MB)            System
Total          16311            Handles            13229
Cached          4182            Threads              697
Available      14623            Processes             60
Free           10470            Up Time       0:14:21:45
                                Commit (GB)       1 / 31
Kernel Memory (MB)
Paged            145
Nonpaged         168

Processes: 60    CPU Usage: 100%    Physical Memory: 10%
```

Some Results on *IPv6 Hackers* Mailing List

- New laptop (fast quad core i7) running Ubuntu 12.10
- it can push up to 120,000 RA packets/second on a Gigabit interface (a faster more powerful attacking device is far more effective)
- Typically crash a new Windows 8 laptop in 10-30 seconds
- Windows 7 is unusable while flood_router26 is running but quickly recovers after (with KB2750841)
- Windows Vista bogs down and then forever runs at 100% CPU until you reboot it. It's unusable during the flood and usually becomes partially usable sometime after it ends.

Microsoft IPv6 Readiness Update

http://support.microsoft.com/en-us/kb/2750841



This article introduces the IPv6 readiness update for Windows 7 and for Windows Server 2008 R2.

This update improves the performance when you migrate from an IPv4 environment to an IPv6 environment.

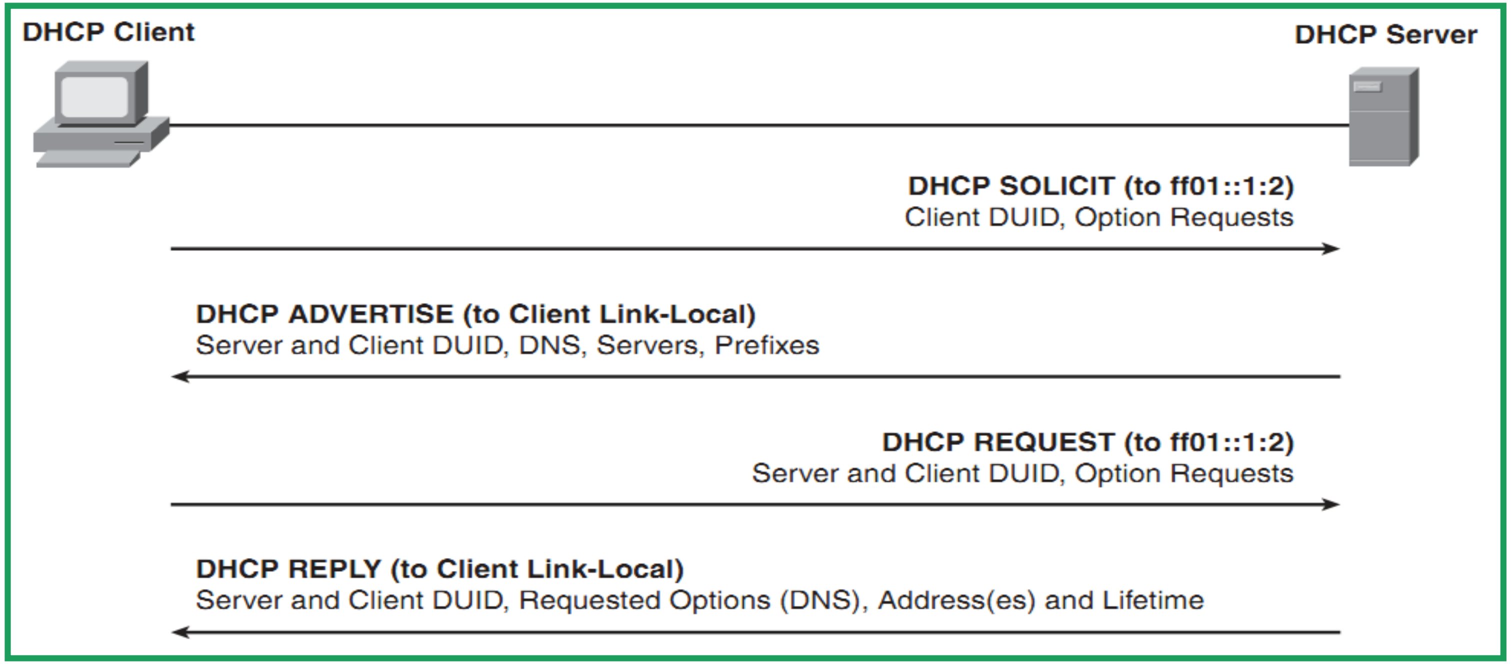
Issue 2

If you use many IPv6 address and IPv6 routes, the kernel memory is exhausted, and CPU usage reaches 100 percent.

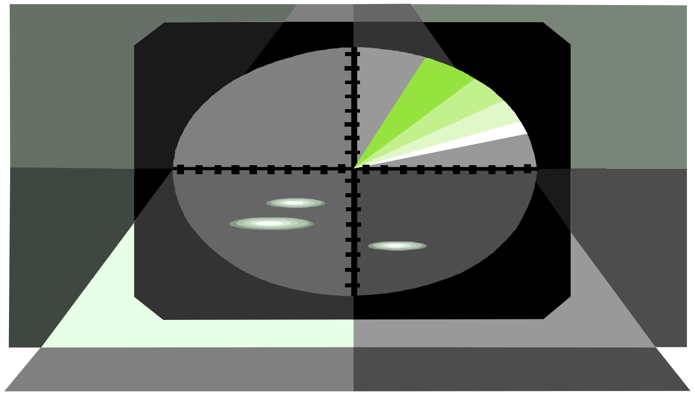
This update limits the number of advertised prefixes and routes that each interface can process to 100.

Attacks against DHCPv6

DHCP Message Exchange


DHCP Client
DHCP Server
DHCP SOLICIT (to ff01::1:2)
Client DUID, Option Requests
DHCP ADVERTISE (to Client Link-Local)
Server and Client DUID, DNS, Servers, Prefixes
DHCP REQUEST (to ff01::1:2)
Server and Client DUID, Option Requests
DHCP REPLY (to Client Link-Local)
Server and Client DUID, Requested Options (DNS), Address(es) and Lifetime

Threats against DHCPv6



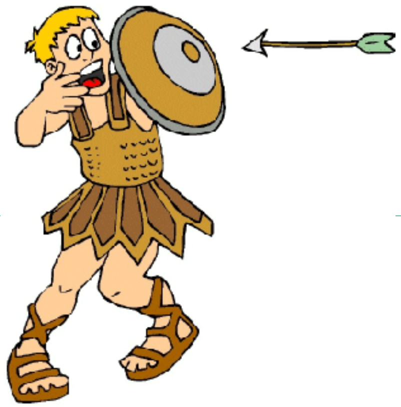
- Rogue DHCPv6 server
  - The Attacker sends malicious ADVERTISE and REPLY messages to legitimate clients. These messages contain falsified information about prefixes, DNS servers that could be used to redirect traffic.

1-Slide Sec Discussion

- As in v4 *rogue DHCP servers* can cause harm.
  - Nothing new here.
- Overall risk pretty much the same as in v4.
- Same mitigation techniques will apply.
  - In case DHCPv6 Guard is available for $YOUR_PLATFORM.

Defense Strategies

For Local Link Attacks



Problem Statement

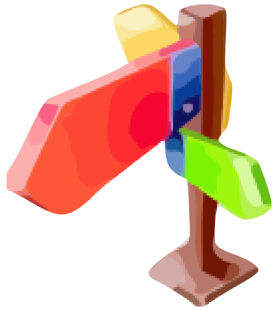
- Defending against those link local attacks is actually pretty hard
- As we are all brothers on the local link, we cannot rely on protocol properties to protect our IPv6 network
- Which is unfortunate and sad, but we have to deal with the situation

Suppress RA Processing on Hosts

- Operationally expensive & severe *deviation from default*.
- Note: just assigning a static IP address might not suffice.
  - E.g. MS Windows systems can still generate additional addresses/interface identifiers.
- Still we know and – somewhat – understand that most of you have a strong affinity to this approach
  - Human (and in particular: sysadmin) nature wants to *control* things…

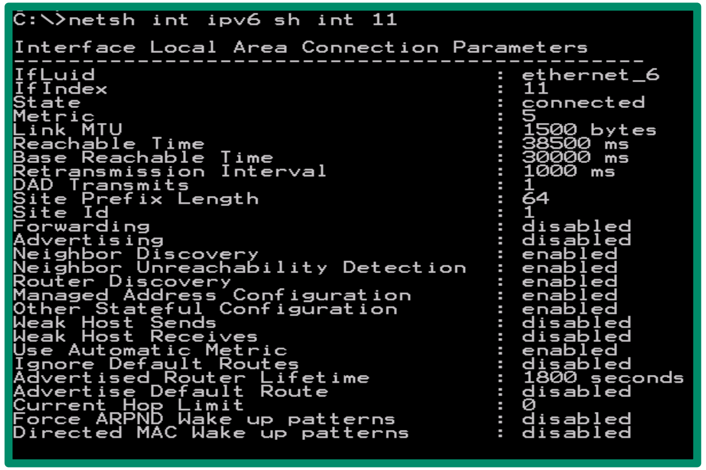
“Deviation from Default”

- By this term we designate any deviation from a default setting of any IT system which happens by means of some configuration step(s).
  - Change some parameter from “red” to “black” or 0 to 1 or …
- *Deviation from default* always requires OpEx.
  - In particular if to be maintained through affected systems’ lifecycle.
  - Even more so if affected system base is heterogeneous.
  - By its very nature, OpEx is limited. You knew that, right? ;-)
- *Deviation from default* doesn’t scale.
  - $IPV6_NETWORK might have 50 systems today. And tomorrow?
- *Deviation from default* adds complexity.
  - In particular if it’s “just some small modifications” combined…
    - Remember RFC 3439’s *Coupling Principle*?



Deactivation of RA processing on *Windows* Hosts (e.g. within DMZ)

- `netsh int ipv6 set int [index] routerdiscovery=disabled`



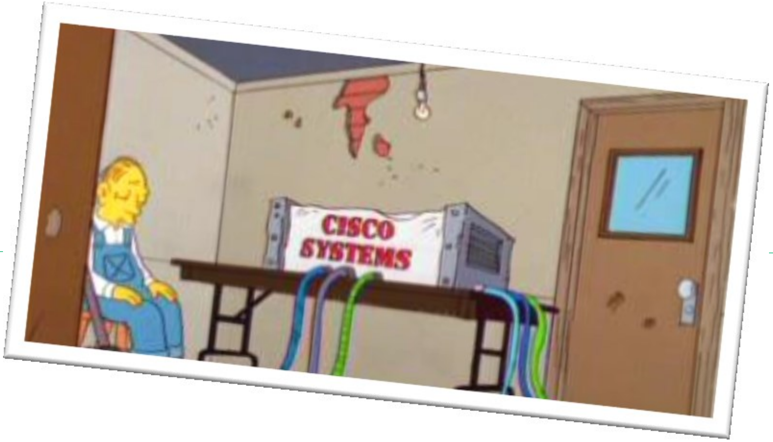
```
C:\>netsh int ipv6 sh int 11

Interface Local Area Connection Parameters
----------------------------------------------
IfLuid                              : ethernet_6
IfIndex                             : 11
State                               : connected
Metric                              : 5
Link MTU                            : 1500 bytes
Reachable Time                      : 38500 ms
Base Reachable Time                 : 30000 ms
Retransmission Interval             : 1000 ms
DAD Transmits                       : 1
Site Prefix Length                  : 64
Site Id                             : 1
Forwarding                          : disabled
Advertising                         : disabled
Neighbor Discovery                  : enabled
Neighbor Unreachability Detection   : enabled
Router Discovery                    : enabled
Managed Address Configuration       : enabled
Other Stateful Configuration        : enabled
Weak Host Sends                     : disabled
Weak Host Receives                  : disabled
Use Automatic Metric                : enabled
Ignore Default Routes               : disabled
Advertised Router Lifetime          : 1800 seconds
Advertise Default Route             : disabled
Current Hop Limit                   : 0
Force ARPND Wake up patterns        : disabled
Directed MAC Wake up patterns       : disabled
```

Linux: sysctl -w net.ipv6.conf.eth1.accept_ra=0

Overview for Different OS

- MS Windows
  - `netsh int ipv6 set int [index] routerdiscovery=disabled`
- FreeBSD
  - `sysctl net.inet6.ip6.accept_rtadv=0`
  - Do not run/invoke `rtsold`. (but the above prevents this anyway).
- Linux
  - Sth like: `echo 0 > /proc/sys/net/ipv6/conf/*/accept_ra`
  - See also IPv6 sect. of https://www.kernel.org/doc/Documentation/networking/ip-sysctl.txt

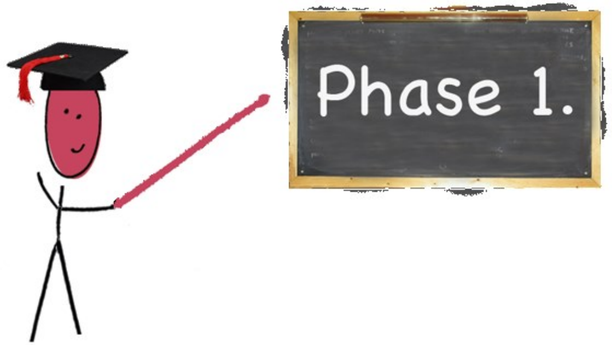
Cisco First-Hop-Security



Cisco First-Hop-Security

- Cisco name for various security features in IPv6
- Rollout is/was planned in three stages
- Every Phase will release/released more IPv6 security features to achieve feature parity with the IPv4 world

Phase I



- Available since Summer 2010
- Introduced RA Guard and Port based IPv6 ACLs
- In the beginning, only supported on datacenter switches
  - Since 15.0(2) supported on C2960S and C3560/3750-X

RA Guard

- Implements *isolation* principle similar to other L2 protection mechanisms already deployed in v4 world.
- RFC 6105
- Works quite well against some flavors of problem.
  - On most platforms no logging or port deactivation can be implemented. RA packets are just dropped.

RA Guard, Sample

```
Router(config-if)#ipv6 nd ?
  raguard  RA_Guard Configuration Command
Router(config-if)#ipv6 nd raguard ?
  <cr>
Router(config-if)#switchport mode access
Router(config-if)#ipv6 nd raguard
Router(config-if)#exit
Router(config)#exit

Router# show version
Cisco IOS Software, s3223_rp Software (s3223_rp-IPBASEK9-M), Version 12.2(33)SXI5, RELEASE SOFTWARE (fc2)
```

Phase II

- Available since end of 2011/ beginning of 2012 (depending on the platform)
- Introduced DHCPv6 Guard and NDP Snooping
  - The equivalent to DHCP Snooping and Dynamic ARP Inspection in the IPv4 World
- In the meantime good support on current access layer platforms

DHCPv6 Guard

- Similar functionality to DHCP Snooping in the IPv4 world
  - But more sophisticated
- Blocks reply and advertisement messages that originates from “malicious” DHCP servers and relay agents
- Provides finer level of granularity than DHCP Snooping.
- Messages can be filtered based on the address of the DHCP server or relay agent, and/or by the prefixes and address range in the reply message.

DHCPv6 Guard

```
Switch(config)#ipv6 access-list dhcpv6_server

Switch(config-ipv6-acl)#permit host FE80::1 any

Switch(config)#ipv6 prefix-list dhcpv6_prefix permit
2001:DB8:1::/64 le 128

Switch(config)#ipv6 dhcp guard policy dhcpv6guard_pol

Switch(config-dhcp-guard)#device-role server

Switch(config-dhcp-guard)#match server access-list
dhcpv6_server

Switch(config-dhcp-guard)#match reply prefix-list
dhcpv6_prefix

Switch(config)#vlan configuration 1

Switch(config-vlan-config)#ipv6 dhcp guard attach-policy
dhcpv6guard_pol
```

Security Binding Table

```
Switch#show ipv6 neighbors binding
Binding Table has 6 entries, 6 dynamic
Codes: L - Local, S - Static, ND - Neighbor Discovery, DH - DHCP, PKT - Other Packet, API - API created
Preflevel flags (prlvl):
0001:MAC and LLA match     0002:Orig trunk            0004:Orig access
0008:Orig trusted trunk    0010:Orig trusted access   0020:DHCP assigned
0040:Cga authenticated     0080:Cert authenticated    0100:Statically assigned IPv6

address                                   Link-Layer addr Interface vlan prlvl   age   state     Time left
ND  FE80::81E2:1562:E5A0:43EE             28D2.4448.E276  Gi1/15       1  0005    3mn REACHABLE  94 s
ND  FE80::3AEA:A7FF:FE85:C926             38EA.A785.C926  Gi1/2        1  0005   26mn STALE      86999 s
ND  FE80::10                              38EA.A785.C926  Gi1/2        1  0005   26mn STALE      85533 s
ND  FE80::1                               E4C7.228B.F180  Gi1/7        1  0005   35s  REACHABLE  272 s
DH  2001:DB8:1:0:BCC1:41C0:D904:E1B9      28D2.4448.E276  Gi1/15       1  0024    3mn REACHABLE  87 s
```

Syslog Message for dropped DHCPv6 packets:

```
%SISF-4-PAK_DROP: Message dropped A=FE80::1 G=2001:DB8:1:0:1146:8DF:1E2F:E079 V=1 I=Gi1/1 P=DHCPv6::ADV
Reason=Packet not authorized on port
```

Cisco IPv6 Snooping

- IPv6 Snooping is the basis for several FHS security mechanisms
- When configured on a target (VLAN, Interface etc.), it redirects NDP and DHCP traffic to the switch integrated security module

IPv6 ND Inspection

- Learns and secures bindings for addresses in layer 2 neighbor tables.
- Builds a trusted binding table database based on the IPv6 Snooping feature
- IPv6 ND messages that do not have valid bindings are dropped.
- A message is considered valid if the MAC-to-IPv6 address is verifiable

Example Output – Security Binding Table

```
switch#show ipv6 neighbors binding
Binding Table has 4 entries, 4 dynamic
Codes: L - Local, S - Static, ND - Neighbor Discovery, DH - DHCP, PKT - Other Packet, API - API created
IPv6 address                                Link-Layer addr Interface    vlan prlvl  age   state     Time left
ND  FE80::81E2:1562:E5A0:43EE                28D2.4448.E276  Gi1/15        1  0005    3mn REACHABLE  94 s
ND  FE80::3AEA:A7FF:FE85:C926                38EA.A785.C926  Gi1/2         1  0005   26mn STALE      86999 s
ND  FE80::10                                 38EA.A785.C926  Gi1/2         1  0005   26mn STALE      85533 s
ND  FE80::1                                  E4C7.228B.F180  Gi1/7         1  0005   35s  REACHABLE  272 s
```

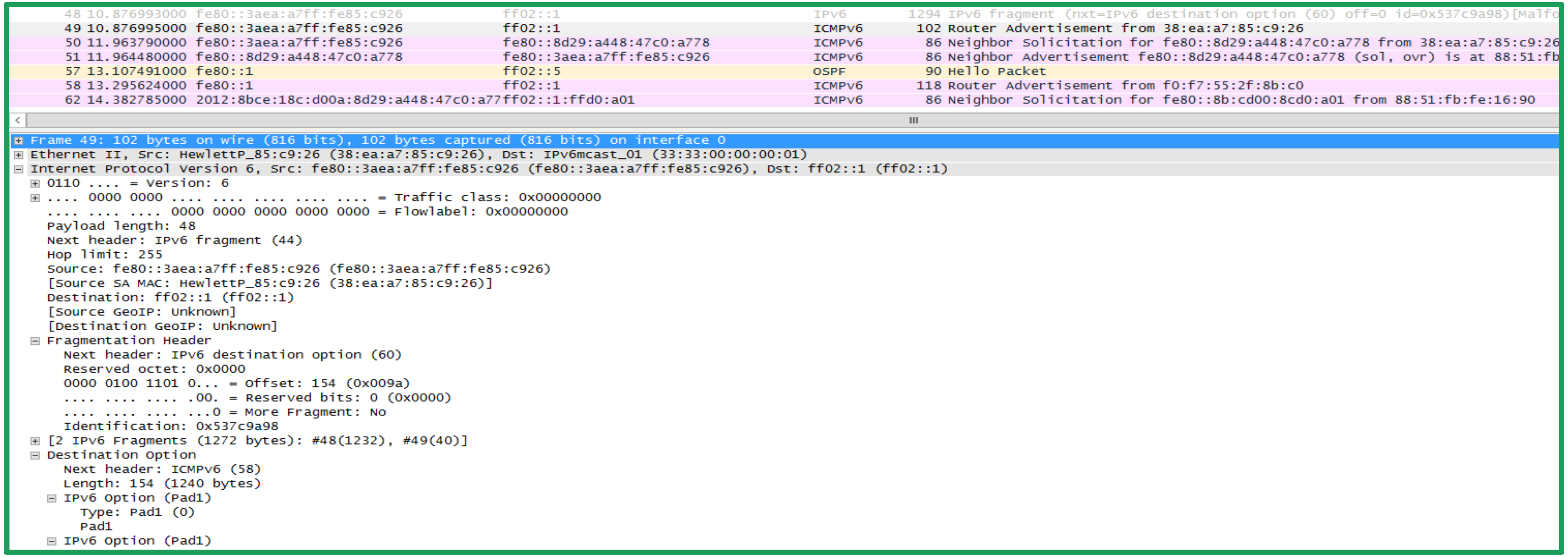
RA Guard Availability, Cisco

Feature/Platform	Catalyst 6500 Series	Catalyst 4500 Series	Catalyst 2K/3K Series	ASR1000 Router	7600 Router	Catalyst 3850	Wireless LAN Controller (Flex 7500, 5508, 2500, WISM-2)	Nexus 3k/5k/6k/7k
RA Guard	15.0(1)SY	15.1(2)SG	15.0.(2)SE		15.2(4)S	15.0(1)EX	7.2	NX-OS 7.2
IPv6 Snooping	15.0(1)SY[1]	15.1(2)SG	15.0.(2)SE	XE 3.9.0S	15.2(4)S	15.0(1)EX	7.2	NX-OS 7.2
DHCPv6 Guard	15.2(1)SY	15.1(2)SG	15.0.(2)SE		15.2(4)S	15.0(1)EX	7.2	NX-OS 7.2
Source/Prefix Guard	15.2(1)SY	15.2(1)E	15.0.(2)SE[2]	XE 3.9.0S	15.3(1)S		7.2	NX-OS 7.2
Destination Guard	15.2(1)SY	15.1(2)SG	15.2(1)E	XE 3.9.0S	15.2(4)S			NX-OS 7.2
RA Throttler	15.2(1)SY	15.2(1)E	15.2(1)E			15.0(1)EX	7.2	
ND Multicast Suppress	15.2(1)SY	15.1(2)SG	15.2(1)E	XE 3.9.0S		15.0(1)EX	7.2	

Sounds good? ;)

- Well, unfortunately all these features can be easily circumvented rendering them useless
- You maý ask yourself how?
- Using Extension Header to enforce fragmentation of ND packets

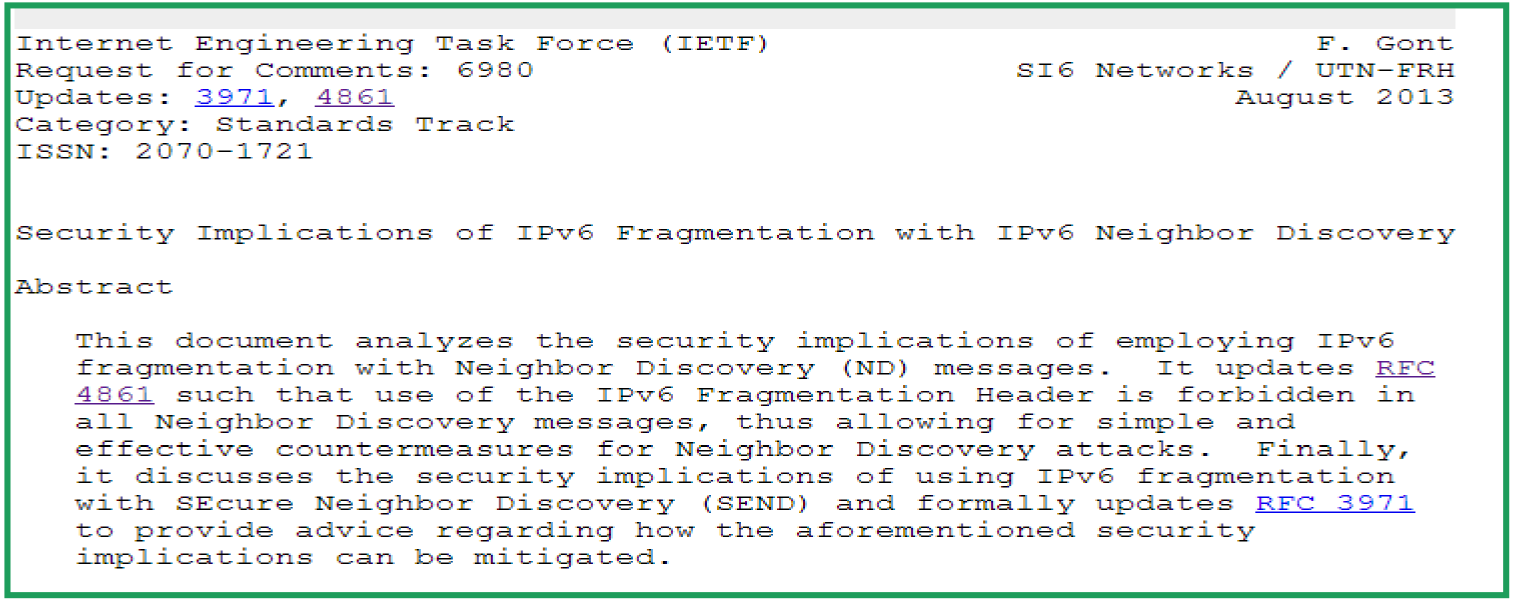
FHS Evasion



```
48 10.876993000 fe80::3aea:a7ff:fe85:c926    ff02::1                IPv6    1294 IPv6 fragment (nxt=IPv6 destination option (60) off=0 id=0x537c9a98)[Malfo
49 10.876995000 fe80::3aea:a7ff:fe85:c926    ff02::1                ICMPv6   102 Router Advertisement from 38:ea:a7:85:c9:26
50 11.963790000 fe80::3aea:a7ff:fe85:c926    fe80::8d29:a448:47c0:a778 ICMPv6  86 Neighbor Solicitation for fe80::8d29:a448:47c0:a778 from 38:ea:a7:85:c9:26
51 11.964480000 fe80::8d29:a448:47c0:a778    fe80::3aea:a7ff:fe85:c926 ICMPv6  86 Neighbor Advertisement fe80::8d29:a448:47c0:a778 (sol, ovr) is at 88:51:fb
57 13.107491000 fe80::1                      ff02::5                OSPF      90 Hello Packet
58 13.295624000 fe80::1                      ff02::1                ICMPv6   118 Router Advertisement from f0:f7:55:2f:8b:c0
62 14.382785000 2012:8bce:18c:d00a:8d29:a448:47c0:a77ff02::1:ffd0:a01 ICMPv6  86 Neighbor Solicitation for fe80::8b:cd00:8cd0:a01 from 88:51:fb:fe:16:90
```

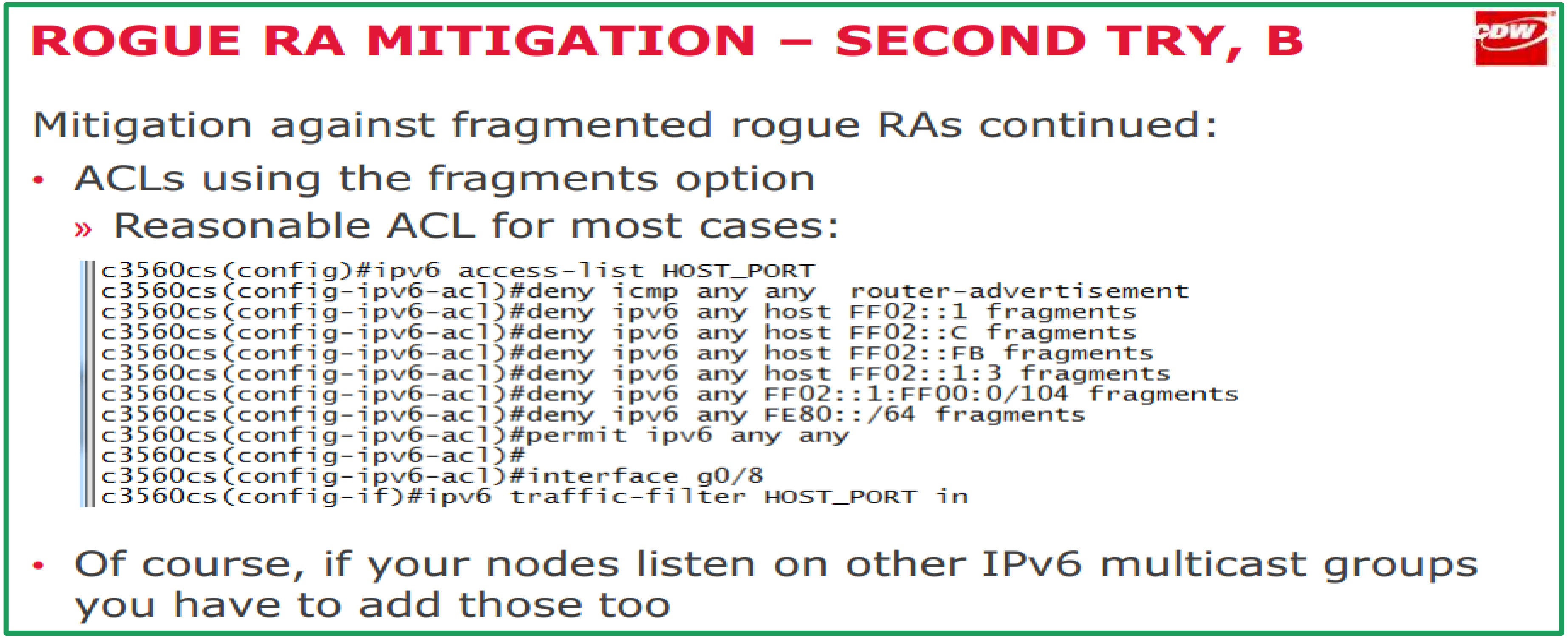
```
⊞ Frame 49: 102 bytes on wire (816 bits), 102 bytes captured (816 bits) on interface 0
⊞ Ethernet II, Src: HewlettP_85:c9:26 (38:ea:a7:85:c9:26), Dst: IPv6mcast_01 (33:33:00:00:00:01)
⊟ Internet Protocol Version 6, Src: fe80::3aea:a7ff:fe85:c926 (fe80::3aea:a7ff:fe85:c926), Dst: ff02::1 (ff02::1)
  ⊞ 0110 .... = Version: 6
  ⊞ .... 0000 0000 .... .... .... .... .... = Traffic class: 0x00000000
    .... .... .... 0000 0000 0000 0000 0000 = Flowlabel: 0x00000000
    Payload length: 48
    Next header: IPv6 fragment (44)
    Hop limit: 255
    Source: fe80::3aea:a7ff:fe85:c926 (fe80::3aea:a7ff:fe85:c926)
    [Source SA MAC: HewlettP_85:c9:26 (38:ea:a7:85:c9:26)]
    Destination: ff02::1 (ff02::1)
    [Source GeoIP: Unknown]
    [Destination GeoIP: Unknown]
  ⊟ Fragmentation Header
      Next header: IPv6 destination option (60)
      Reserved octet: 0x0000
      0000 0100 1101 0... = Offset: 154 (0x009a)
      .... .... .... .00. = Reserved bits: 0 (0x0000)
      .... .... .... ...0 = More Fragment: No
      Identification: 0x537c9a98
  ⊞ [2 IPv6 Fragments (1272 bytes): #48(1232), #49(40)]
  ⊟ Destination Option
      Next header: ICMPv6 (58)
      Length: 154 (1240 bytes)
    ⊟ IPv6 Option (Pad1)
        Type: Pad1 (0)
        Pad1
    ⊟ IPv6 Option (Pad1)
```

RFC 6980



```
Internet Engineering Task Force (IETF)                           F. Gont
Request for Comments: 6980                      SI6 Networks / UTN-FRH
Updates: 3971, 4861                                         August 2013
Category: Standards Track
ISSN: 2070-1721


Security Implications of IPv6 Fragmentation with IPv6 Neighbor Discovery

Abstract

   This document analyzes the security implications of employing IPv6
   fragmentation with Neighbor Discovery (ND) messages.  It updates RFC
   4861 such that use of the IPv6 Fragmentation Header is forbidden in
   all Neighbor Discovery messages, thus allowing for simple and
   effective countermeasures for Neighbor Discovery attacks.  Finally,
   it discusses the security implications of using IPv6 fragmentation
   with SEcure Neighbor Discovery (SEND) and formally updates RFC 3971
   to provide advice regarding how the aforementioned security
   implications can be mitigated.
```

ACLs



ROGUE RA MITIGATION – SECOND TRY, B

Mitigation against fragmented rogue RAs continued:

- ACLs using the fragments option
  - » Reasonable ACL for most cases:

```
c3560cs(config)#ipv6 access-list HOST_PORT
c3560cs(config-ipv6-acl)#deny icmp any any  router-advertisement
c3560cs(config-ipv6-acl)#deny ipv6 any host FF02::1 fragments
c3560cs(config-ipv6-acl)#deny ipv6 any host FF02::C fragments
c3560cs(config-ipv6-acl)#deny ipv6 any host FF02::FB fragments
c3560cs(config-ipv6-acl)#deny ipv6 any host FF02::1:3 fragments
c3560cs(config-ipv6-acl)#deny ipv6 any FF02::1:FF00:0/104 fragments
c3560cs(config-ipv6-acl)#deny ipv6 any FE80::/64 fragments
c3560cs(config-ipv6-acl)#permit ipv6 any any
c3560cs(config-ipv6-acl)#
c3560cs(config-ipv6-acl)#interface g0/8
c3560cs(config-if)#ipv6 traffic-filter HOST_PORT in
```

- Of course, if your nodes listen on other IPv6 multicast groups you have to add those too

Conclusion

- Different attack surface than in IPv4 and lots of old and new attacks
  - Because of different protocol behavior
- You are vulnerable to those kinds of attacks even if you do not use IPv6 in your corporate network
  - As the IPv6 stack is enabled by default on all modern operating systems.
- Defending against those link local attacks today is pretty hard
  - Due to potential hardware limitations of your access-layer switches
  - Paired with the easy circumvention of those FHS features
- We have to see how thinks develop in the future
- When you are a vendor or somebody who wants/must implement IPv6 stacks
  - Please do us all a favor and implement RFC 6980

IPv6 Perimeter Protection

Remote DoS Attacks

- Neighbor Cache Exhaustion
  - At the local-link.
  - They can also be launched remotely.
- There are more but not covered in this Workshop

Neighbor Cache Exhaustion

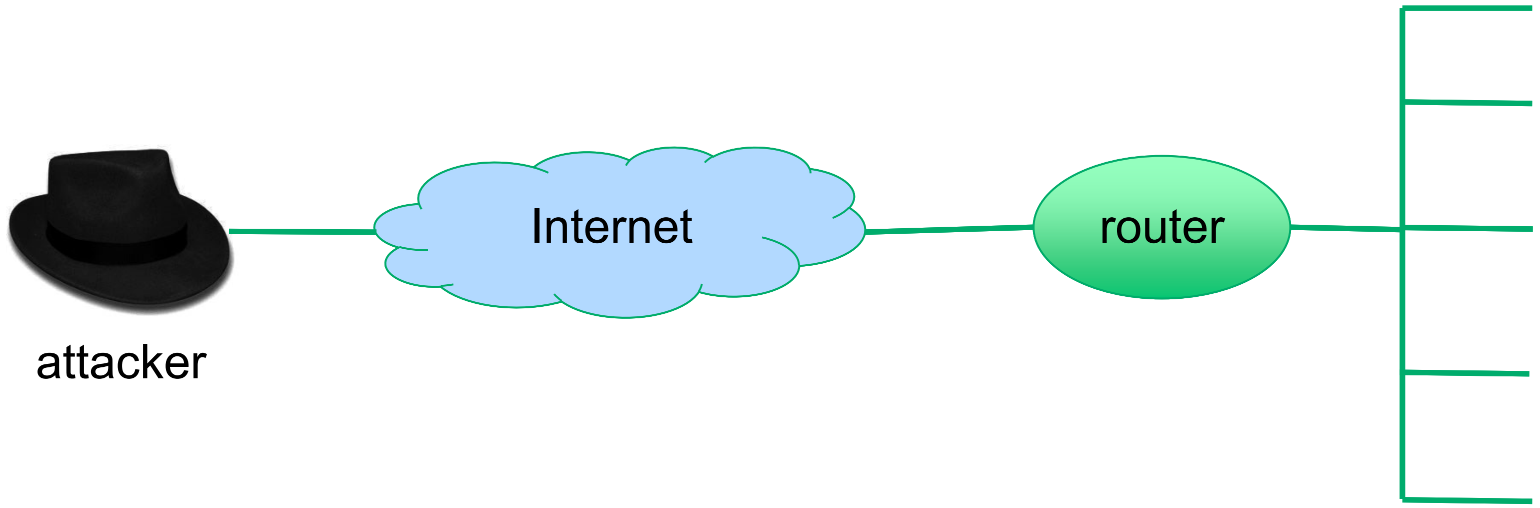
- First described in http://inconcepts.biz/~jsw/IPv6_NDP_Exhaustion.pdf - also discussed in RFC 6583

- Route cause: Huge default address space (/64) vs finite Neighbor Cache at devices.

- An attacker can simply launch a kind of scan at (part of) /64 subnet
  - Routers will attempt to perform address resolution for large numbers of unassigned addresses
  - Will fill-up the Neighbor Cache of the Router at target's side with INCOMPLETE states.

  → DoS for new or existing connections

Neighbor Cache Exhaustion - Example

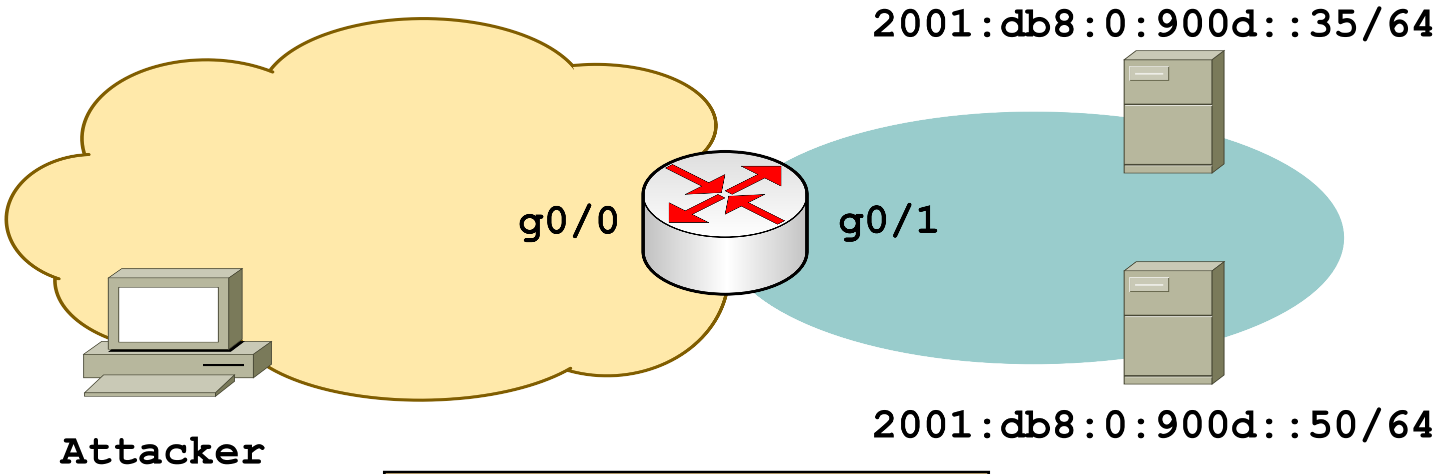


- Attack from outside (can be originated from inside, too).

Reproducing Neighbor Cache Exhaustion

- Launch a few *nmap* (-T 5) instances for /64.
- Use thc-ipv6 *ndpexhaust6* or, *ndpexhaust26* (more effective – floods the target /64 network with ICMPv6 TooBig error messages)

NCE, Some Notes from the Lab

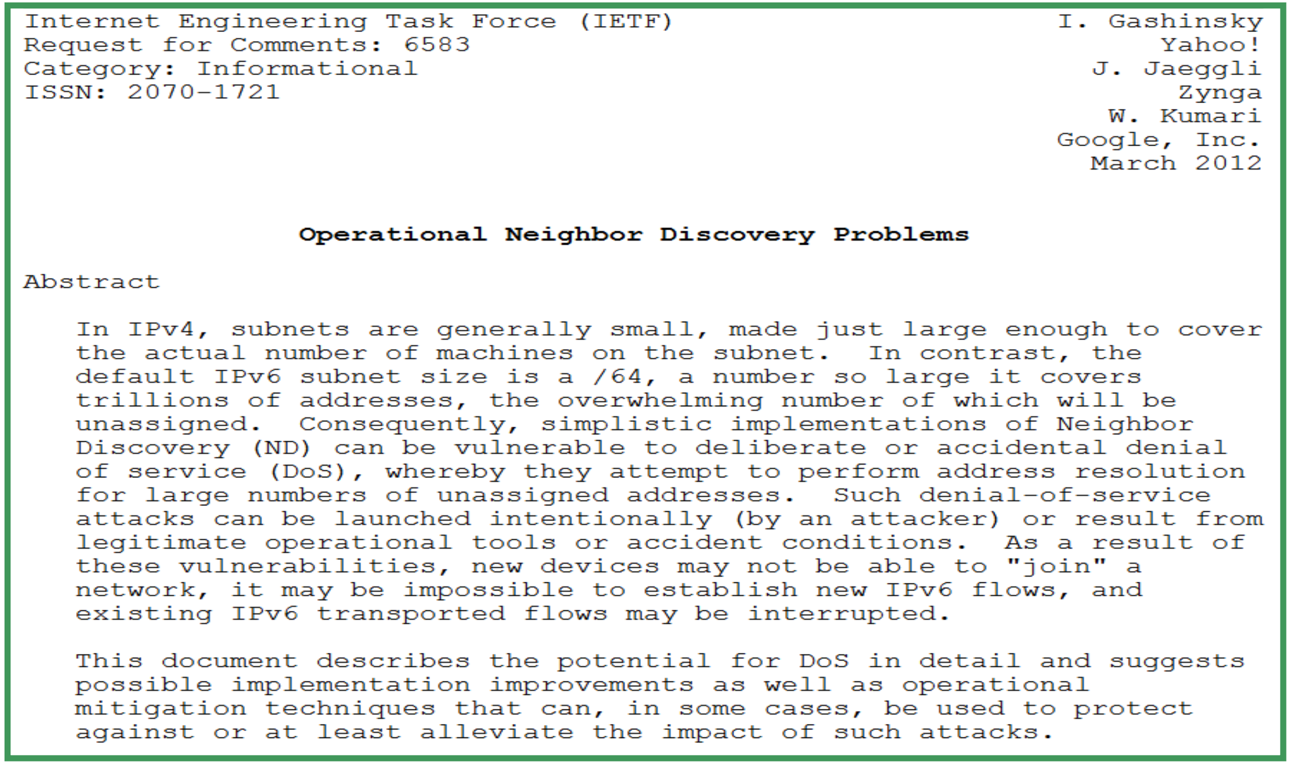


```
GigabitEthernet0/0
     FE80::BAAD:1
     2001:DB8:0:BAAD::1/64
GigabitEthernet0/1
     FE80::900D:1
     2001:DB8:0:900D::1/64
```

NCE, Conclusions from the Lab

- All tested Cisco devices do not store more than 512 INCOMPLETE entries in neighbor cache, at any given time.
  - Four different IOS-based medium-end devices tested.
- Furthermore reading RFC 4861 sect. 7.2.2 indicates INCMP entries will be deleted after three seconds anyway.
- So NCE *seems* not to be a major problem here (C land).
- Details of testing to be found here
  - http://www.insinuator.net/2013/03/ipv6-neighbor-cache-exhaustion-attacks-risk-assessment-mitigation-strategies-part-1/

RFC 6583



```
Internet Engineering Task Force (IETF)                      I. Gashinsky
Request for Comments: 6583                                        Yahoo!
Category: Informational                                        J. Jaeggli
ISSN: 2070-1721                                                    Zynga
                                                               W. Kumari
                                                            Google, Inc.
                                                              March 2012
```

Operational Neighbor Discovery Problems

Abstract

In IPv4, subnets are generally small, made just large enough to cover the actual number of machines on the subnet. In contrast, the default IPv6 subnet size is a /64, a number so large it covers trillions of addresses, the overwhelming number of which will be unassigned. Consequently, simplistic implementations of Neighbor Discovery (ND) can be vulnerable to deliberate or accidental denial of service (DoS), whereby they attempt to perform address resolution for large numbers of unassigned addresses. Such denial-of-service attacks can be launched intentionally (by an attacker) or result from legitimate operational tools or accident conditions. As a result of these vulnerabilities, new devices may not be able to "join" a network, it may be impossible to establish new IPv6 flows, and existing IPv6 transported flows may be interrupted.

This document describes the potential for DoS in detail and suggests possible implementation improvements as well as operational mitigation techniques that can, in some cases, be used to protect against or at least alleviate the impact of such attacks.

RFC 6583, Potential Controls

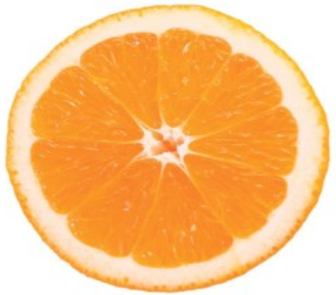
- Filtering of Unused Address Space
  - RFC 6583: “it is fully understood that this is ugly (and difficult to manage); but failing other options, it may be a useful technique especially when responding to an attack.“
- Obviously this requires static addressing.
- If you do this, use *stateless* filtering.
  - ACLs might be your friend.
  - Do *not* induce additional state by stateful filtering!
    - The more overall state maintained, the higher the overall vulnerability for DoS.

TROOPERS

ERNW providing security.

RFC 6583, Potential Controls

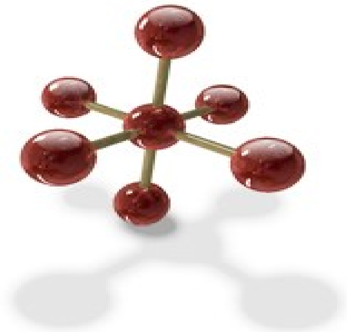
- *Minimal Subnet Sizing*
  - RFC 6583: “this approach is not suitable for use with hosts that are not statically configured.”
- Well, this violates the /64 paradigm.
  - Doesn’t RFC 6164 “allow” this violation anyway?
  - Still, this is about leaving “a standard path”. Be careful!
    - “Organization’s culture” may play a role here.
  - Yes, we are aware of sect. 3 of RFC 5375.
    - We don’t regard this as relevant here though.
  - See also: ID “Analysis of the 64-bit Boundary in IPv6 Addressing”
- Overall this approach might have quite good *operational feasibility*. Provided nothing breaks due to deviation f. /64.
- If you do this, still assign full /64, but configure /120 or sth.
  - So you can revert to /64 in case of problems or once better solutions are available (see below).



RFC 6583, Potential Controls

- Routing Mitigation
  - “For obvious reasons, host participation in the IGP makes many operators uncomfortable, but it can be a very powerful technique if used in a disciplined and controlled manner. One method to help address these concerns is to have the hosts participate in a different IGP (or difference instance of the same IGP) and carefully redistribute into the main IGP.”

- Honestly, this approach is so ridiculous both from an architecture and operations perspective, that we’ll not discuss this further.
  - Anybody remembers the days of `routed` on some Unix systems… and how happy we were to get rid of it?



RFC 6583, Potential Controls

- Tuning of the NDP Queue Rate Limit
  - "It is worth noting that this technique is worth investigating only if the device has separate queues for resolution of unknown addresses and the maintenance of existing entries."
- We expect this to become "the main approach"
  - Vendors already start to implement this. (see below)
- In Cisco land:
  - `ipv6 nd cache interface-limit`
    - See also http://www.cisco.com/en/US/docs/ios-xml/ios/ipv6/command/ipv6-i3.html#GUID-FC37F82B-5AAC-4298-BB6C-851FB7A06D88
    - This one provides some logging, too. Might come in handy for attack detection.
      - `Mar 10 15:11:51.719: %IPV6_ND-4-INTFLIMIT: Attempt to exceed interface limit on GigabitEthernet0/1 for 2001:DB8:0:900D::2:329A` (So use it in any case!)
  - on IOS-XE 2.6: `ipv6 nd resolution data limit`
    - Thanks to Jim Small for this hint.Might address another problem though.

Remote Attacks – Scanning

- If a system has a DNS record it will be found anyway.
  - Derive your own conclusions…
- See also
  - http://7bits.nl/blog/2012/03/26/finding-v6-hosts-by-efficiently-mapping-ip6-arpa
  - Full thread on *IPv6 hackers* mailing list: http://lists.si6networks.com/pipermail/ipv6hackers/2012-March/000526.html

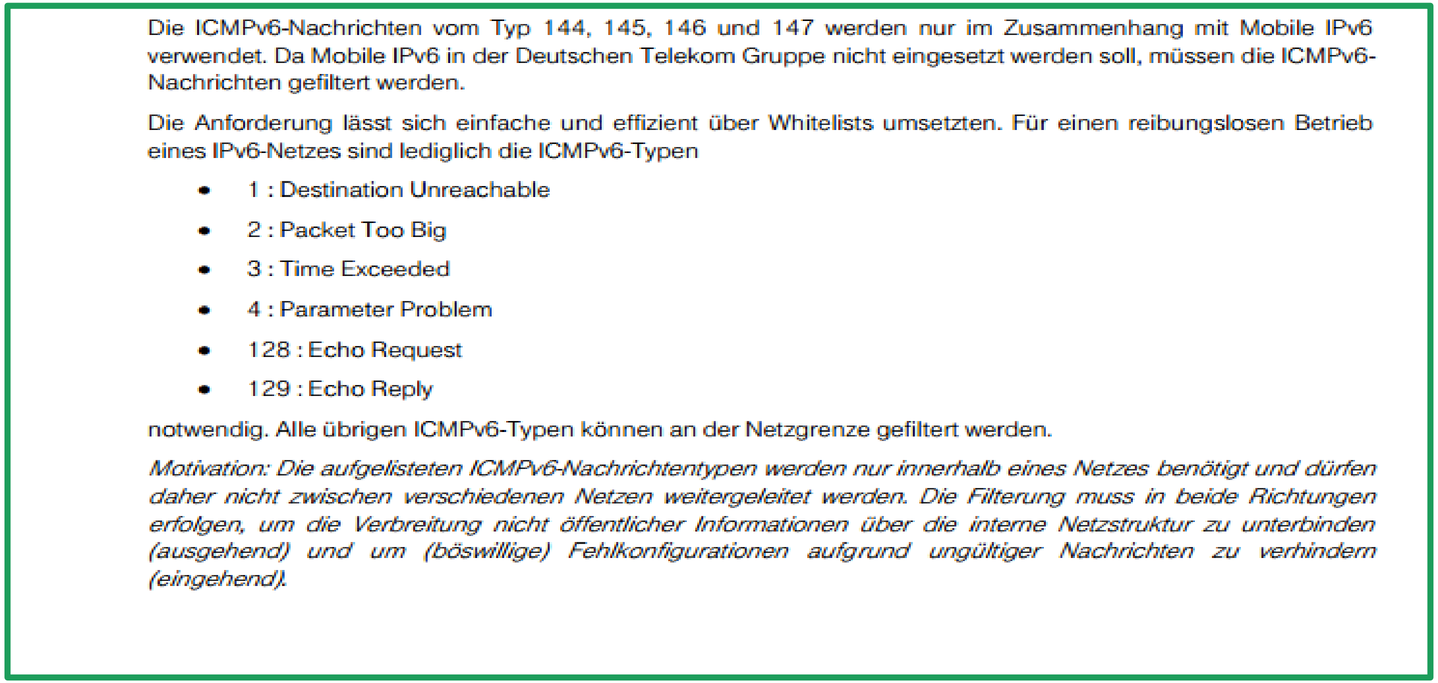
Traffic Filtering – Main Questions

- Does IPv6 require a different filtering paradigm?
  - ICMPv6
  - Extension Headers & Fragmentation
  - Bogons?
    - http://www.team-cymru.org/Services/Bogons/fullbogons-ipv6.txt
- Can we use the same tools?
  - Feature parity of security functions
  - Performance?

Filtering ICMPv6

- See RFC 4890.
- In one sentence:
  - You MUST NOT touch *packet-too-big* messages. Ever.

Filtering ICMP, Real-Life Approach
(again, link in appendix...)



Die ICMPv6-Nachrichten vom Typ 144, 145, 146 und 147 werden nur im Zusammenhang mit Mobile IPv6 verwendet. Da Mobile IPv6 in der Deutschen Telekom Gruppe nicht eingesetzt werden soll, müssen die ICMPv6-Nachrichten gefiltert werden.

Die Anforderung lässt sich einfache und effizient über Whitelists umsetzten. Für einen reibungslosen Betrieb eines IPv6-Netzes sind lediglich die ICMPv6-Typen

- 1 : Destination Unreachable
- 2 : Packet Too Big
- 3 : Time Exceeded
- 4 : Parameter Problem
- 128 : Echo Request
- 129 : Echo Reply

notwendig. Alle übrigen ICMPv6-Typen können an der Netzgrenze gefiltert werden.

Motivation: Die aufgelisteten ICMPv6-Nachrichtentypen werden nur innerhalb eines Netzes benötigt und dürfen daher nicht zwischen verschiedenen Netzen weitergeleitet werden. Die Filterung muss in beide Richtungen erfolgen, um die Verbreitung nicht öffentlicher Informationen über die interne Netzstruktur zu unterbinden (ausgehend) und um (böswillige) Fehlkonfigurationen aufgrund ungültiger Nachrichten zu verhindern (eingehend).

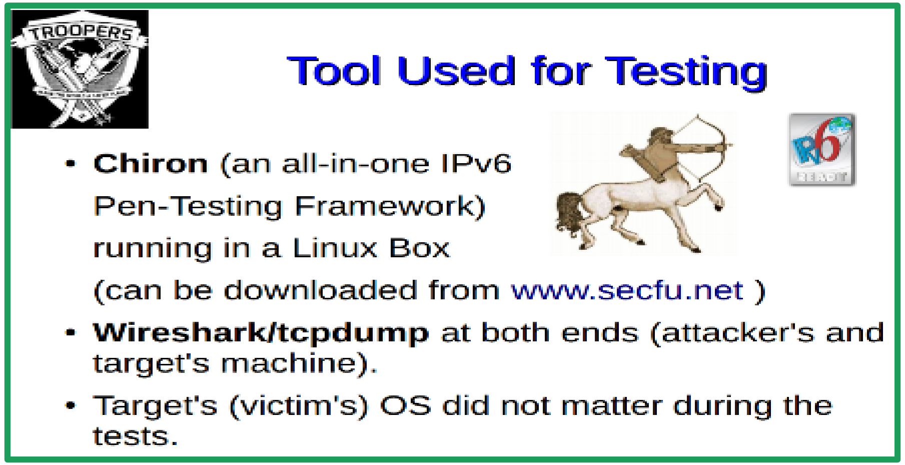
Extensions Headers & Fragmentation

The Mess

Researching IPv6 Security Capabilities (RISC)

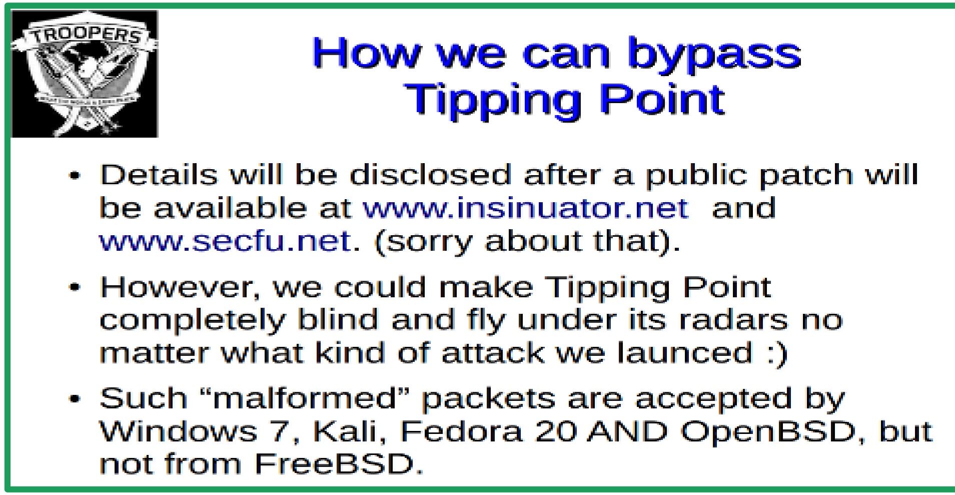
https://www.troopers.de/wp-content/uploads/2014/01/TROOPERS14-Overview_of_the_Real-World_Capabilities_of_Major_Commercial_Security_Products-Christopher_Werny+Antonios_Atlasis-Part2_2.pdf

The Mess (II)



Tool Used for Testing

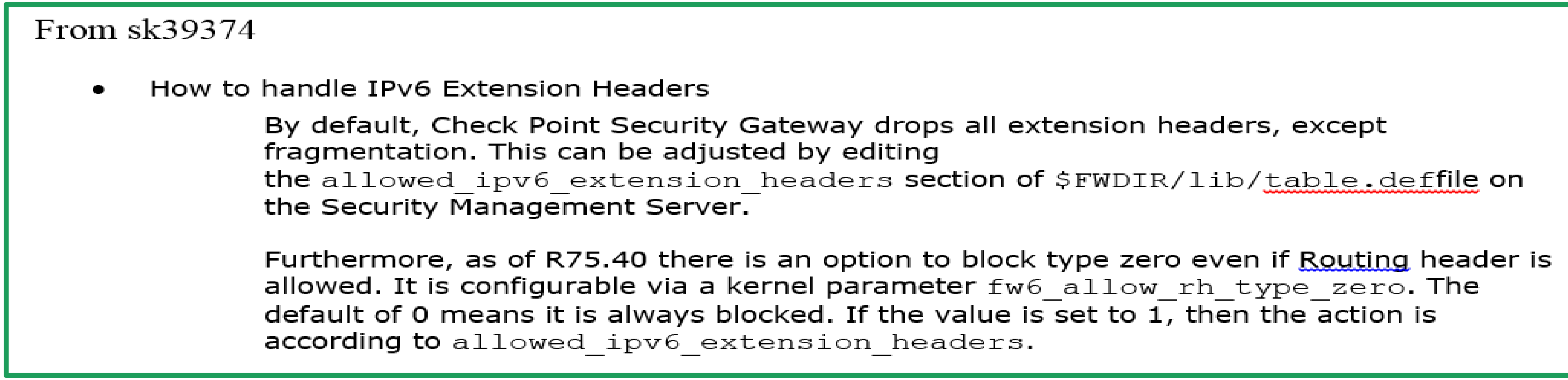
- **Chiron** (an all-in-one IPv6 Pen-Testing Framework) running in a Linux Box (can be downloaded from www.secfu.net)
- **Wireshark/tcpdump** at both ends (attacker's and target's machine).
- Target's (victim's) OS did not matter during the tests.



How we can bypass Tipping Point

- Details will be disclosed after a public patch will be available at www.insinuator.net and www.secfu.net. (sorry about that).
- However, we could make Tipping Point completely blind and fly under its radars no matter what kind of attack we launced :)
- Such “malformed” packets are accepted by Windows 7, Kali, Fedora 20 AND OpenBSD, but not from FreeBSD.

Filtering Fragments / EHs
Can it be done/configured? (Sample)



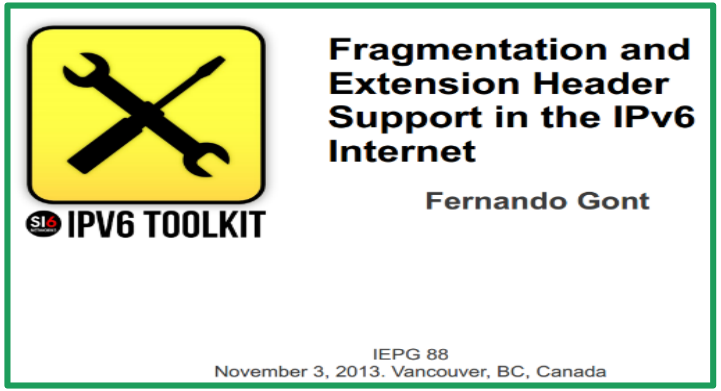
From sk39374

- How to handle IPv6 Extension Headers

  By default, Check Point Security Gateway drops all extension headers, except fragmentation. This can be adjusted by editing the `allowed_ipv6_extension_headers` section of `$FWDIR/lib/table.def`file on the Security Management Server.

  Furthermore, as of R75.40 there is an option to block type zero even if Routing header is allowed. It is configurable via a kernel parameter `fw6_allow_rh_type_zero`. The default of 0 means it is always blocked. If the value is set to 1, then the action is according to `allowed_ipv6_extension_headers`.

So in the End of the Day


IPV6 TOOLKIT

Fragmentation and Extension Header Support in the IPv6 Internet

Fernando Gont

IEPG 88
November 3, 2013. Vancouver, BC, Canada

- Keep it simple: just block/drop all this crap.
- The others do it the same way

Feature Parity in March 2016

- Firewalls: we‘re mostly there.
  - Performance can be an issue → ask $VENDOR (or test lab, industry peers or guys like Johannes, Marc, us) for data points.
- IDPS: a huge mess.
  - See, in particular, our presentation at Black Hat US 2014.
- Others (WAF, SIEM, content filtering): depends.
  - Some can, some have deficiences.

- IPv6 security awareness.
  - Read the RFCs
  - Build your lab
  - Test and play with it

- You will have to to do it, sooner or later, anyway…

- So get IPv6 Ready! ☺

Thank you very much!

Go make the world a safer place!

Questions?

u can reach us at:

:werny@ernw.de, www.insinuator.net

schaefer@ernw.de

Speak with us during the next ys. ☺

Appendix: Tools

- scapy6 [http://namabiiru.hongo.wide.ad.jp/scapy6/]
- ip6sic [http://ip6sic.sourceforge.net/]
- THC IPv6 [http://freeworld.thc.org/thc-ipv6/]
- ERNW fuzzing toolkit
  - http://www.insinuator.net/2011/05/update-for-your-fuzzing-toolkit/
- LOKI
  - http://www.insinuator.net/2010/08/try-loki/

Links

- IETF Draft Operational Security Considerations:
  - http://tools.ietf.org/html/draft-ietf-opsec-v6-01
- Design Guidelines for IPv6 Networks
  - http://tools.ietf.org/html/draft-matthews-v6ops-design-guidelines-01
- Enterprise IPv6 Deployment Guidelines
  - http://tools.ietf.org/html/draft-ietf-v6ops-enterprise-incremental-ipv6-01
- DC Migration to IPv6
  - http://tools.ietf.org/html/draft-lopez-v6ops-dc-ipv6-02
- Sicherheitsanforderungen DTAG
  - http://www.telekom.com/static/-/155996/4/technische-sicherheitsanforderungen-si
  - http://www.telekom.com/verantwortung/sicherheit/155994

Links, Filtering

- ICMP Filtering
  - http://tools.ietf.org/html/draft-ietf-opsec-icmp-filtering-03
- Cisco FHS Wiki
  - http://docwiki.cisco.com/wiki/FHS
- Sample ASA config
  - http://www.cluebyfour.org/ipv6/
- Eldad Zack‘s presentation at Berlin IPv6 Hackers meeting
  - https://a13725d0-a-62cb3a1a-s-sites.googlegroups.com/site/ipv6hackers/meetings/ipv6-hackers-1/zack-ipv6hackers1-firewall-security-assessment-and-benchmarking.pdf