Practical Network Defense

Master's degree in Cybersecurity 2024-25

Link-local attacks with lab

Angelo Spognardi
spognardi@di.uniroma1.it
Dipartimento di Informatica
Sapienza Università di Roma

Agenda

- Network eavesdropping
- ARP poisoning
  - With lab activity
- MITM
  - With lab activity
- IPv6 Neighbor Discovery threats
- Rogue DHCP
- IPv6 Rogue RA (or RA spoofing)
  - With lab activity on ICMPv6 redirection attack

Network sniffing

Network eavesdropping (or sniffing)

- Capturing packets from the network transmitted by others' nodes and reading the data content in search of sensitive information
  - passwords, session tokens, or alike
- Done by using tools called network sniffers (or protocol analyzers)
  - Huge list of tools: not exhaustive list includes Ettercap, bettercap, networkminer, driftnet, dsniff, macof…
- Analyze the collected data like protocol decoders or stream reassembling
- Work in passive mode
  - packets are simply captured, copied, and passed at user level for further analysis
- Requires to be along the path or a broadcasting domain

Realize network sniffing

- Networking interface in promiscuous mode
- Sniffer must be along the path or, at least, in the same network
  - Non-switched LAN (LAN with HUBs)
    - the ideal case because the hub duplicates every frame to all ports
  - LAN with switches
    - breaking switch segmentation, sometimes by flooding the switch with a large amount of frames (MAC flooding)
    - performing arp spoof attack to redirect the traffic from one port to another
      - possible Man-In-The-Middle attack
  - wireless LAN
    - possible if no encryption is used or weak encryption is used (scenario becomes equivalent to LAN wtih HUBs)

Breaking the switch segmentation mechanism

- Flashback: bridge and switch
- Bridges: first way to reduce collisions and segment a network.
  - Have two ports joining to network segments
  - Only frames supposed to go on the other segment of the network are replicated (filtering)
  - “store & forward”: read and regenerate a frame only if needed
- Switches: multiport bridges
  - Regenerate a frame only in the segment of the destination
  - Learn the host in each network segment in real time

MAC Address/CAM Table Review

- CAM Table stands for Content Addressable Memory
- The CAM Table stores information such as MAC addresses available on physical ports with their associated VLAN parameters
- CAM Tables have a fixed size
- As frames move in the switches, the CAM is filled with the MAC addresses
  - Ex: source MAC address are associated with the related port
- If a MAC is unknown, it is replicated on ALL the ports → flood

CAM overflow

- Theoretical attack until May 1999
  - macof tool since May 1999
  - About 100 lines of perl from Ian Viteck
  - Later ported to C by Dug Song for "dsniff"
- Based on CAM Table's limited size
- Usually switches use hash to place MAC in CAM table
  - Like hashed lists, where buckets can keep a limited number of values
  - If the value is the same there are n buckets to place CAM entries, if all n are filled the packet is flooded

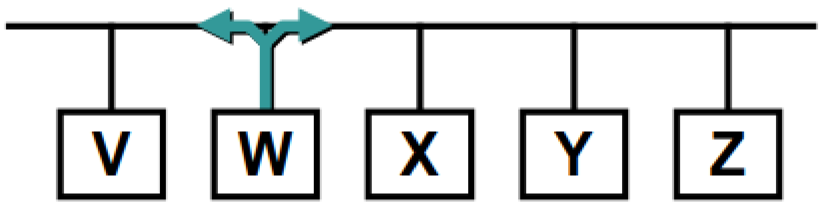
What happens next?

- It depends…
  - Switch starts flooding (attack success)
  - Switch freezes (denial of service)
  - Switch crash (denial of service)
- Today not really effective: port security in switches
  - Allows you to specify MAC addresses for each port, or to learn a certain number of MAC addresses per port
  - Upon detection of an invalid MAC the switch can be configured to block only the offending MAC or just shut down the port

ARP spoofing

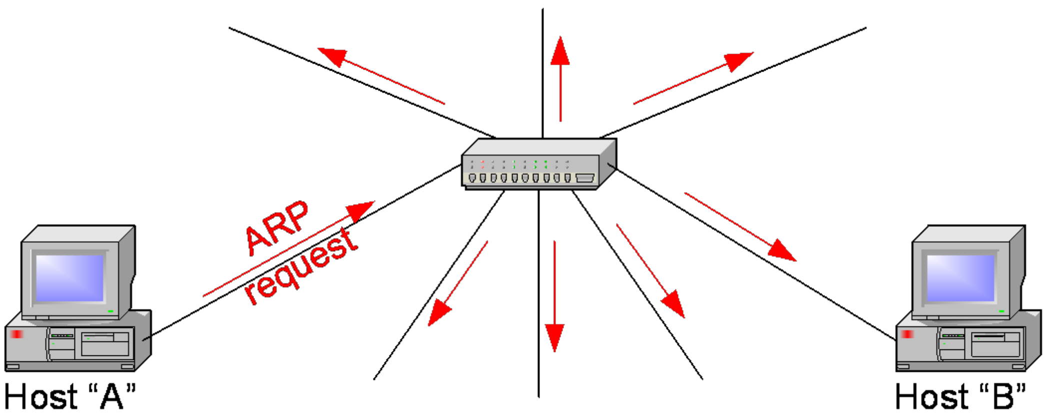
ARP

- An ARP request message should be placed in a frame and broadcast to all computers on the network
- Each computer receives the request and examines the IP address
- The computer mentioned in the request sends a response; all other computers process and discard the request without sending a response



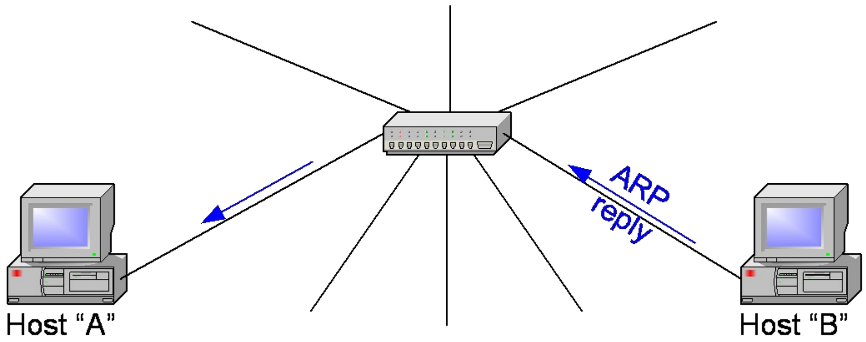
ARP 1

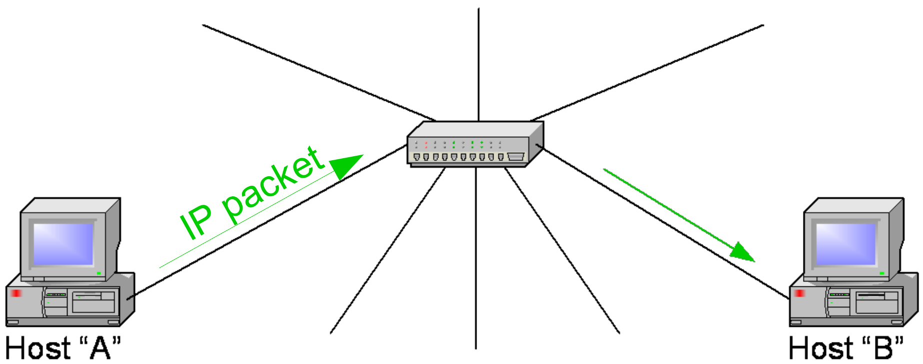
- Host A has the IP address of host B
- It knows it is in the same network → has to use Ethernet
- It needs to know the MAC address of host B
- It broadcasts an ARP request for IP of host B (MAC Dest = ff:ff:ff:ff:ff:ff)



ARP 2

- Host B sends back an ARP-reply
- The ARP reply has the MAC address of B as source and MAC address of A as destination
  - It was in the original ARP-request
- Then Host a can finally send the IP packet



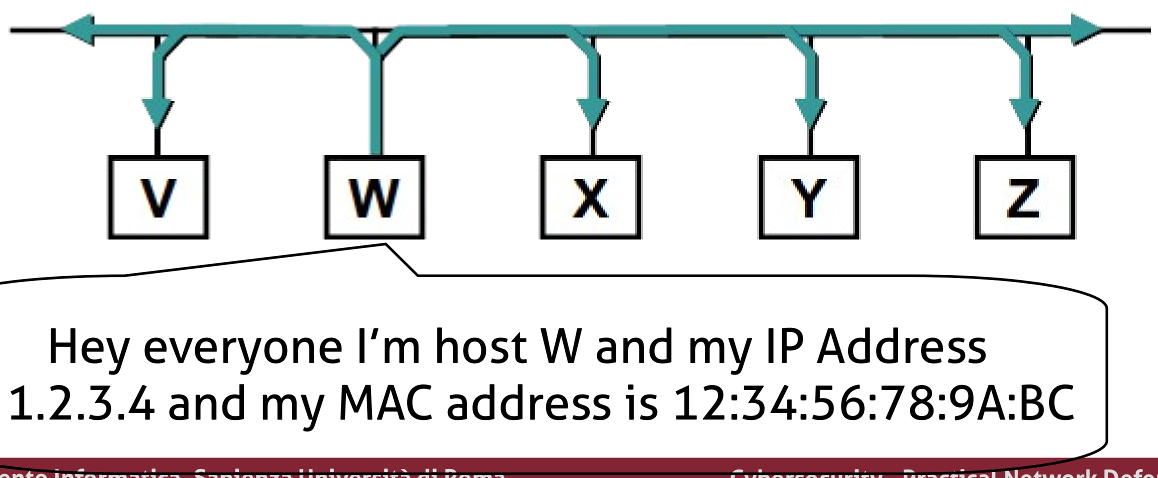


ARP table

- Dynamic table that holds the IP-MAC pairings
- It is accessed before sending any Ethernet frame
- It starts empty and is filled as the MAC addresses are collected
- Unused MAC addresses are removed after a timeout (address ageing) in the order of minutes
- According to RFC 826 (ARP), when receiving an ARP reply, the IP-MAC pairing is updated (age and pairing...)

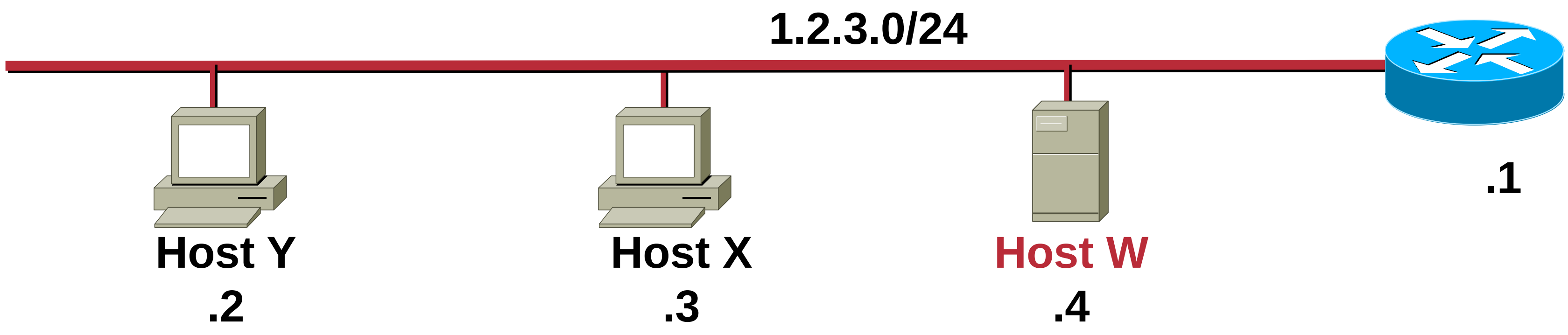
Gratuitous ARP responses

- Gratuitous ARP is used by hosts to "announce" their IP address to the local network and avoid duplicate IP addresses on the network
- Routers and other network hardware may use cache information gained from gratuitous ARP responses
- Gratuitous ARP is a broadcast packet (like an ARP-request)

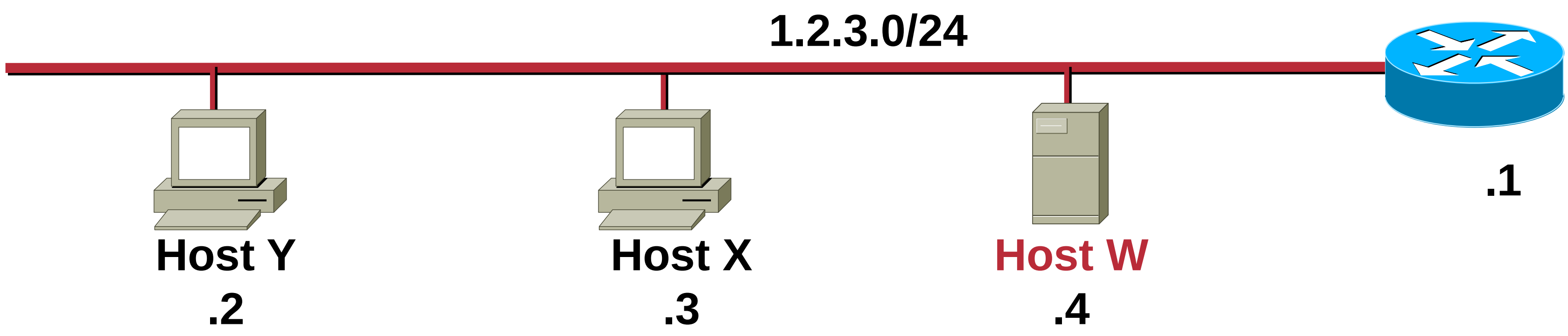


Misuse of Gratuitous ARP

- ARP has no security or ownership of IP or MAC addresses
  - Host W broadcasts I'm 1.2.3.1 with MAC 12:34:56:78:9A:BC
  - Wait 5 seconds, and then host W broadcasts I'm 1.2.3.1 with MAC 12:34:56:78:9A:BC
  - Repeat…
- What happens?



Misuse of Gratuitous ARP

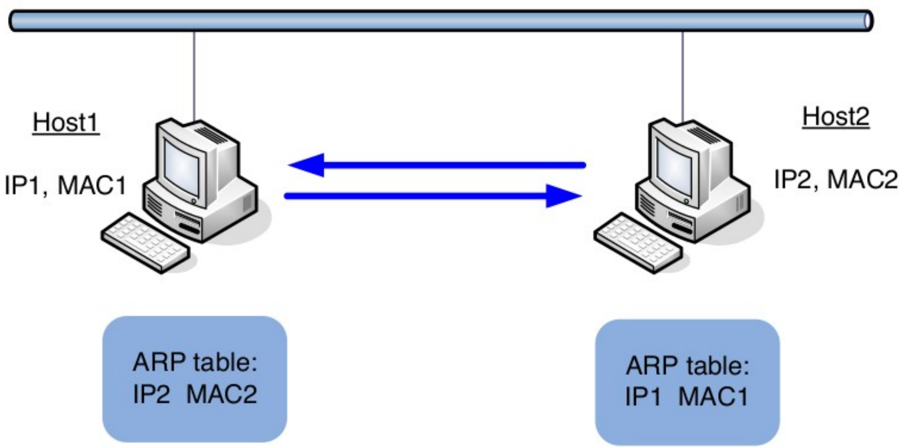


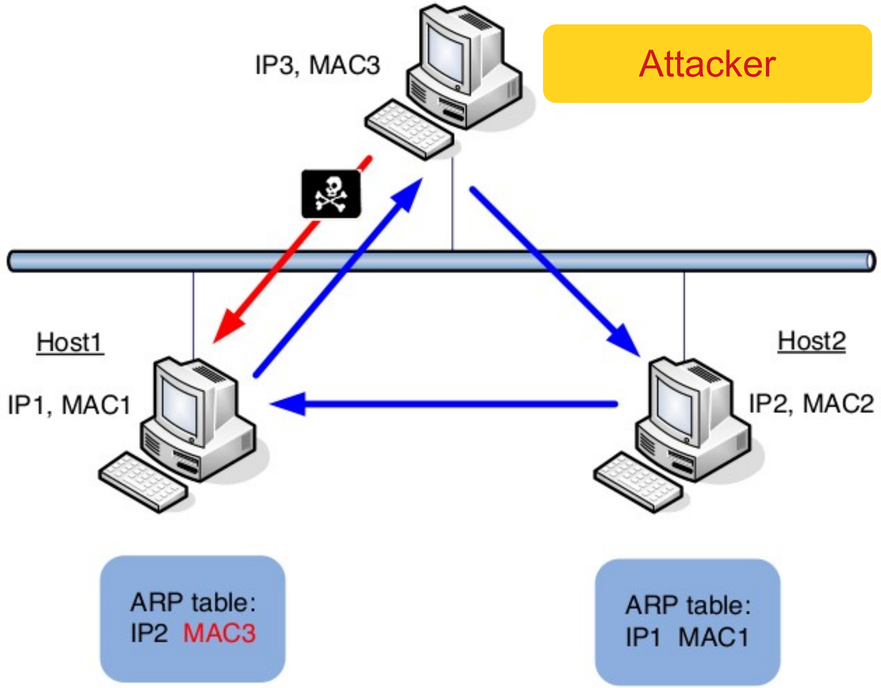
- Host X and Y will likely ignore the message unless they currently have an ARP table entry for 1.2.3.1
- When host Y requests the MAC of 1.2.3.1 the real router will reply and communications will work until host W sends a gratuitous ARP again
  - Even a static ARP entry for 1.2.3.1 on Y will get overwritten by the Gratuitous ARP on some OSs

Hijacking in the local network

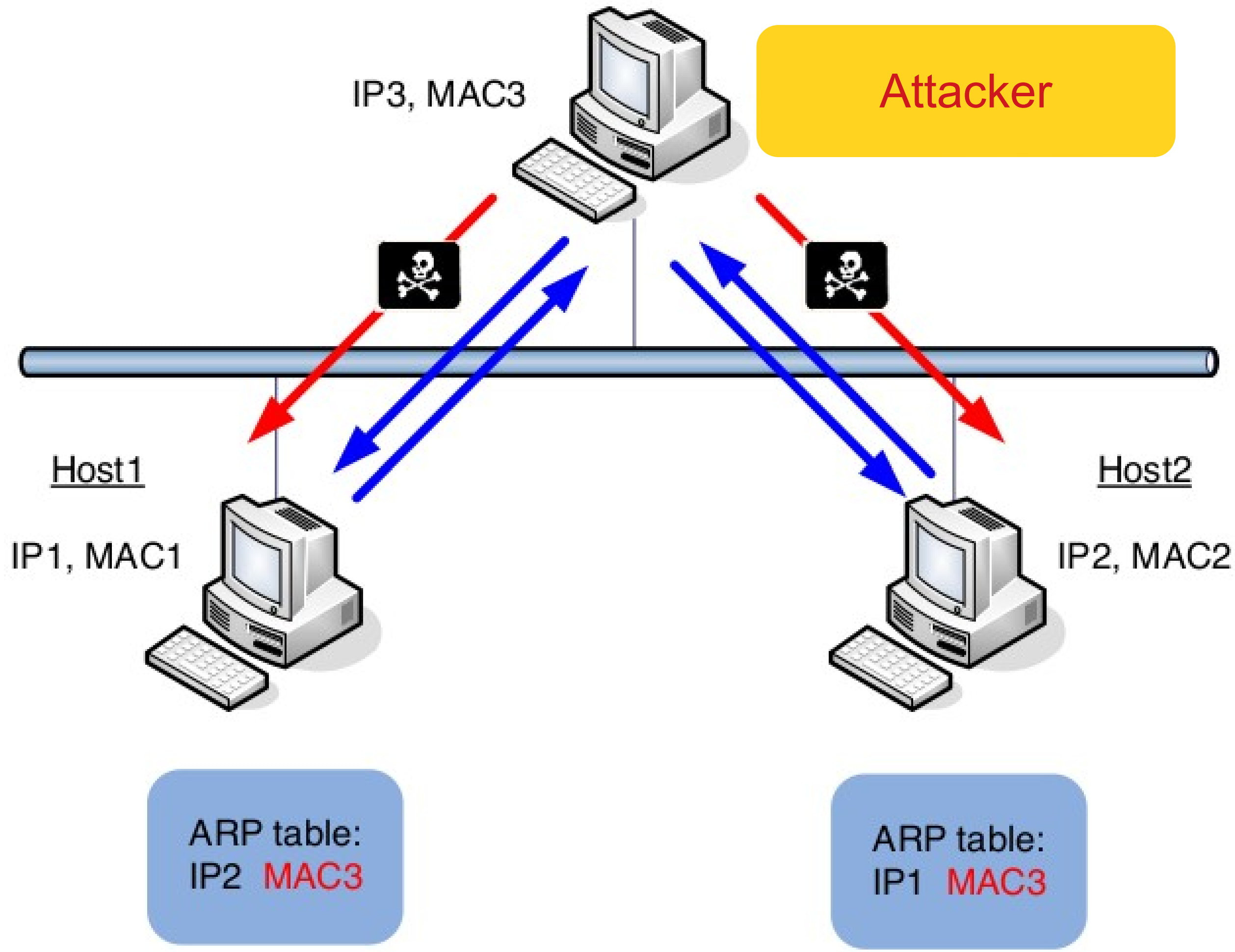
- With an ARP spoofing you can pretend to be anybody…
  - One of the host in the network
  - The default gateway
  - The DNS
  - …
- First level of attack: denial of service
- More interesting, you can launch a MITM attack
  - Intercept the traffic and reroute it to get the reply, then forward the reply back… In the meanwhile, sniff/forge/alter
    - What about SSH/SSL?

Man-in-the-middle 1





Man-in-the-middle 2



Lab activity

Main tasks

- Network eavesdropping
- ARP poisoning
  - MITM
- Reference links:
  - https://www.bettercap.org/intro/
  - https://www.bettercap.org/usage/interactive/

To do the activities

- We will use Kathará (formerly known as netkit)
  - A container-based framework for experimenting computer networking: http://www.kathara.org/
- A virtual machine is made ready for you
  - https://drive.google.com/file/d/12w2wwdFo7jmokVxDWlUdpVWDgf4g8sRe/view?usp=sharing
- For not-Cybersecurity students, please have a look at the Network Infrastructure Lab material
  - http://stud.netgroup.uniroma2.it/~marcos/network_infrastructures/current/cyber/
    - Instructions are for netkit, we will use kathara

The kathara VM

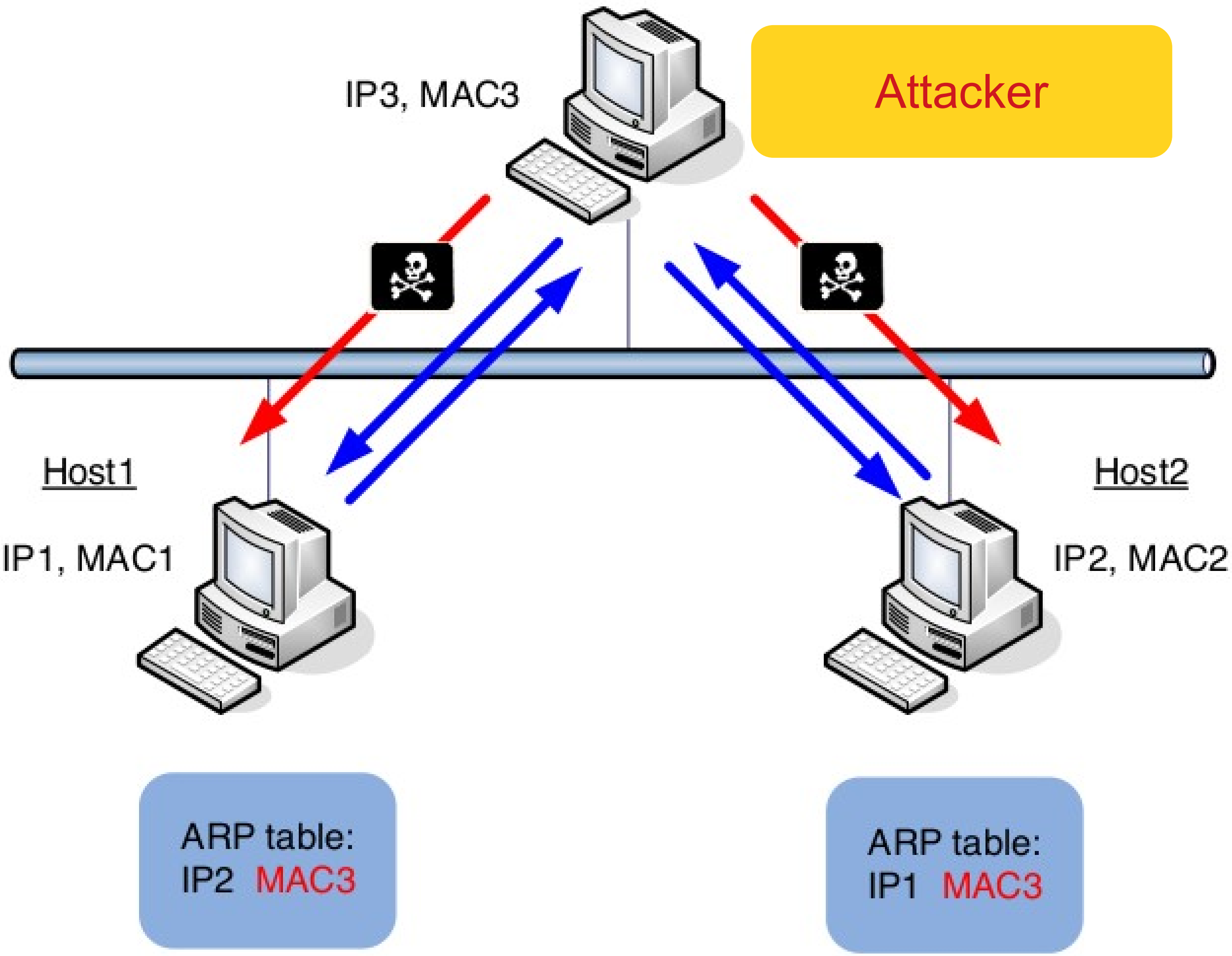
- It <u>should</u> work in both Virtualbox and VMware
- It <u>should</u> work in Linux, Windows and MacOS
- There are some alias (shortcuts) prepared for you
  - Check with **`alias`**
- All the exercises can be found in the git repository:
  - https://github.com/vitome/pnd-labs.git
- You can move in the directory and run lstart
  - **NOTE**: launch docker first or the first lstart attempt can (…will…) fail

Lab activity

Exercise 1: pnd-labs/lab3/ex2

- A lan with a host, a server, a router and the attacker machine.
- In the attacker machine you have to install bettercap and perform a MITM attack. Bettercap does this with the ARP poisoning
  - https://www.cyberpunk.rs/bettercap-usage-examples-overview-custom-setup-caplets
- Set up the links as for ex1, so that the victim machine can be the hosting box (assign it 192.168.100.200)
- Verify the arp poisoning is effective
- Try to use the proxy script included in the folder to alter the image in the server s1
  - kittens → jollypwn

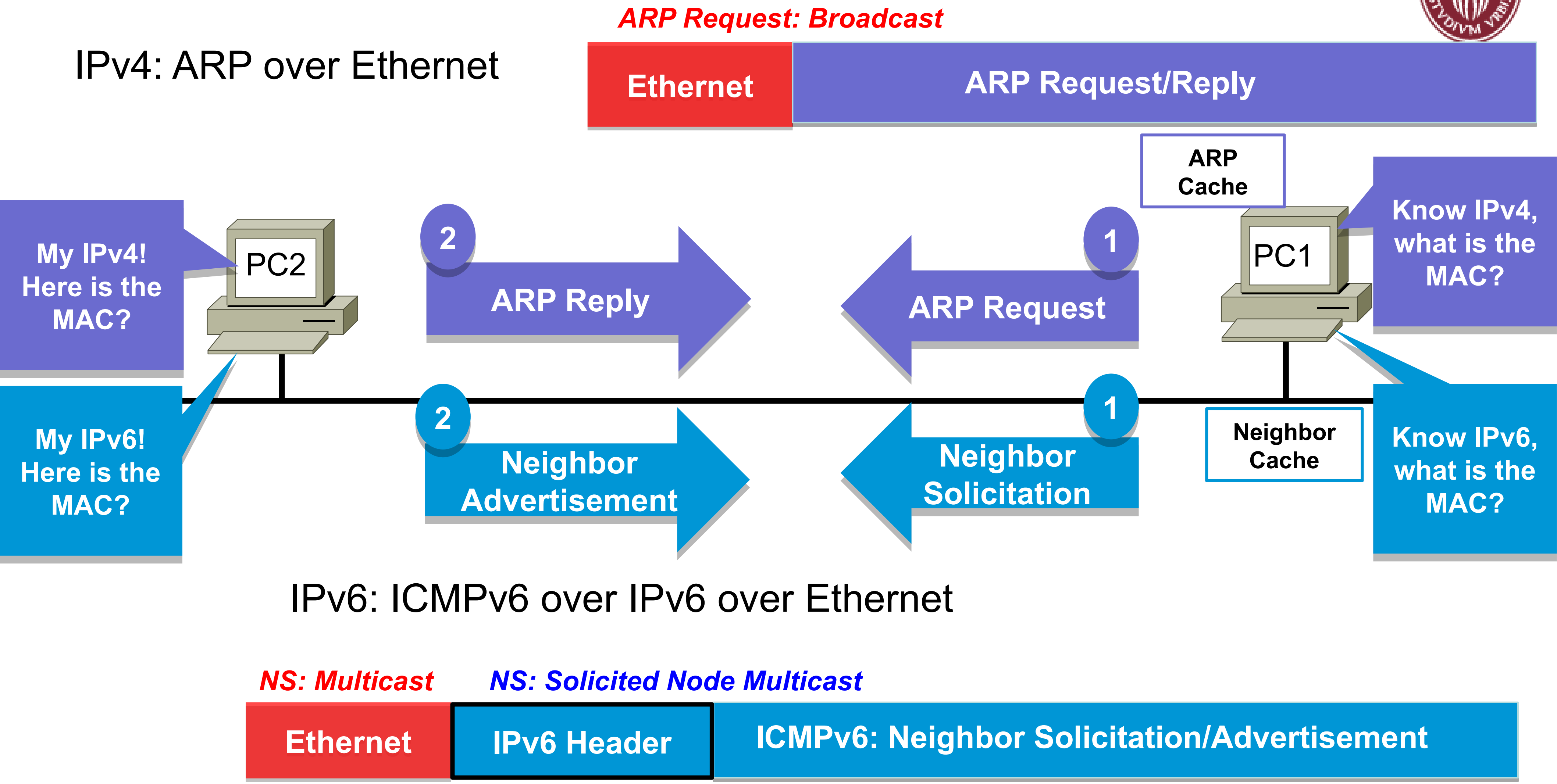
Man-in-the-middle with ARP spoofing



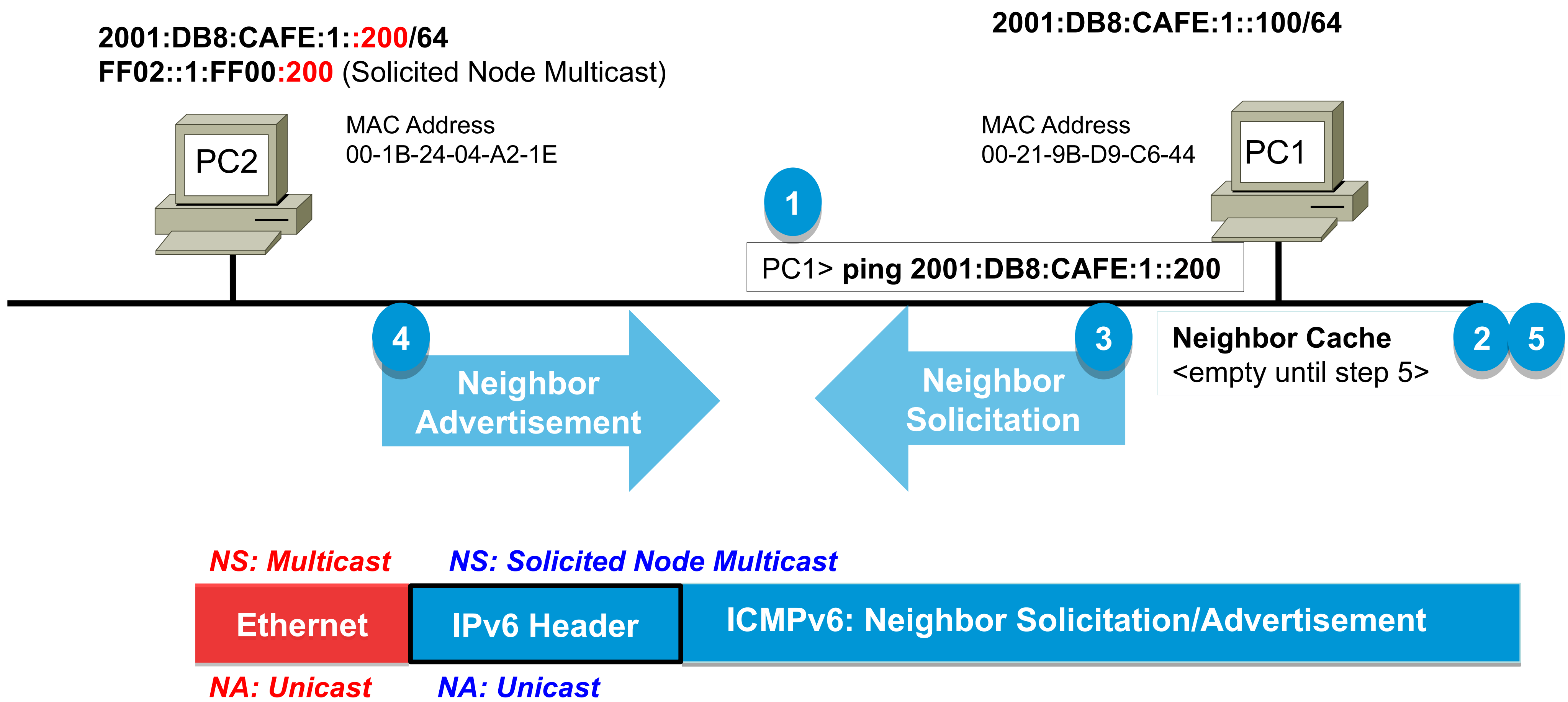
IPv6 Neighbor Discovery issues

Address Resolution: IPv4 and IPv6

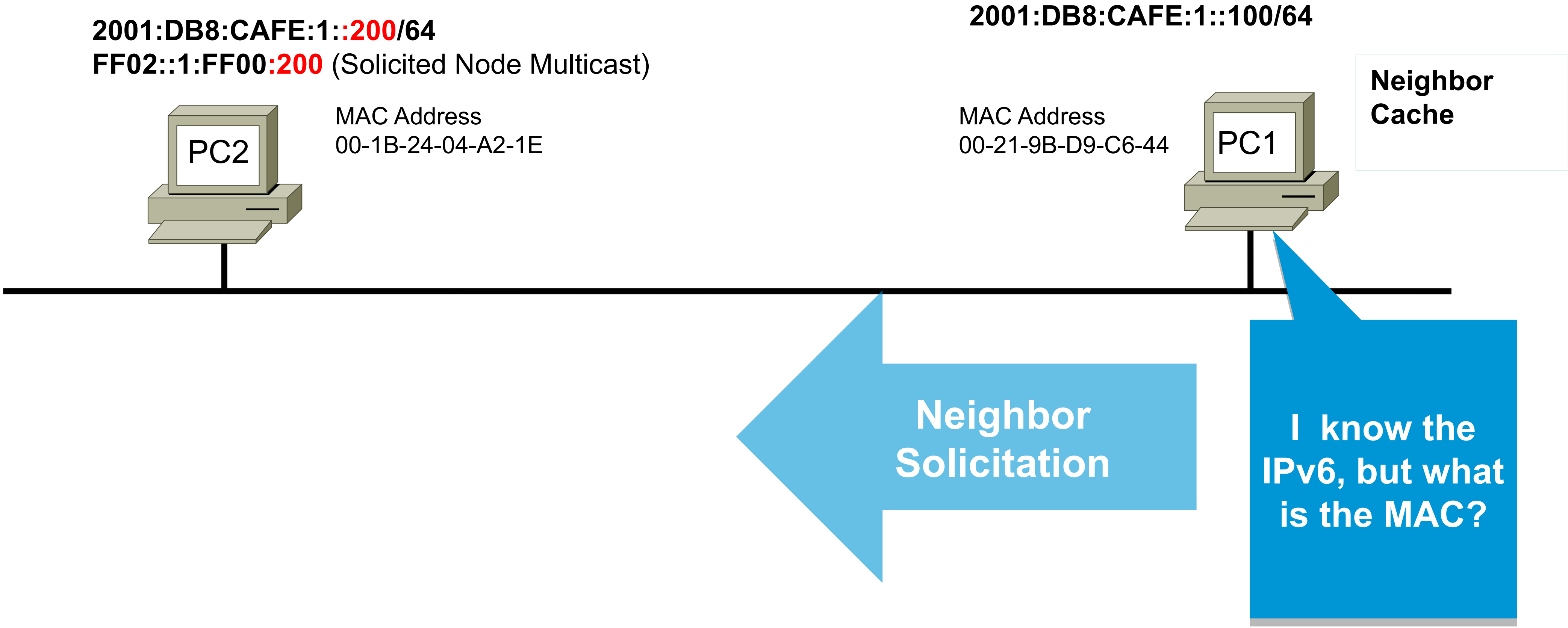
Address Resolution: IPv4 and IPv6



Neighbor Solicitation and Neighbor Advertisement



Neighbor Solicitation



PC1
NS

```
Ethernet II, Src: 00:21:9b:d9:c6:44, Dst: 33:33:ff:00:02:00

Internet Protocol Version 6
    0110 .... = Version: 6
    .... 0000 0000 .... .... .... .... .... = Traffic class: 0x00000000
    .... .... .... 0000 0000 0000 0000 0000 = Flowlabel: 0x00000000
    Payload length: 32
    Next header: ICMPv6 (0x3a)
    Hop limit: 255
    Source: 2001:db8:cafe:1::100
    Destination: ff02::1:ff00:200

Internet Control Message Protocol v6
    Type: 135 (Neighbor solicitation)
    Code: 0
    Checksum: 0xbbab [correct]
    Reserved: 0 (Should always be zero)
    Target: 2001:db8:cafe:1::200
    ICMPv6 Option (Source link-layer address)
        Type: Source link-layer address (1)
        Length: 8
        Link-layer address: 00:21:9b:d9:c6:44
```

Mapped multicast address for PC2

Next header is an ICMPv6 header

Global unicast address of PC1

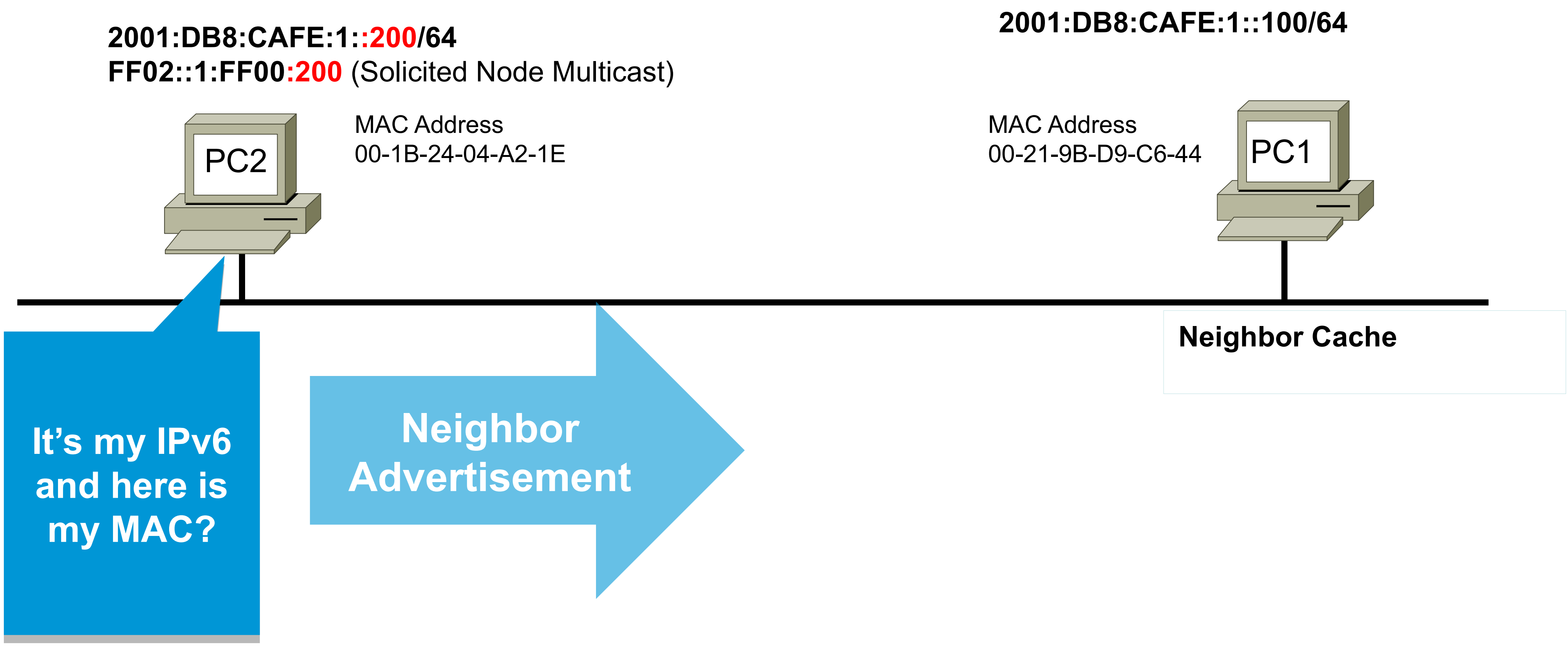
Solicited-node multicast address of PC2

Neighbor Solicitation message

Target IPv6 address, needing MAC address (if two devices have the same solicited node address, this resolves the issue)

MAC address of the sender, PC1

Neighbor Advertisement



PC2
NA

```
Ethernet II, Src: 00:1b:24:04:a2:1e, Dst: 00:21:9b:d9:c6:44

Internet Protocol Version 6
    0110 .... = Version: 6
    .... 0000 0000 .... .... .... .... .... = Traffic class: 0x00000000
    .... .... .... 0000 0000 0000 0000 0000 = Flowlabel: 0x00000000
    Payload length: 32
    Next header: ICMPv6 (0x3a)
    Hop limit: 255
    Source: 2001:db8:cafe:1::200
    Destination: 2001:db8:cafe:1::100

Internet Control Message Protocol v6
    Type: 136 (Neighbor advertisement)
    Code: 0
    Checksum: 0x1b4d [correct]
    Flags: 0x60000000
    Target: 2001:db8:cafe:1::200
    ICMPv6 Option (Target link-layer address)
        Type: Target link-layer address (2)
        Length: 8
        Link-layer address: 00:1b:24:04:a2:1e
```

Unicast MAC address of PC1

Next header is an ICMPv6 header

Global unicast address of PC2

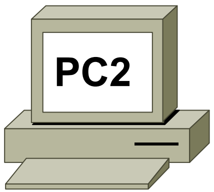
Global unicast address of PC1

Neighbor Advertisement message

IPv6 address of the sender, PC2

MAC address of the sender, PC2

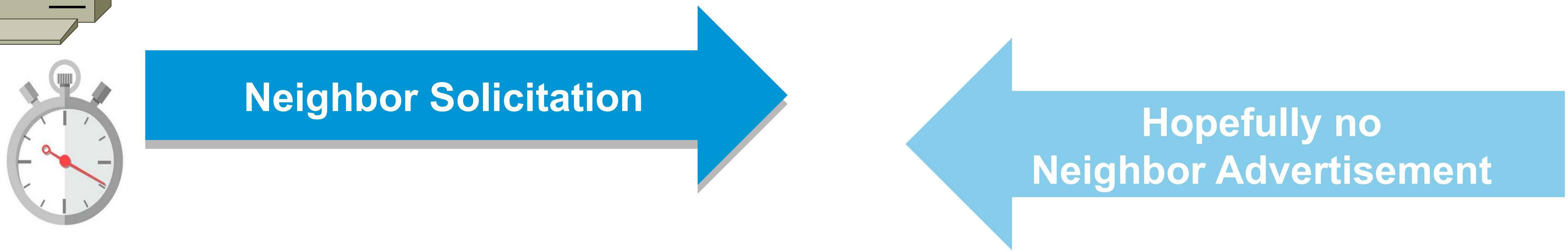
ICMPv6 Duplicate Address Detection (DAD)



Global Unicast - 2001:DB8:CAFE:1::200
Link-local - FE80::1111:2222:3333:4444

See the process with:

```
R1# debug ipv6 nd
```



- Duplicate Address Detection (DAD) is used to guarantee that an IPv6 unicast address is unique on the link.
- A device will send a Neighbor Solicitation for its own unicast address (static or dynamic).
- After a period of time, if a NA is not received, then the address is deemed unique.
- Once required, RFC was updated to where it is only recommended - /64 Interface ID makes duplicates unlikely!

ND threats (RFC 3756)

- Non router/routing related threats
  - Neighbor Solicitation/Advertisement Spoofing
  - Neighbor Unreachability Detection (NUD) failure
  - Duplicate Address Detection DoS Attack
- Router/routing involving threats
  - Malicious Last Hop Router
  - Default router compromise
  - Spoofed Redirect Message
  - Bogus On-Link Prefix
  - Bogus Address Configuration Prefix
  - Parameter Spoofing
- Replay attacks
- Neighbor Discovery DoS Attack

Some examples: DAD DoS

- Step 1, Host: Can I use IPv6 address AA:BB::CC?
- Step 2, Attacker: No the address is used!
- Step 3, Host: Can I use IPv6 address AA:BB::DD?
- Step 4, Attacker: No the address is used!
- …
- Step X, Attacker: No the address is used!
  - dos-new-ip6 from thc-toolkit (https://github.com/vanhauser-thc/thc-ipv6 The Hackers' Choice)

IPv6 Router Advertisement issues

Some examples: Rogue RA

- What happens when an IPv6 enabled system receives a router advertisement?
  - If SLAAC enabled, it will be part of another network and will receive a new route, optionally a default gateway…
- Common problem: VPN bypass (tunnel split)
- In the end, with control of DNS and IPv6, the attacker can
  - sniff all client traffic
  - attempt Man-In-The-Middle attacks
  - impersonate servers/systems and capture presented user credentials (e.g. NTLM)
  - gain access into the other networks of the system

Some examples: RA flooding

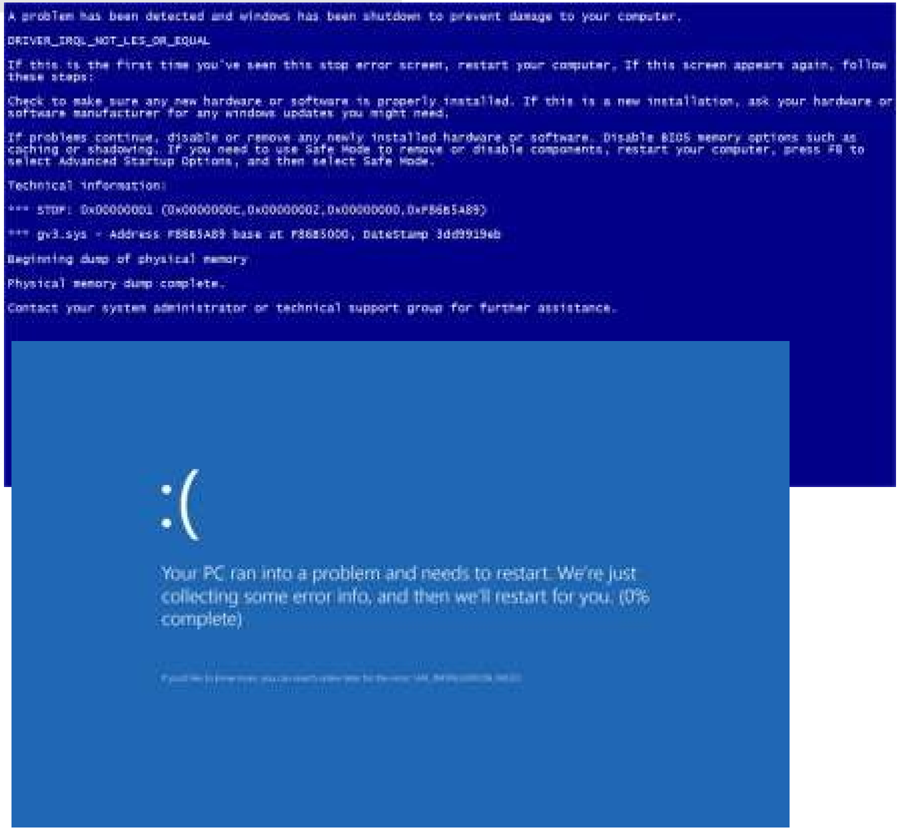
- Flooding IPv6 hosts with Router Advertises
  - flood_router6 from thc-toolkit (https://github.com/vanhauser-thc/thc-ipv6 The Hackers' Choice)
- Old OSes (Windows Vista/7/8) boxes frozen
- Other platforms had severe problems with IPv6 connectivity
- More info:
  - http://samsclass.info/ipv6/proj/flood-router6a.htm

RA flooding, effects…

```
Ethernet II, Src: WistronI_59:61:8b (3c:97:0e:59:61:8b), Dst: IPv6mcast_00:00:00:01 (33:33:00:00:00:01)
Internet Protocol Version 6, Src: fe80::76:a3e9:7636:3901 (fe80::76:a3e9:7636:3901), Dst: ff02::1 (ff02::1)
Internet Control Message Protocol v6
  Type: Router Advertisement (134)
  Code: 0
  Checksum: 0x0fff [correct]
  Cur hop limit: 255
⊞ Flags: 0x08
  Router lifetime (s): 65535
  Reachable time (ms): 16384000
  Retrans timer (ms): 1966080
⊞ ICMPv6 Option (MTU : 1500)
⊞ ICMPv6 Option (Source link-layer address : 00:0c:e9:76:36:39)
⊞ ICMPv6 Option (Prefix information : 2012:76a4:ea76:3639::/64)
⊞ ICMPv6 Option (Prefix information : 2012:76a5:ec76:3639::/64)
⊞ ICMPv6 Option (Prefix information : 2012:76a6:ee76:3639::/64)
⊞ ICMPv6 Option (Prefix information : 2012:76a7:f076:3639::/64)
```

(Lots of Prefix/Route Information options omitted…)

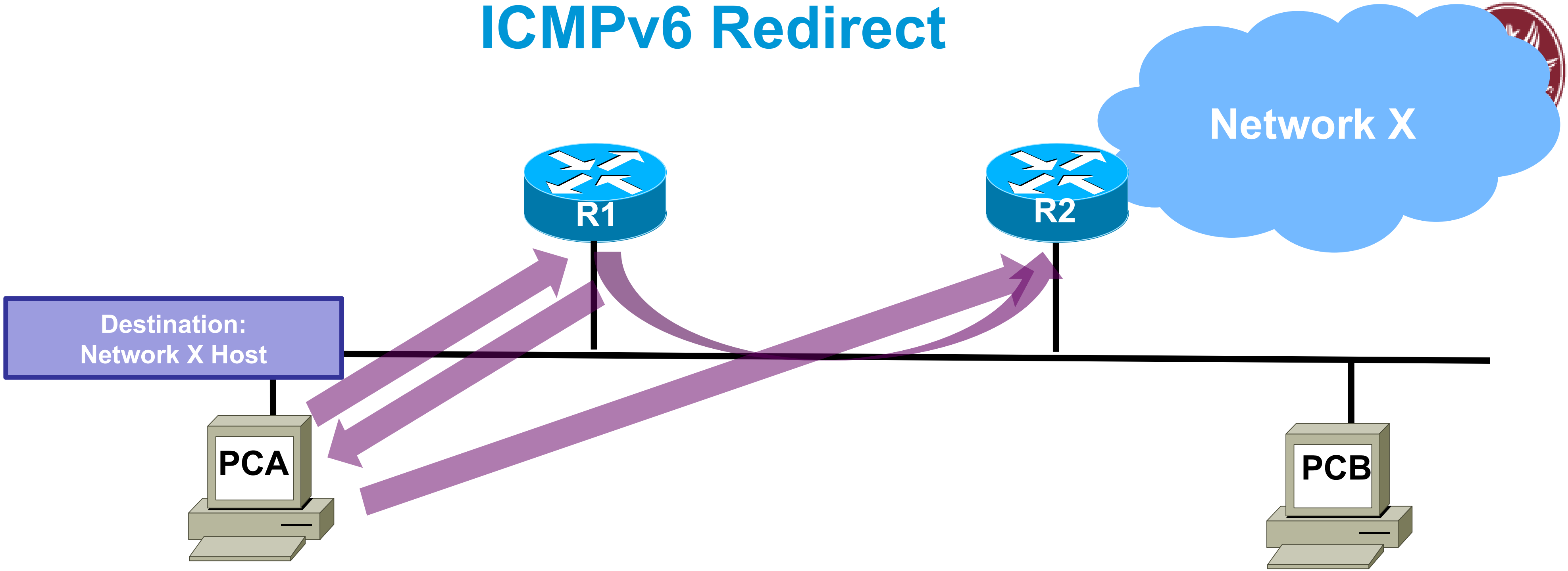
```
⊞ ICMPv6 Option (Route Information : High 2004:76be:fd76:3639::/64)
⊞ ICMPv6 Option (Route Information : High 2004:76bf:ff76:3639::/64)
⊞ ICMPv6 Option (Route Information : High 2004:76c0:177:3639::/64)
⊞ ICMPv6 Option (Route Information : High 2004:76c1:377:3639::/64)
⊞ ICMPv6 Option (Route Information : High 2004:76c2:577:3639::/64)
```



DHCP rogue server (DHCP starvation)

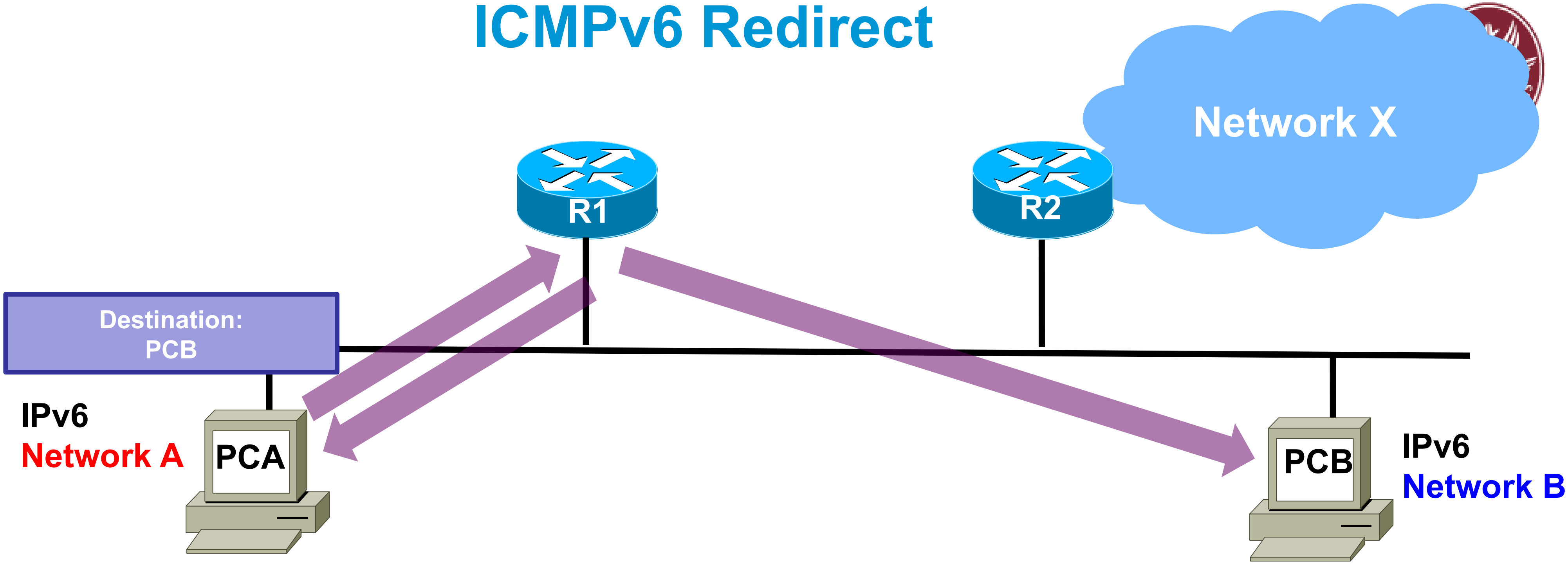
- Anyplace where macof works, you can DoS a network by requesting all of the available DHCP addresses
- Once the addresses are gone, an attacker could use a rogue DHCP server to provide addresses to clients
- Since DHCP responses include DNS servers and default gateway entries, the attacker can PRETEND to be anyone…
- All the MITM attacks are now possible
- Mitigations are mainly based on enabling switch security capabilities:
  - RFC7610 F. Gont, W. Liu, G. Van de Velde, "DHCPv6-Shield: Protecting against Rogue DHCPv6 Servers", August 2015, Best Current Practice
  - DHCP snooping (see before)
  - Dynamic ARP inspection (see before)
  - IEEE 802.1x

ICMPv6 Redirect



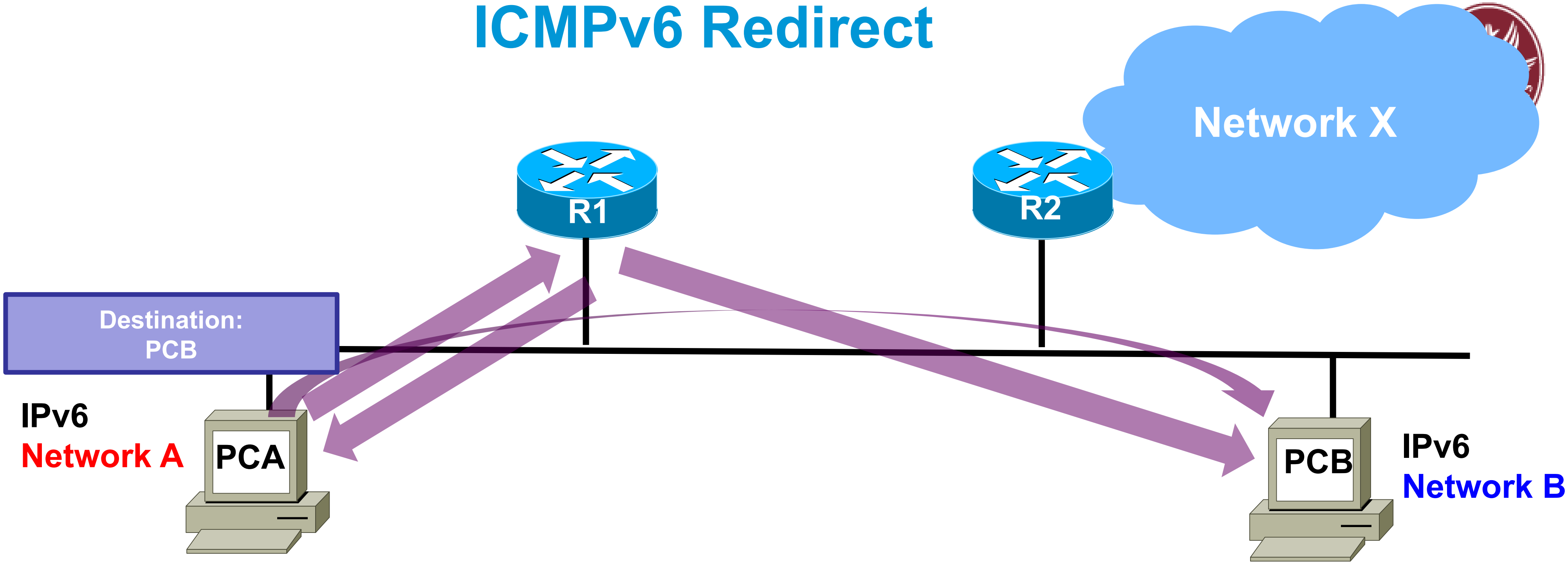
- Similar functionality as ICMPv4.
- Like IPv4, a router informs an originating host of the IP address of a router that is on the local link and is closer to the destination.

ICMPv6 Redirect



- Similar functionality as ICMPv4.
- Like IPv4, a router informs an originating host of the IP address of a router that is on the local link and is closer to the destination.
- Unlike IPv4, a router informs an originating host that the destination host (on a different prefix/network) is on the same link as itself.

ICMPv6 Redirect



- Similar functionality as ICMPv4.
- Like IPv4, a router informs an originating host of the IP address of a router that is on the local link and is closer to the destination.
- Unlike IPv4, a router informs an originating host that the destination host (on a different prefix/network) is on the same link as itself.

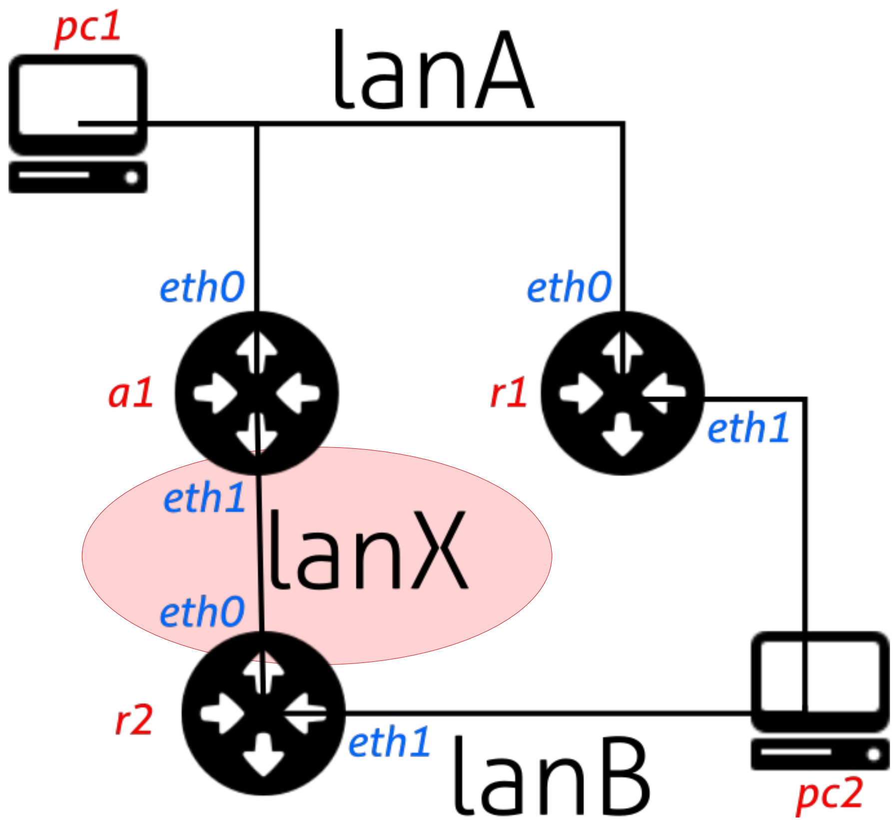
Lab activity: ex4

IPv6 Attack Tools

- THC IPv6 Attack Toolkit – parasite6, alive6, fake_router6, redir6, toobig6, detect-new-ip6, dos-new-ip6, fake_mld6, fake_mipv6, fake_advertiser6, smurf6, rsmurf6
  - https://github.com/vanhauser-thc/thc-ipv6
- SI6 Networks' IPv6 Toolkit – flow6, frag6, icmp6, jumbo6, na6, ni6, ns6, ra6, rd6, rs6, scan6, tcp6
  - https://github.com/fgont/ipv6toolkit
- Already in the kathara environment
  - Just run `make` in the `thc-ipv6-3.6` directory of the attacker machine

Exercise 2: pnd-labs/lab3/ex4

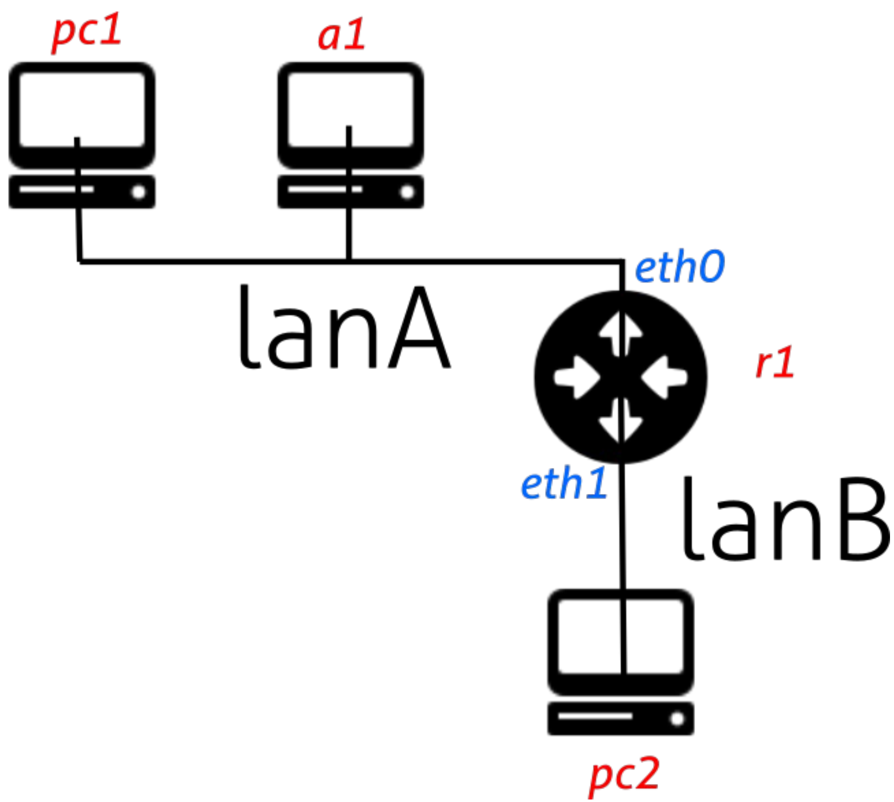
- PC1 reaches PC2 via r1
- The assignment is to use ICMP redirect to hijack the traffic from pc1 and capture the traffic in **lanX**
- Observe the type of packet exchange of a1
- The tool to be used is **`redir6`**



Lab activity: ex5

Exercise 3: pnd-labs/lab3/ex5

- PC1 reaches PC2 via r1
- The assignment is to use ICMP redirect to convince pc1 that a1 is a best hop for pc2
- Observe the type of packet exchange of a1 using wireshark
- The tool to be used is **`redir6`**



Lesson learned: reject redirects!

- If you check the default parameters:
  - /proc/sys/net/ipv4/conf/all/accept_redirects
    - TRUE (host)
    - FALSE (router)
  - /proc/sys/net/ipv4/conf/all/secure_redirects
    - TRUE
  - /proc/sys/net/ipv4/conf/all/shared_media
    - TRUE
  - /proc/sys/net/ipv6/conf/all/accept_redirects
    - Functional default: enabled if local forwarding is disabled
    - disabled if local forwarding is enabled.
- Then: accept_redirects and alike → FALSE
- Try the patch on the labs and see the effects

That’s all for today

- **Questions?**
- IPv6 security references: https://www.ripe.net/support/training/material/ipv6-security/ipv6security-references.pdf
  - TCP-IP guide
    - http://www.tcpipguide.com/free/t_InternetProtocolVersion6IPv6IPNextGenerationIPng.htm
  - IPv6 Dissemination and Exploitation (6diss) European project
    - http://6diss.6deploy.eu/
  - Cabrillo’s publications
    - http://www.cabrillo.edu/~rgraziani/ipv6-presentations.html
  - RIPE NCC Academy: *IPv6 Fundamentals (free course)*
    - https://academy.ripe.net/course/view.php?id=13
  - Book chapter 11 (even if quite obsoleted)