



# MLD Considered Harmful

Breaking Another IPv6 Subprotocol

Antonios Atlasis, [aatlasis@secfu.net](mailto:aatlasis@secfu.net)

Enno Rey, [erey@ernw.de](mailto:erey@ernw.de)

Jayson Salazar, [jsalazar@ernw.de](mailto:jsalazar@ernw.de)





## Who We Are



### – Antonios

- IT security enthusiast
- Author of *Chiron*



### – Enno

- Old-school networking guy



### – Jayson

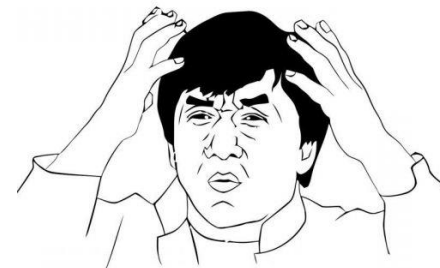
- Security researcher at ERNW



Research inside.™

## Agenda

- The Object of Interest
- How We Tackled It
- What We Observed
- What All This Means

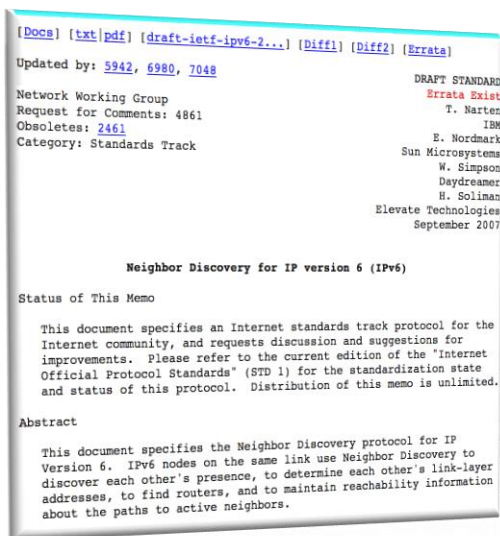




No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	Windows7.1-linklocal	ff02::16	ICMPv6	90	Multicast Listener Report Message v2
2	0.000013	Windows7.1-linklocal	ff02::16	ICMPv6	90	Multicast Listener Report Message v2
3	0.008497	Windows7.1-linklocal	ff02::16	ICMPv6	90	Multicast Listener Report Message v2
4	0.008506	Windows7.1-linklocal	ff02::16	ICMPv6	90	Multicast Listener Report Message v2
5	0.023971	Windows7.1-linklocal	ff02::16	ICMPv6	90	Multicast Listener Report Message v2
6	0.023984	Windows7.1-linklocal	ff02::16	ICMPv6	90	Multicast Listener Report Message v2
7	0.025772	Windows7.1-linklocal	ff02::16	ICMPv6	90	Multicast Listener Report Message v2
8	0.025777	Windows7.1-linklocal	ff02::16	ICMPv6	90	Multicast Listener Report Message v2
9	0.261958	Windows7.1-linklocal	ff02::16	ICMPv6	90	Multicast Listener Report Message v2
10	0.261967	Windows7.1-linklocal	ff02::16	ICMPv6	90	Multicast Listener Report Message v2
11	600.048733	Windows7.1-linklocal	ff02::16	ICMPv6	90	Multicast Listener Report Message v2
12	600.048746	Windows7.1-linklocal	ff02::16	ICMPv6	90	Multicast Listener Report Message v2
13	600.063445	Windows7.1-linklocal	ff02::16	ICMPv6	90	Multicast Listener Report Message v2
14	600.063458	Windows7.1-linklocal	ff02::16	ICMPv6	90	Multicast Listener Report Message v2
15	600.075012	Windows7.1-linklocal	ff02::16	ICMPv6	90	Multicast Listener Report Message v2
16	600.075020	Windows7.1-linklocal	ff02::16	ICMPv6	90	Multicast Listener Report Message v2
17	600.077356	Windows7.1-linklocal	ff02::16	ICMPv6	90	Multicast Listener Report Message v2
18	600.077366	Windows7.1-linklocal	ff02::16	ICMPv6	90	Multicast Listener Report Message v2
19	600.264367	Windows7.1-linklocal	ff02::16	ICMPv6	90	Multicast Listener Report Message v2
20	600.264378	Windows7.1-linklocal	ff02::16	ICMPv6	90	Multicast Listener Report Message v2
21	1199.407524	Windows7.1-linklocal	ff02::16	ICMPv6	90	Multicast Listener Report Message v2
22	1199.407537	Windows7.1-linklocal	ff02::16	ICMPv6	90	Multicast Listener Report Message v2
23	1199.423790	Windows7.1-linklocal	ff02::16	ICMPv6	90	Multicast Listener Report Message v2
24	1199.423802	Windows7.1-linklocal	ff02::16	ICMPv6	90	Multicast Listener Report Message v2
25	1199.428513	Windows7.1-linklocal	ff02::16	ICMPv6	90	Multicast Listener Report Message v2

## Why This Talk (I)

## Why This Talk (II)



RFC 4861 Neighbor Discovery for IP version 6 (IPv6), sect. 7.2.1

Descriptive or prescriptive (“normative”)??



- “Joining the solicited-node multicast address **is done** using a Multicast Listener Discovery such as [MLD] or [MLDv2] protocols.”



RFC 6434

IPv6 Node Requirements

December 2011

### 5.10. Multicast Listener Discovery (MLD) for IPv6

Nodes that need to join multicast groups MUST support MLDv1 [RFC2710]. MLDv1 is needed by any node that is expected to receive and process multicast traffic. Note that Neighbor Discovery (as used on most link types -- see Section 5.2) depends on multicast and requires that nodes join Solicited Node multicast addresses.

MLDv2 [RFC3810] extends the functionality of MLDv1 by supporting Source-Specific Multicast. The original MLDv2 protocol [RFC3810] supporting Source-Specific Multicast [RFC4607] supports two types of "filter modes". Using an INCLUDE filter, a node indicates a multicast group along with a list of senders for the group from which it wishes to receive traffic. Using an EXCLUDE filter, a node indicates a multicast group along with a list of senders from which it wishes to exclude receiving traffic. In practice, operations to block source(s) using EXCLUDE mode are rarely used but add considerable implementation complexity to MLDv2. Lightweight MLDv2 [RFC5790] is a simplified subset of the original MLDv2 specification that omits EXCLUDE filter mode to specify undesired source(s).

## Complexity

Here's another gem for you: MLD

## Why This Talk (III)

From:

[https://www.troopers.de/wp-content/uploads/2013/11/TROOPERS14-Why\\_IPv6\\_Security\\_is\\_so\\_hard-Structural\\_Deficits\\_of\\_IPv6\\_and\\_their\\_Implications-Enno\\_Rey.pdf](https://www.troopers.de/wp-content/uploads/2013/11/TROOPERS14-Why_IPv6_Security_is_so_hard-Structural_Deficits_of_IPv6_and_their_Implications-Enno_Rey.pdf)

3/17/14

3/1/14

#36 | www.ernw.de

#36 | www.ernw.de

## So here's a Protocol...



- Apparently every IPv6 stack
  - has to support.
  - might have enabled by default (most do).
- It's not really clear if it is always needed or not.
- It's a complex beast (as we will see).
- Not much public sec research so far.
  - We'll close this gap today ;-)





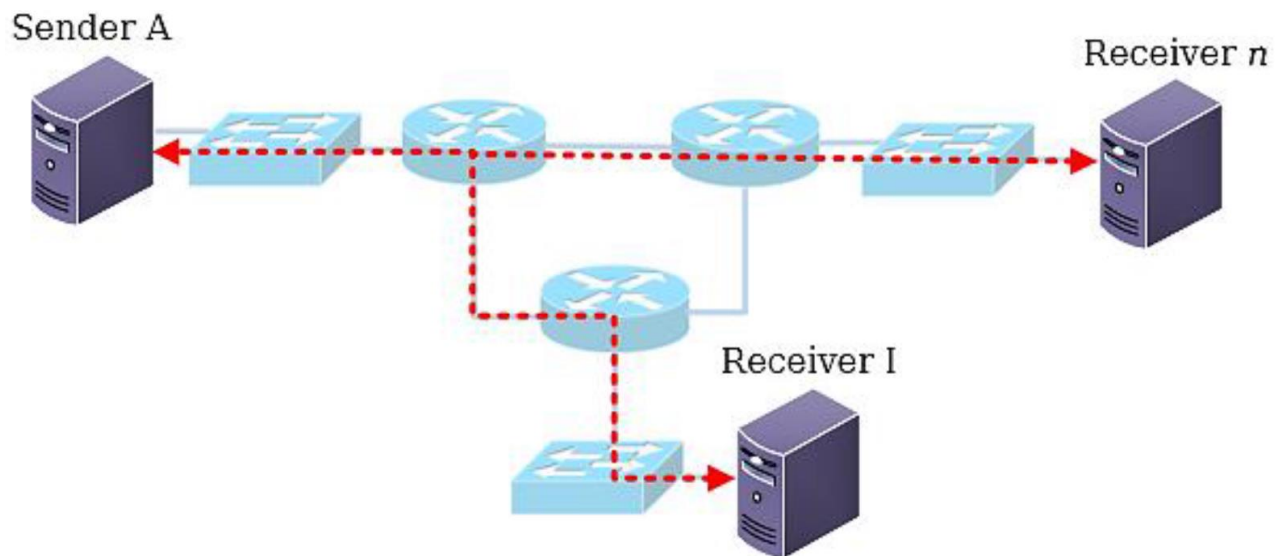
# MLD Fundamentals

---





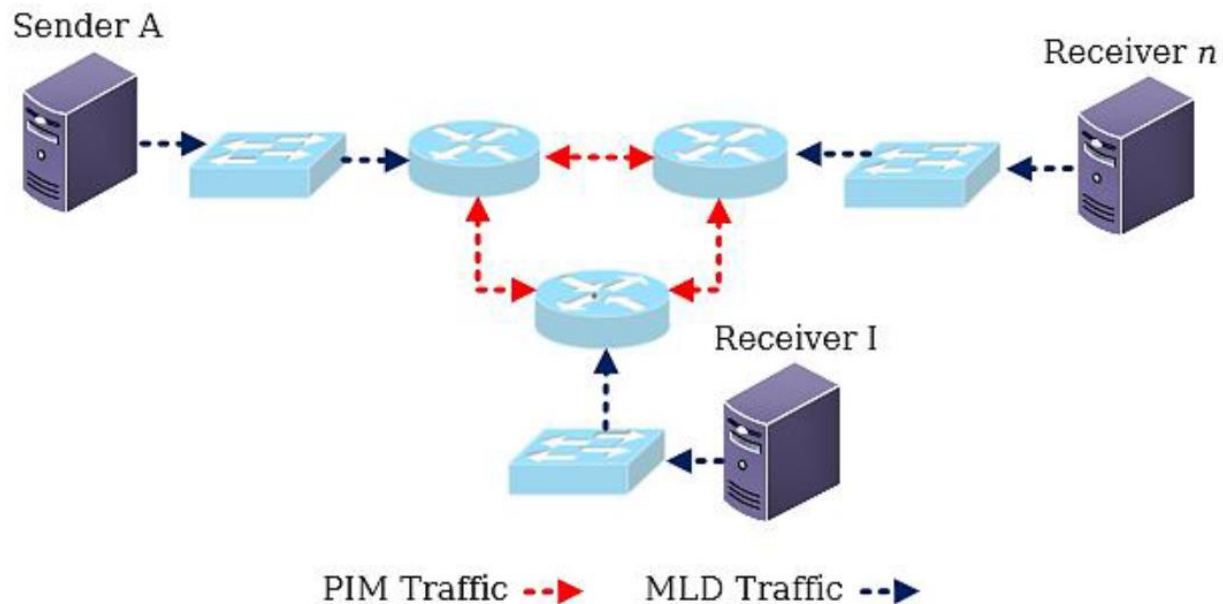
## Multicast in a Nutshell (I)



Communication between a [group of] source[s] and several receivers.

## Multicast in a Nutshell (II)

Receiver[s] have to signal to the routers that they're interested in certain channels.



## Where Multicast Is Used



- The usual suspects:
  - Video-conferencing
  - IPTV
  - Sensor-networks
  - Monitoring and logging



## IPv6 Multicast Listener Protocol (MLD)



- Replaces IPv4's IGMP
  - MLDv1 (RFC 2710) based on IGMPv2.
  - MLDv2 based on IGMPv3.
- Queriers & Hosts
  - Querier: network device (usually a router) that sends *query* message to discover which network devices are members of a given multicast group.
  - Receiver: node that sends *report* messages to inform querier about a group membership.



## MLD Version 1



- All MLD versions are based on ICMPv6.
- First defined in RFC 2710, derived from IPv4's IGMPv2.
- Used by IPv6 routers for discovering directly attached multicast listeners.
- In its original form MLD doesn't learn the exact identity or number of multicast listeners.

## MLD Version 2



- Specified in **RFC 3810** and equivalent to IGMPv3.
- Designed to be **interoperable** with **MLDv1**.
- Adds support for "source filtering". The nodes can report interest in traffic **only from a set** of source addresses or **from all except a set** source addresses.

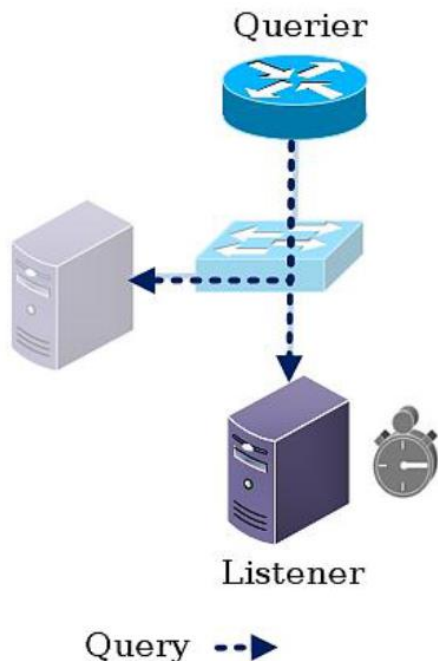


## MLDv1 Message Types

- Query (130)
  - General: Multicast address field set to 0 to learn which multicast addresses have listeners on an attached link.
  - Group/multicast-address specific.
- Report (131)
  - Sender of message (= a “receiver”) indicates which specific IPv6 multicast addresses it listens to.
- Done (132)
  - Sender of message (= a [former] “receiver”) indicates which address it no longer listens to.



## MLDv1 Query Messages



### General Queries:

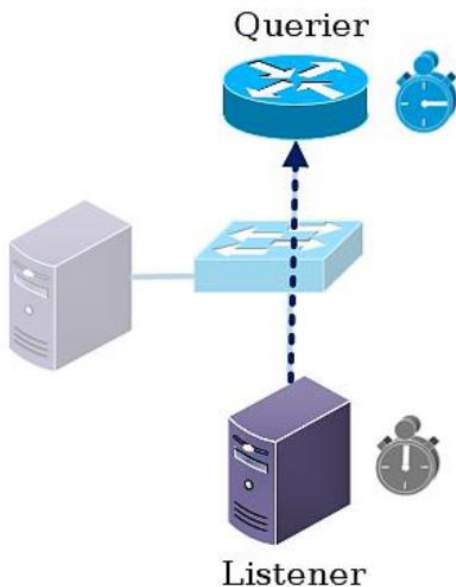
- Asks all listeners about multicast addresses of interest.
- Sent to `FF02::1` (link-scope all-nodes).

### Multicast-Address-Specific Queries:

- Ask listeners about a particular multicast address.
- Sent to the multicast address being queried.



## MLDv1 Listener Messages



Report -->

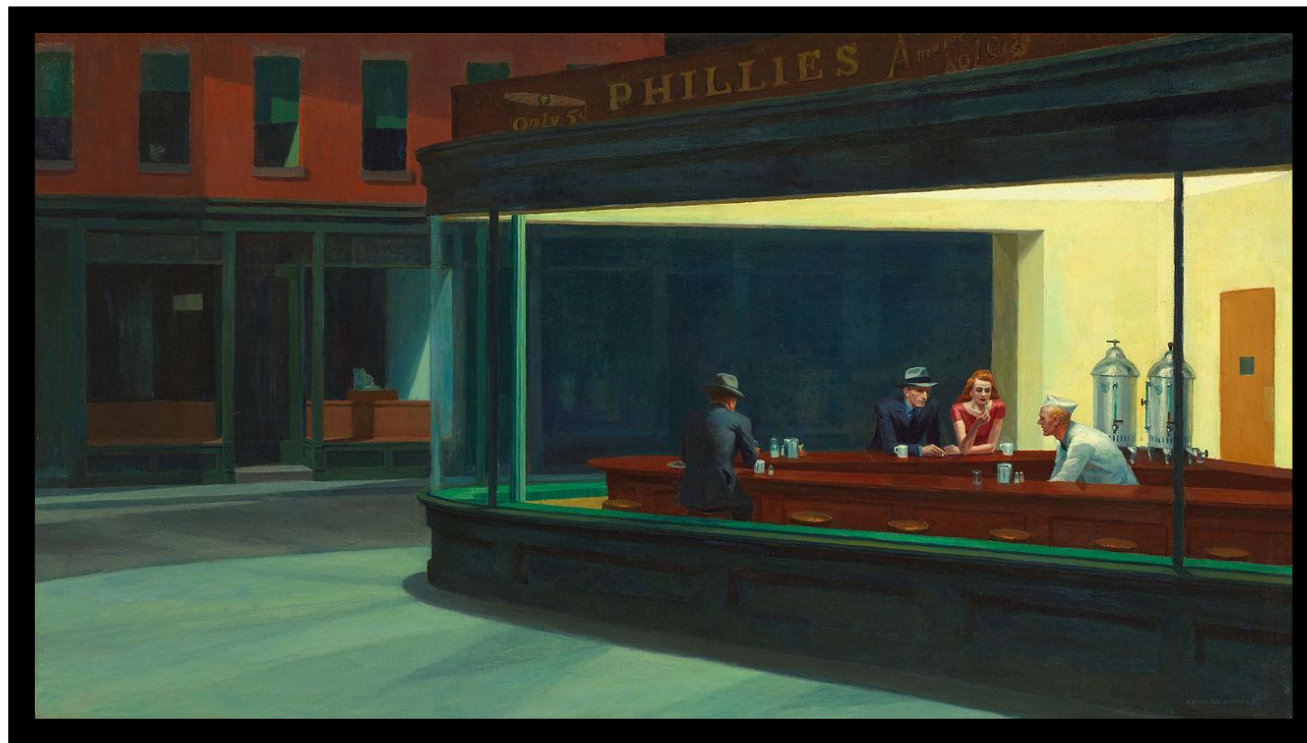
- Multicast Listener Report: ICMPv6 Type 131
  - Sent to the multicast address being reported.
- Multicast Listener Done: Type 132
  - Sent to FF02::2 (link-scope all-routers).



## MLDv2 Messages

- General Queries: ICMPv6 Type 130
  - Sent to FF02::1.
- Specific Queries: ICMPv6 Type 130
  - Inclusion of Address-and-Source-Specific queries.
  - All specific queries are sent to the multicast address being queried.
- MLDv2 Reports : ICMPv6 Type 143
  - Sent to FF02::16 (all MLDv2-capable routers).
  - No more MLD *Done* messages.



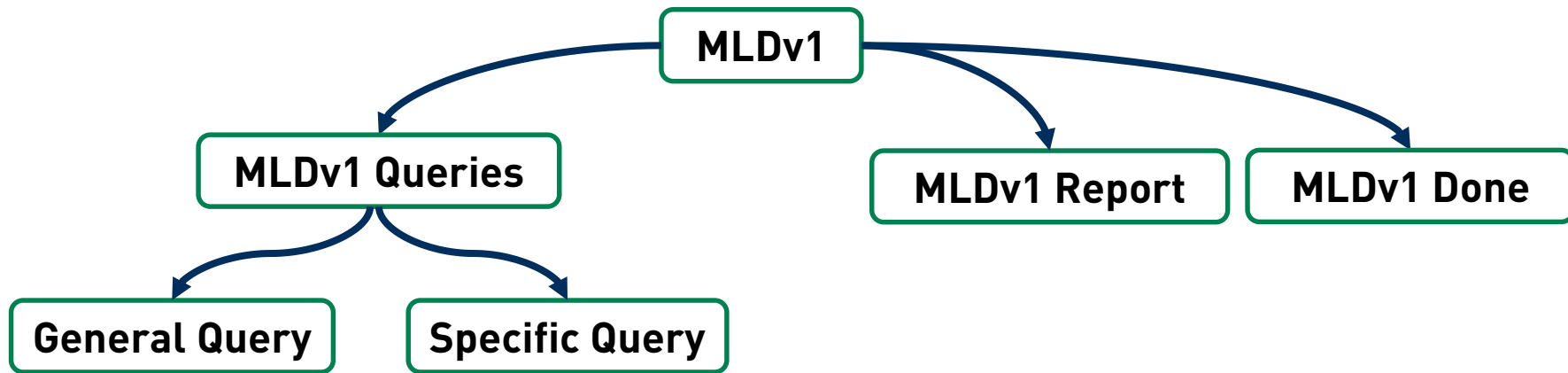


One  
Particularly  
Interesting  
Functionality:

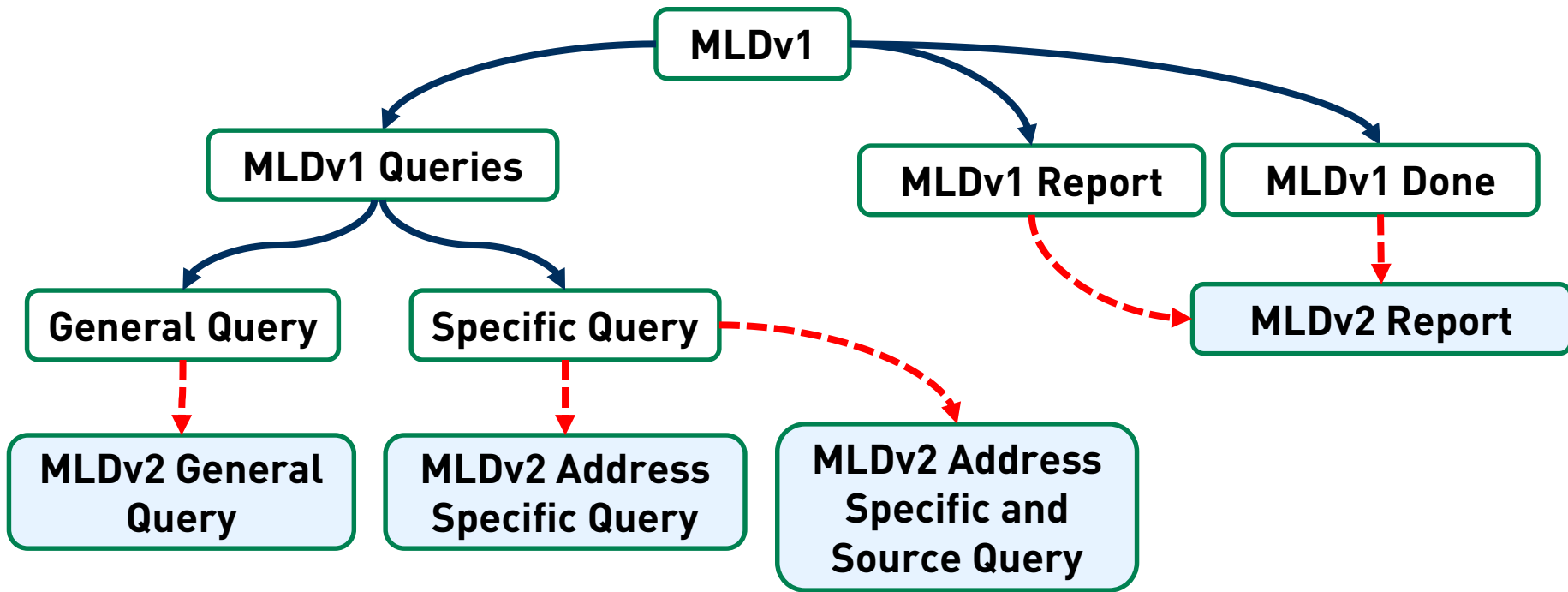
## Last Call

aka [The *last listener query*]

# Quick Recap: (Development of) MLD Messages



# Quick Recap: (Development of) MLD Messages



## MLD Snooping



- Switch based, somewhat proprietary feature that constrains multicast traffic to only the ports that have receivers attached.
- The switch builds an MLD table that basically maps a multicast group to all the switch ports that have requested it.

## Security Precautions



- All MLD messages must be sent with:
  - A *link-local* IPv6 source address.
  - An IPv6 Hop-Limit of 1.
  - A *Router Alert Option* in the Hop-by-Hop extension header.
  
- Non compliant messages must be dropped.

## Convenient RFC Conditions



- A node **MUST** process any *Query* whose destination address matches **any** of the addresses assigned to the receiving interface, unicast or multicast.

### Result:

- This allows one-to-one communication with the routers and listeners.





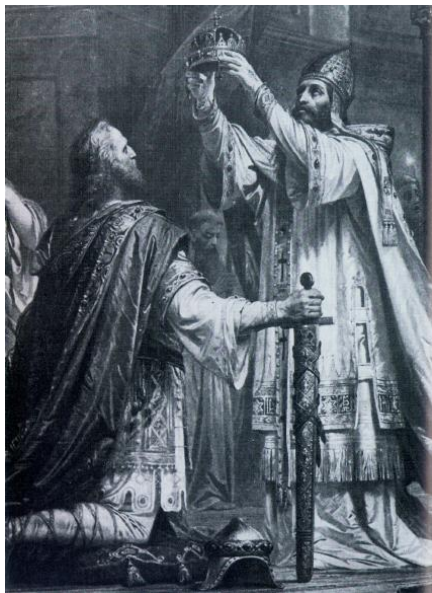
## Global Unicast Address as Destination?

---



- All but FreeBSD accept the Queries and respond.
- This means that we can interact directly with nodes without the Router/Querier being involved.

## Convenient RFC Conditions (II)



- A router in querier mode enters the non-querier state upon receiving a query from a lower IPv6 address than its own. It thus ceases to send queries.
- **Result:**
  - In most networks we can easily become a *Querier*.  
→ “Win the election”.



## Convenient RFC Conditions (III)



- In the presence of MLDv1 Routers, MLDv2 hosts **MUST** operate in version 1 compatibility mode.
- In the presence of MLDv1 Multicast Address Listeners, an MLDv2 node **MAY** allow its MLDv2 Report to be suppressed by a Version 1 Report.
- **Result:** We can easily force MLDv1 to be used.
  - In the 90s we called this a “forced dialect downgrade”...

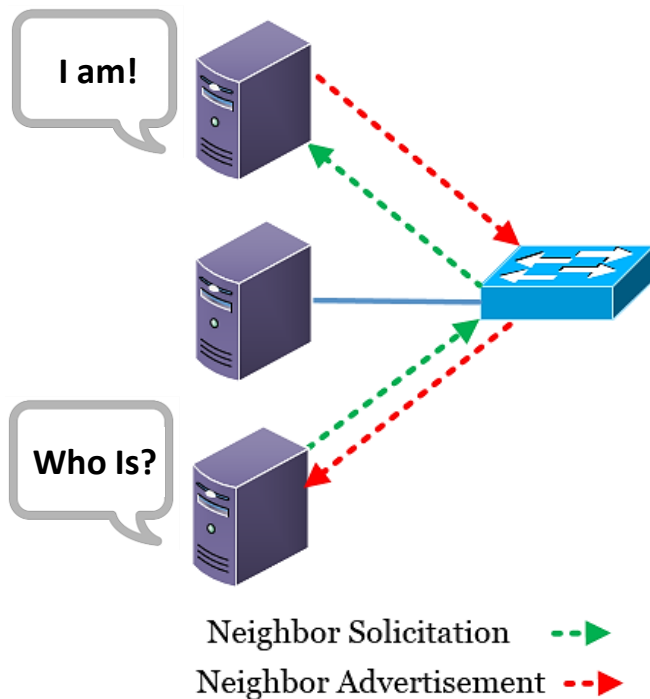


# Myths and Facts

The Other Face of MLD



## Let's Discuss the Neighbor Discovery Protocol



- **No broadcast, all-nodes** multicast address **instead**.
- **Every IPv6** address has a **associated** derived **Solicited-Node** multicast **group**.
- All relevant **Solicited-Node** groups **must be joined** by a node during interface initialization.
- RFC 4861: “**joining** the **solicited-node** multicast address **is done using** a Multicast Listener Discovery protocol such as the [MLD] or [MLDv2] protocols.”



## Duplicate Address Detection

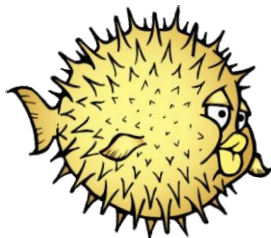
Note that when a node joins a multicast address, it typically sends a Multicast Listener Discovery (MLD) report message [[RFC2710](#)] [[RFC3810](#)] for the multicast address. In the case of Duplicate Address Detection, the MLD report message is required in order to inform MLD-snooping switches, rather than routers, to forward multicast packets. In the above description, the delay for joining the multicast address thus means delaying transmission of the corresponding MLD report message. Since the MLD specifications do not request a random delay to avoid race conditions, just delaying Neighbor Solicitation would cause congestion by the MLD report messages. The congestion would

RFC 4862



## Myths and Facts – MLD and ND

---



- If disabled in Windows, the Neighbor Discovery (ND) process does **not** work.
  - RFC 4862, sect. 5.4.2: “In the case of Duplicate Address Detection, the MLD report message is **required** in order to inform **MLD-Snooping** switches, rather than routers, to forward multicast packets.”
- However, if MLD messages are blocked by a host based firewall, ND works (even in Windows). See MLD/ND discussion on <http://www.insinuator.net/>.
- In OpenBSD, the ND process works normally without MLD being enabled.



## Myths and Facts - MLD and ND cont'd

- Moreover, when MLD-Snooping is enabled on a Cisco Catalyst 2960-S switch (at least with certain images), *solicited-node* multicast addresses are still “broadcasted”. Maybe [tools.ietf.org/html/draft-pashby-magma-simplify-mld-snooping-01](https://tools.ietf.org/html/draft-pashby-magma-simplify-mld-snooping-01) is implemented?

**So, do we really need MLD for Neighbor Discovery?**





## Implementation Facts

- MLD is pre-enabled in Windows, Linux and FreeBSD Operating Systems. It is **NOT** in OpenBSD.
- MLD Reports are sent even before the Neighbor Discovery Process starts.
- To cover the possibility of the initial Report being lost or corrupted, it is recommended to be resent once or twice after short delays.





## Implementation Facts (II)



- All of them join several multicast groups:
  - Each OS joins the corresponding Solicited-Node Multicast Address.
  - Windows joins **FF02::1:3** (Link Local Multicast Name Resolution).
  - FreeBSD joins Node Information Queries multicast groups (**experimental** RFC 4620).



## Trivial Host Enumeration and Fingerprinting (II)

Time	Source	Destination	Protocol	Length
0.000000	Windows7.1-linklocal	ff02::16	ICMPv6	90
0.000013	Windows7.1-linklocal	ff02::16	ICMPv6	90
0.008497	Windows7.1-linklocal	ff02::16	ICMPv6	90
0.008506	Windows7.1-linklocal	ff02::16	ICMPv6	90
0.023971	Windows7.1-linklocal	ff02::16	ICMPv6	90
0.023984	Windows7.1-linklocal	ff02::16	ICMPv6	90
0.025772	Windows7.1-linklocal	ff02::16	ICMPv6	90
0.025777	Windows7.1-linklocal	ff02::16	ICMPv6	90
0.261958	Windows7.1-linklocal	ff02::16	ICMPv6	90
0.261967	Windows7.1-linklocal	ff02::16	ICMPv6	90
600.048733	Windows7.1-linklocal	ff02::16	ICMPv6	90
600.048746	Windows7.1-linklocal	ff02::16	ICMPv6	90
600.063445	Windows7.1-linklocal	ff02::16	ICMPv6	90
600.063458	Windows7.1-linklocal	ff02::16	ICMPv6	90
600.075012	Windows7.1-linklocal	ff02::16	ICMPv6	90
600.075020	Windows7.1-linklocal	ff02::16	ICMPv6	90
600.077356	Windows7.1-linklocal	ff02::16	ICMPv6	90
600.077366	Windows7.1-linklocal	ff02::16	ICMPv6	90
600.264367	Windows7.1-linklocal	ff02::16	ICMPv6	90
600.264378	Windows7.1-linklocal	ff02::16	ICMPv6	90
199.407524	Windows7.1-linklocal	ff02::16	ICMPv6	90

- MLD is the perfect protocol for the job.
- Pre-enabled in Windows, Linux and FreeBSD.
- Reports are sent even before the ND Process starts.
- Hosts must respond to Queries.
- Works even when responses to ICMPv6 Echo Requests are disabled/blocked.
  - As is the default case for Win 8.1.



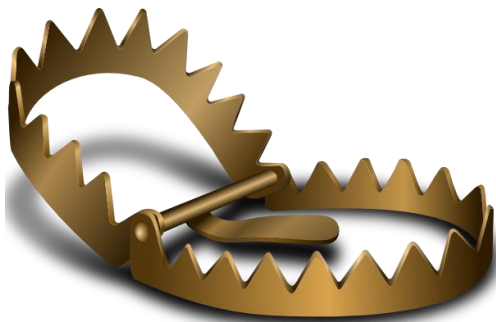
## Trivial Host Discovery and Fingerprinting (II)

OS	Multicast Group	Service
IOS 15.4(3) M	ff02::2	All IPv6 routers on the Link
	ff02::d	PIM routers
	ff02::16	All MLDv2 capable routers
	ff02::1:2	All DHCP servers and relay agents
FreeBSD 10.0	ff02::2:ff2e:b774	IPv6 Node Information <i>Query</i>
	ff02::2:2eb7:74fa	IPv6 Node Information <i>Query</i> (Invalid)
Ubuntu 14.04	ff02::FB	Zero Configuration Networking
Windows 8.1	ff02::C	SSDP
	ff02::1:3	LLMNR



## Homework: MLD- Related Vulnerabilities

- Six MLD-related CVEs as of Nov 2014.
- Four related to Cisco products (2012,2013,2014)
  - Two of them when MLD Snooping is enabled, one related with VRFs and one with MLD tracking.
- One on NetBSD 4.0, FreeBSD, and KAME (2008).
- One on Windows XP, 2003 and Vista (2007).



## Same Procedure Every ~~Year~~ Protocol

– Read the specs



– Build a lab



– Create a test plan



– Have fun ;-)



## What To Look For



- Implementation problems
  - Yes, fuzzing.
  - We mean, what else ;-)
- RFC compliance issues
  - These may sound lame... but we'll see that they can serve as a stepping stone for the next category.
- Design flaws & unwanted/-expected protocol behavior.

## Devices Used in the Lab



- For routers: mainly Cisco 1921, IOS15.4(3)M, plus an ASR 1002.
- For switches: Cisco Catalyst 2960-S IOS 15.2(1)E3.
- As hosts: latest Windows (server, desktop), some Linuces, FreeBSD and OpenBSD.





## Tools

Our approach

### Chiron

- Abusing the protocol
- Chiron now has MLD capabilities  
→ New version will be available:  
<http://www.secfu.net/tools-scripts/>
- There is a Chiron workshop tomorrow ;-)



### Dizzy

- Fuzzing
  - Latest version: <http://www.insinuator.net/2014/02/fresh-meet-from-the-coding-front>
- New description files for MLD available  
→ To be released after the Troopers15.





# Results

---





## RFC Compliance Issues, Linux

---

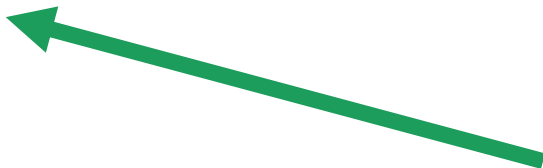


- Linux systems up to kernel v3.16 accept MLD messages with **Hop Limit > 1**.
- This is also the case for MLD messages with no Router Alert Option (but not that important).
- Centos 6.x accepts MLD messages when the source address is a **link-local multicast** one.



# Windows

```
+ Frame 1: 86 bytes on wire (688 bits), 86 bytes captured (688 bits) on interface 0
+ Ethernet II, Src: 00:ee:4c:62:05:6e (00:ee:4c:62:05:6e), Dst: CadmusCo_b2:ef:98 (00:0c:29:00:00:00)
+ Internet Protocol Version 6, Src: fe80::2ee:4cff:fe62:56e (fe80::2ee:4cff:fe62:56e), Dst: fe80::2ee:4cff:fe62:56e (fe80::2ee:4cff:fe62:56e)
- Internet Control Message Protocol v6
  Type: Multicast Listener Query (130)
  Code: 0
  Checksum: 0xfdb0 [correct]
  Maximum Response Delay [ms]: 0
  Reserved: 0000
  Multicast Address: :: (::)
```



MLDv1 query sent to the unicast address of a Windows 2012R2 DHCPv6 Server.



# Windows

No.	Time	Source	Destination
1	0.000000	fe80::2ee:4cff:fe62:56e	2001:db9:1:1::1
2	0.000596	fe80::888:c9b2:1d13:66a2	ff02::1:ff13:66a2
3	0.000616	fe80::888:c9b2:1d13:66a2	ff02::1:ff13:66a2
4	0.001009	fe80::888:c9b2:1d13:66a2	ff02::1:3
5	0.001024	fe80::888:c9b2:1d13:66a2	ff02::1:3
6	0.001336	fe80::888:c9b2:1d13:66a2	ff02::1:ff00:1
7	0.001350	fe80::888:c9b2:1d13:66a2	ff02::1:ff00:1
8	0.001790	2001:db9:1:1::1	ff05::1:3
9	0.001807	2001:db9:1:1::1	ff05::1:3
10	0.002155	fe80::888:c9b2:1d13:66a2	ff02::1:2
11	0.002173	fe80::888:c9b2:1d13:66a2	ff02::1:2
12	0.002566	fe80::888:c9b2:1d13:66a2	ff02::1:ffd0:5adc
13	0.002582	fe80::888:c9b2:1d13:66a2	ff02::1:ffd0:5adc

In this case Windows responds with a report from its global unicast address.

Frame 1: 86 bytes on wire (688 bits), 86 bytes captured (688 bits)  
Ethernet II, Src: 00:ee:4c:62:05:6e (00:ee:4c:62:05:6e), Dst: CadmusCo\_b2:ef:98 (C  
Internet Protocol Version 6, Src: fe80::2ee:4cff:fe62:56e (fe80::2ee:4cff:fe62:56e  
Internet Control Message Protocol v6  
Type: Multicast Listener Query (130)  
Code: 0  
Checksum: 0xadb0 [correct]  
Maximum Response Delay [ms]: 0  
Reserved: 0000  
Multicast Address: :: (::)

## Why the Source Address Matters

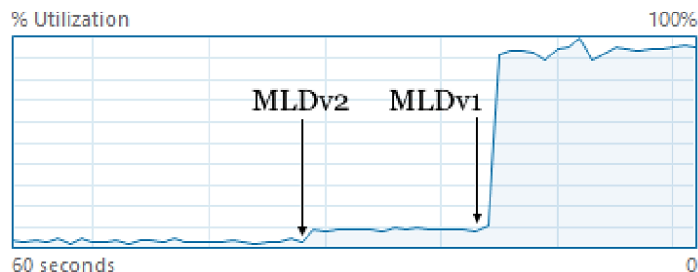


- When an MLD Report with a non link-local address as source is received:
  - In MLDv2, it is strictly defined that it MUST be dropped.
  - In MLDv1 it is not strictly mentioned, but if accepted this would mean that we would be able to interact with routers remotely.

# Windows Behavior

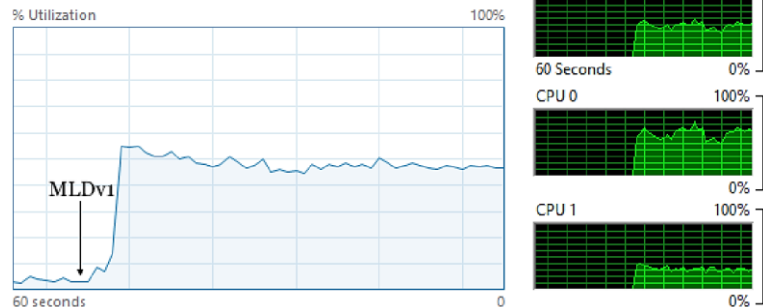
## CPU

AMD Phenom(tm) II X6 1055T Processor



## CPU


AMD Phenom(tm) II X6 1055T Processor



- In Windows 7 and 8.1 systems the process in charge of MLD + Interrupts processing can **consume up to one** processor core.

# Huge MLD Reports, Router Resource Depletion

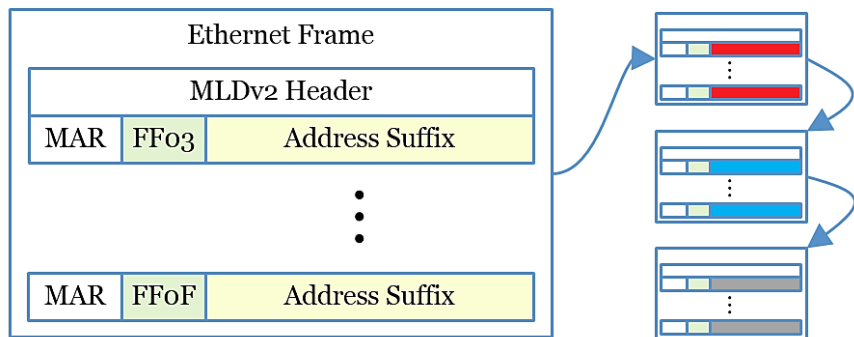
```
user@ubuntu: ~  
  
My traceroute [v0.85]  
ubuntu (::)                               Fri Jan 16 16:43:24 2015  
Keys: Help  Display mode  Restart statistics  Order of fields  quit  
Packets  
Host      Loss%  Snt    Last   Avg    Best   Wrst   StDev  
1. 2001:db8:1::ec:1  0.0%   71     0.6    0.6    0.3    1.0    0.0  
2. 2001:db8:2::ec:1  0.0%   71     0.9    0.8    0.6    2.6    0.2
```



```
user@ubuntu: ~  
  
My traceroute [v0.85]  
ubuntu (::)                               Fri Jan 16 16:36:04 2015  
Keys: Help  Display mode  Restart statistics  Order of fields  quit  
Packets  
Host      Loss%  Snt    Last   Avg    Best   Wrst   StDev  
1. 2001:db8:1::ec:1  0.0%   73     22.1   7.2    0.4    78.2   11.5  
2. 2001:db8:2::ec:1  8.2%   73     0.8    4.5    0.6    80.0   13.5
```



# Huge Reports Fill the Cache in about 30s



- Device **becomes unresponsive**, **packets** start being **dropped** and **latency** goes **up**
- Further **Listeners aren't able** to **join** multicast groups since the table is effectively full
- Putting a **hard limit** on the number of entries **isn't likely** to **help**



# The PIM IPv6 Process Fails, Not that Bad

**%SYS-2-MALLOCFAIL:** Memory allocation of 65536 bytes failed from 0x21028EF4, alignment 0

Pool: Processor Free: 419724 Cause: Memory fragmentation

Alternate Pool: None Free: 0 Cause: No Alternate pool

-Process= "**PIM IPv6**", ipl= 0, pid= 329

-Traceback= 21010528z 210109FCz 2101E0FCz 24B69248z 24B2C374z 24B2F324z  
231FA520z 231F7FA8z 24B30408z 24B30C2Cz 231D41D8z 231D4D40z 231D4F60z  
24B3CDF8z 210329B4z 21032998z



# IPv6 Addresses can't be Leased, Hm

```
%SYS-2-MALLOCFAIL: Memory allocation of 232 bytes failed from  
0x24A42624, alignment 0 Pool: Processor Free: 1800716 Cause: Memory  
Fragmentation  
Alternate Pool: None Free: 0 Cause: No Alternate pool  
-Process= "DHCPv6 Server", ipl= 0, pid= 338  
-Traceback= 210z 24A3782Cz 24A37C2Cz 24A37DD4z 210329B4z 21032998z
```



# Neither does SSH work, Oh Well ...

```
%SYS-2-MALLOCFAIL: Memory allocation of 12252 bytes failed from  
0x249F0200, alignment 0  
Pool: Processor Free: 1312500 Cause: Memory fragmentation  
Alternate Pool: None Free: 0 Cause: No Alternate pool  
-Process= "Exec", ipl= 0, pid= 3  
-Traceback= 210121E8z 249E5408z 24A098B0z 24A062B4z 24A085D8z  
24A08AF4z 22909EA0z 22911F60z 22924164z 210329B4z 21032998z
```

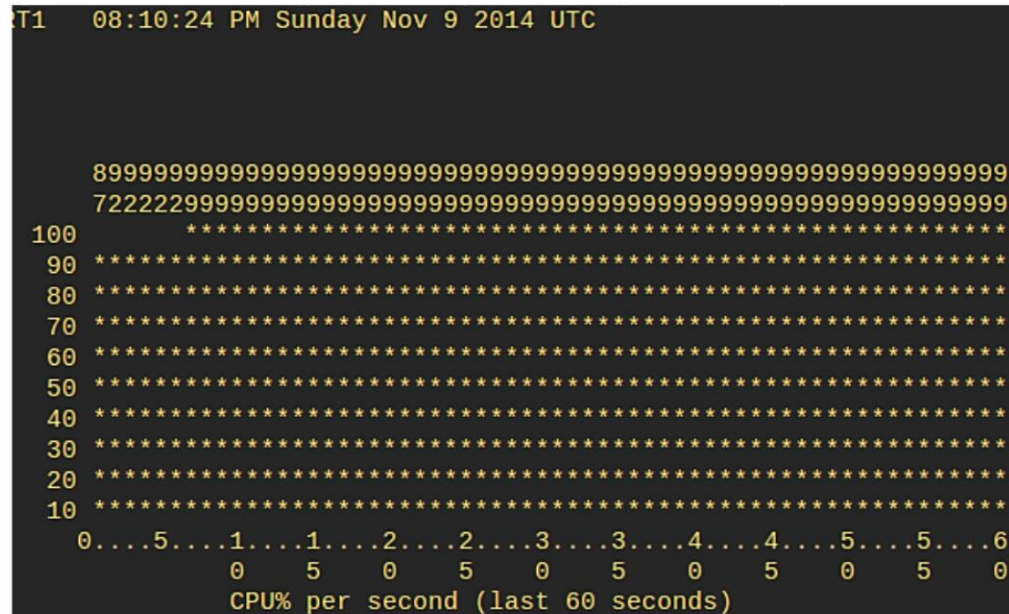


## Flooding

Demo



Here, the router is a Cisco ASR 1002,  
there's only one attacker on *local-link*...





## Amplification Attacks

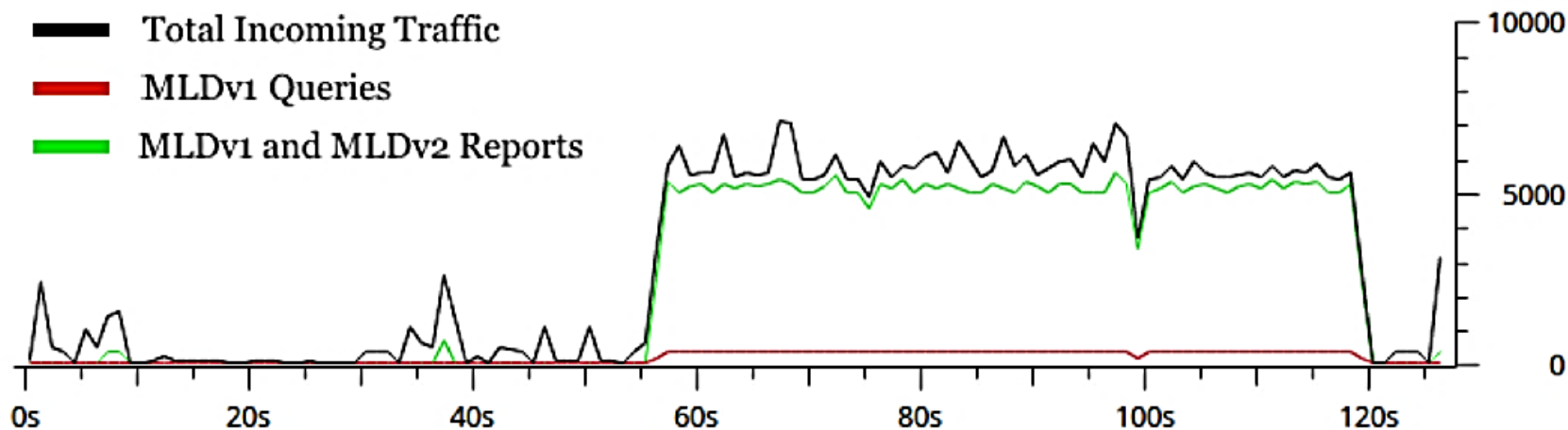
Against the routers on the *local-link* using MLD Queries.



- Windows 8.1 hosts join at least four groups and send two Reports per group.
  - Amplification factor goes up to **8 x Number of machines** for Windows hosts.
  - For example, in a LAN with 200 hosts a single spoofed Query can trigger 1600 Reports all sent immediately to the router.
    - Did you get that? Amplification factor: 1.600!!
- What if we flood the link with such Queries?

# MLDv1 Traffic Amplification

- 1,3kb/s become 49,8kb/s on the router's side, **~3830%** the initial traffic





## How to Break MLD



- Cisco IOS15.4(3)M accepts:
  - MLDv1 and MLDv2 Queries sent to FF02::2.
  - MLDv2 Queries to FF02::16 and its unicast address.
  - MLDv1 and MLDv2 Queries to its link-local address.
  - MLDv2 Reports sent to FF02::2 and FF02::16.
  - MLDv1 Dones sent to the FF02::2, FF02::16, link-local and unicast addresses.
- **Result:** We have several ways to interact with the routers in a **one-to-one** manner.

# Let's Have a Look at a Practical Attack

Sender A  
2001:DB8:2::C001



Receiver n  
2001:DB8:1::C001



**Attacker**

PIM Traffic -->

MLD Traffic -->



## Attack Vector I - MLDv1 and MLDv2

---



- Take over the Querier Role.
- Send spoofed MLDv1 Done or MLDv2 Reports to remove a listener from a multicast group.
- Send a spoofed Last Listener Query to the routers, they believe this to be a real Last Listener Query.
- Periodically send Generic Queries to the routers (FF02::2, FF02::16 or their unicast addresses).

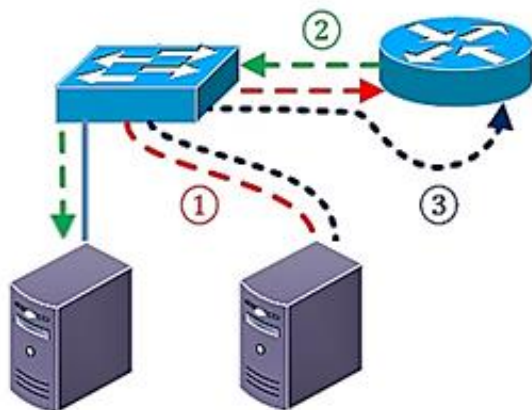


## Attack Vector II – MLDv1



- Become Querier through MLDv1 Queries, forcing use of MLDv1. Same can be done by sending MLDv1 Reports.
- Send MLDv1 Done messages.  
The Querier (or you) sends a “last call” Query.
- Send MLDv1 Report to the unicast address of the legitimate listeners to trigger Report suppression on their side.
- Legitimate routers do not receive any Reports and thus traffic to the group is no longer forwarded.

# The Last Call for Drinks, Last-Listener-Queries



MLDv2 Report or MLDv1 Done -->

Last Listener Query -->

MLD General Query -->

- **Last-Listener-Queries** are **sent** by the Querier **when** a Listener expresses its **lack of interest** in certain traffic.
- Is **sent** as a **Specific-Query** to the multicast address which is being queried.
- An **attacker** can **become** the **Querier**, **leave** a **group** on behalf of a client and **fake** a **Last-Listener-Query**.



# However, Something was Missing

	SRC MAC	SRC ADD	MLD MADDR	Len.
47.373682000	ubuntu_eth0	ubuntu.local	ff08::db8	90
47.373696000	ubuntu_eth0	ubuntu.local	ff08::db8	90
56.087140000	Cisco_15:c0:11	fe80::200:cff:fe15:c011		90
58.028565000	ubuntu_eth0	ubuntu.local	ff08::db8, ff02::fb, ff02::1:ff00:12	130
58.028578000	ubuntu_eth0	ubuntu.local	ff08::db8, ff02::fb, ff02::1:ff00:12	130
38.885241000	kali_eth0	fe80::200:ff:fe00:14	ff08::db8	90
38.885255000	kali_eth0	fe80::200:ff:fe00:14	ff08::db8	90
01.332813000	Cisco_15:c0:11	fe80::200:cff:fe15:c011		90
09.418357000	ubuntu_eth0	ubuntu.local	ff08::db8, ff02::fb, ff02::1:ff00:12	130
09.418367000	ubuntu_eth0	ubuntu.local	ff08::db8, ff02::fb, ff02::1:ff00:12	130
06.582484000	Cisco_15:c0:11	fe80::200:cff:fe15:c011		90
13.996287000	ubuntu_eth0	ubuntu.local	ff08::db8, ff02::fb, ff02::1:ff00:12	130
13.996304000	ubuntu_eth0	ubuntu.local	ff08::db8, ff02::fb, ff02::1:ff00:12	130

# Let's Have a Look at a Practical Attack

Sender A  
2001:DB8:2::C001



Receiver n  
2001:DB8:1::C001



**Attacker**

PIM Traffic -->

MLD Traffic -->



Real Life Scenario:  
**Shareholders'  
Meeting**

**Demo**





# Mitigation

---





# ERNW's *Seven Sisters* of Infrastructure Security



Access Control



Isolation



Restriction



Encryption



Entity Protection



Secure Management



Visibility

See also: <http://www.insinuator.net/tag/seven-sisters/>

## General Use

- The building blocks can be “applied” to all components / technologies / protocols.

Just ask yourselves:

- What is the “scope”? Can it be limited?
- Can (the traffic) be filtered / restricted?
- Are there authentication mechanisms?
- How’s the stuff being managed?
- Any hardening (of a device or service) possible?
- What about logging / monitoring?



## Mitigations for Admins



- Filter MLD Queries on the switch port level
  - Think “MLD Guard” (which does not exist).
  - = Port based ACL filtering ICMPv6 type 130
    - `deny icmp any any mld-query`
- Alternatively, in a MLD snooping scenario statically configure a port as an **mrouter** port.



## Mitigations for Admins (II)

---



- At routers specify a limit on the rate that MLD Reports should be accepted from each host. MUST drop all the reports that exceed this limit.
- Consider “`no ipv6 mld router`” if there’s no inter-domain multicast routing in the environment.



## Mitigations for Admins (III)



- At switches with MLD-snooping enabled:
  - You might use *static-groups* to protect critical multicast based services (e.g. DHCPv6)
    - Keep operational impact/effort in mind ;-)
  - MLD snooping listener message suppression is enabled by default → forwards **only one** MLD report per response to multicast router queries.
  - If technically possible, limit the rate at which MLD messages are accepted by nodes.

## In the Standards Space



- **MLDv2:** Routers shouldn't accept Queries destined to **FF02::2**, **FF02::16**, or unicast addresses (link-local or global).
- **MLDv1:** Nodes **MUST** not accept Reports to their unicast addresses (not even for debugging purposes).
- **Both:** Do not permit querier role take over by simply using a “lower” IPv6 address.

## Conclusions



- In the IPv6 world there's a protocol called MLD.
  - It's complex & somewhat flawed, we think.
  - It's ubiquitous.
  - There's quite some potential for abuse
    - Huge local amplification attacks.
    - Disruption of network services.
- Security research in the IPv6 world is much needed.
  - And it's fun. Get your hands dirty.





There's never enough time...

**THANK YOU...**



**...for yours!**

**Tool & Slides:**


<https://www.insinuator.net>


<http://www.secfu.net/tools-scripts/>



## Questions?



- You can reach us at: 
  - [aatlasis@secfu.net](mailto:aatlasis@secfu.net), [www.secfu.net](http://www.secfu.net)
  - [erey@ernw.de](mailto:erey@ernw.de), [www.insinuator.net](http://www.insinuator.net)
  - [jsalazar@ernw.de](mailto:jsalazar@ernw.de), [www.insinuator.net](http://www.insinuator.net)

- Follow us at: 
  - @AntoniosAtlasis
  - @Enno\_Insinuator