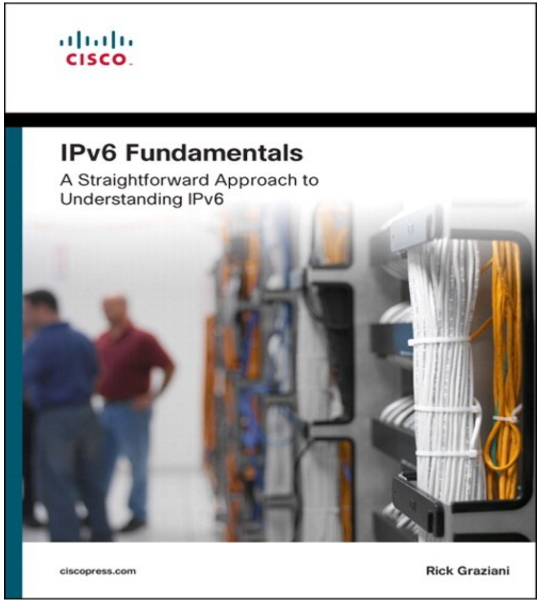
Practical Network Defense

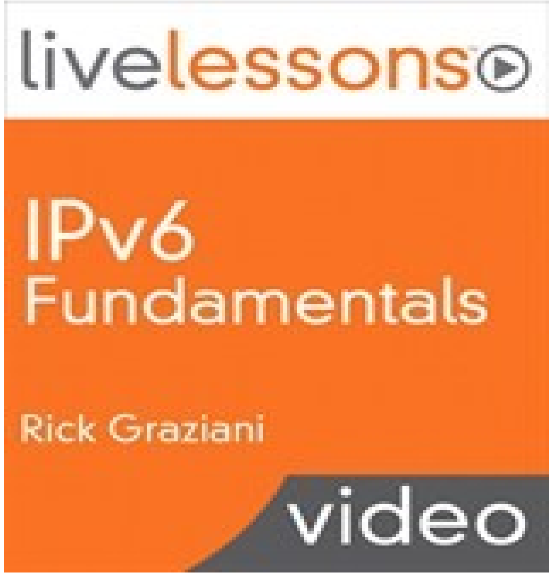
Master's degree in Cybersecurity 2020-21

IPv6: addressing and ICMPv6 lab

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

Material taken from Rick Graziani IPv6 courses





Recap last lectures

- IPv6 address types
  - Global Unicast Address
  - Local-link Unicast Address
- IPv6 dynamic assignment options
- Multicast Addresses
  - Permanent addresses (“well known multicast groups”)
  - Scope of multicast addresses
- IPv6 packet header
- IPv6 Extension headers

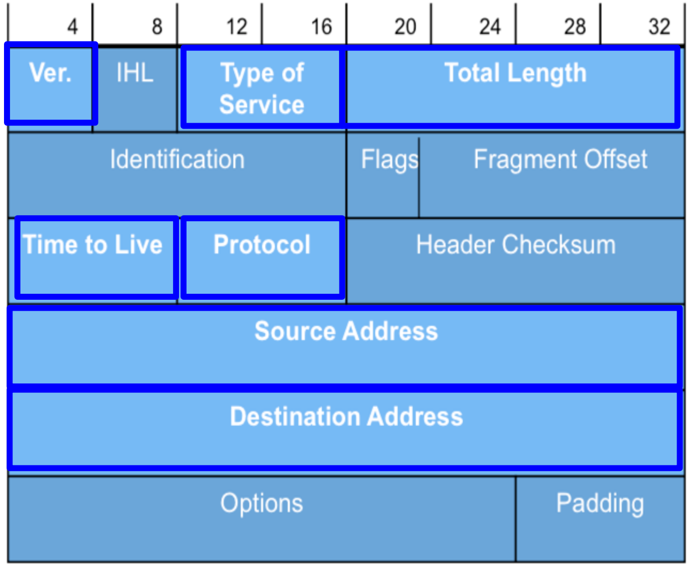
IPv6 header

IPv6 Header

Similar fields

- Understanding IPv6 begins with the IPv6 header.
- IPv6 takes advantage of 64-bit CPUs.
- Several differences between IPv4 and IPv6 headers.

IPv4

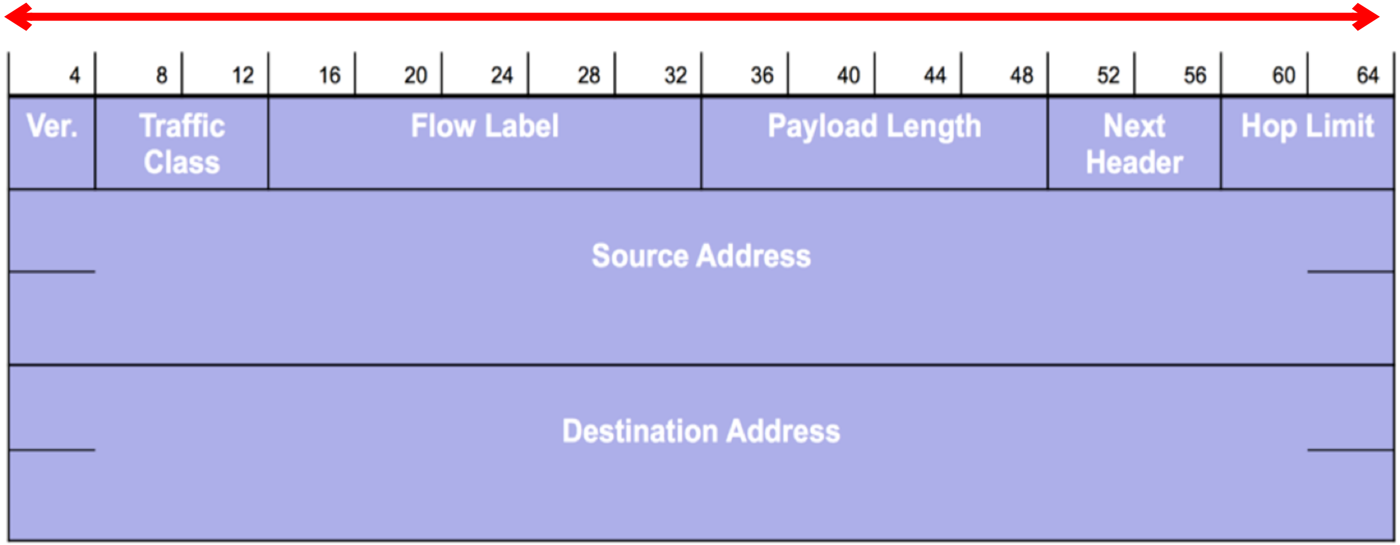


4	8	12	16	20	24	28	32
Ver.	IHL	Type of Service		Total Length			
Identification				Flags	Fragment Offset		
Time to Live		Protocol		Header Checksum			
Source Address							
Destination Address							
Options						Padding	

- Simpler IPv6 header.
- Fixed 40 byte IPv6 header.
- Lets look at the differences…

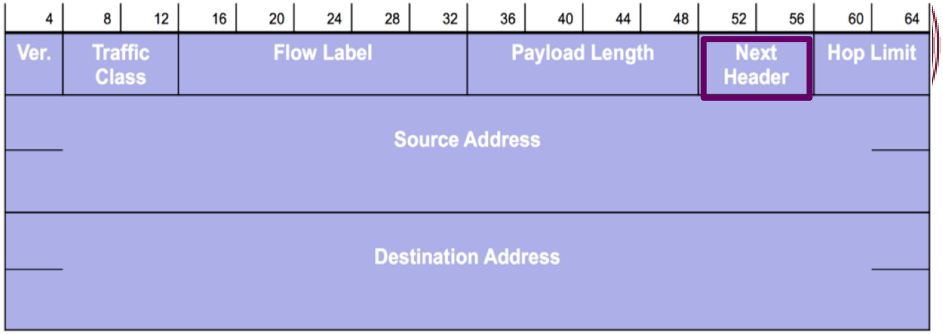
IPv6

64-bit memory word

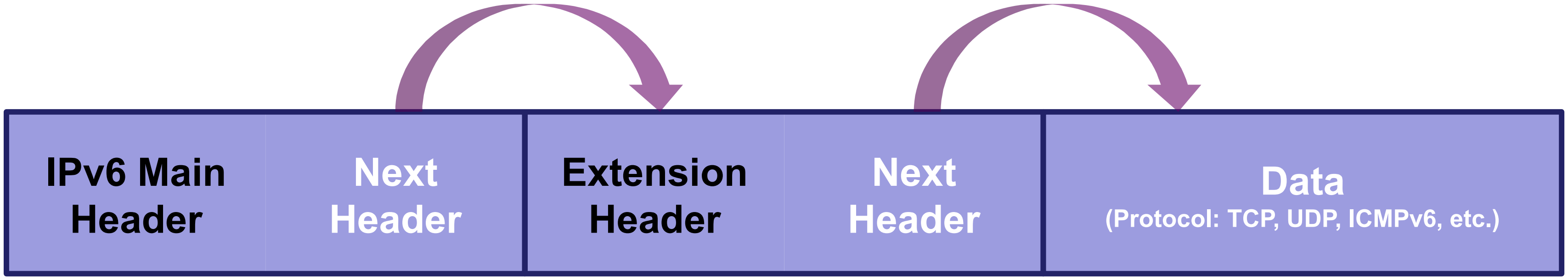


4	8	12	16	20	24	28	32	36	40	44	48	52	56	60	64
Ver.	Traffic Class		Flow Label					Payload Length				Next Header		Hop Limit	
Source Address															
Destination Address															

IPv6 Extension Header



- **Next Header** identifies:
  - The protocol carried in the data portion of the packet.
  - The presence of an extension header.
- **Extension headers** are optional and follow the main IPv6 header.
- Provide flexibility and features to the main IPv6 header for future enhancements without having to redesign the entire protocol.
- Allows the main IPv6 header to have a fixed size for more efficient processing.



IPv6 Main Header

Next Header

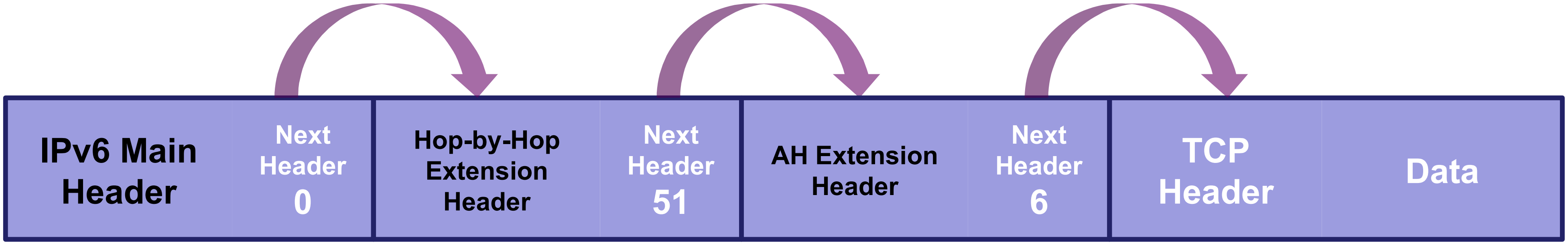
Extension Header

Next Header

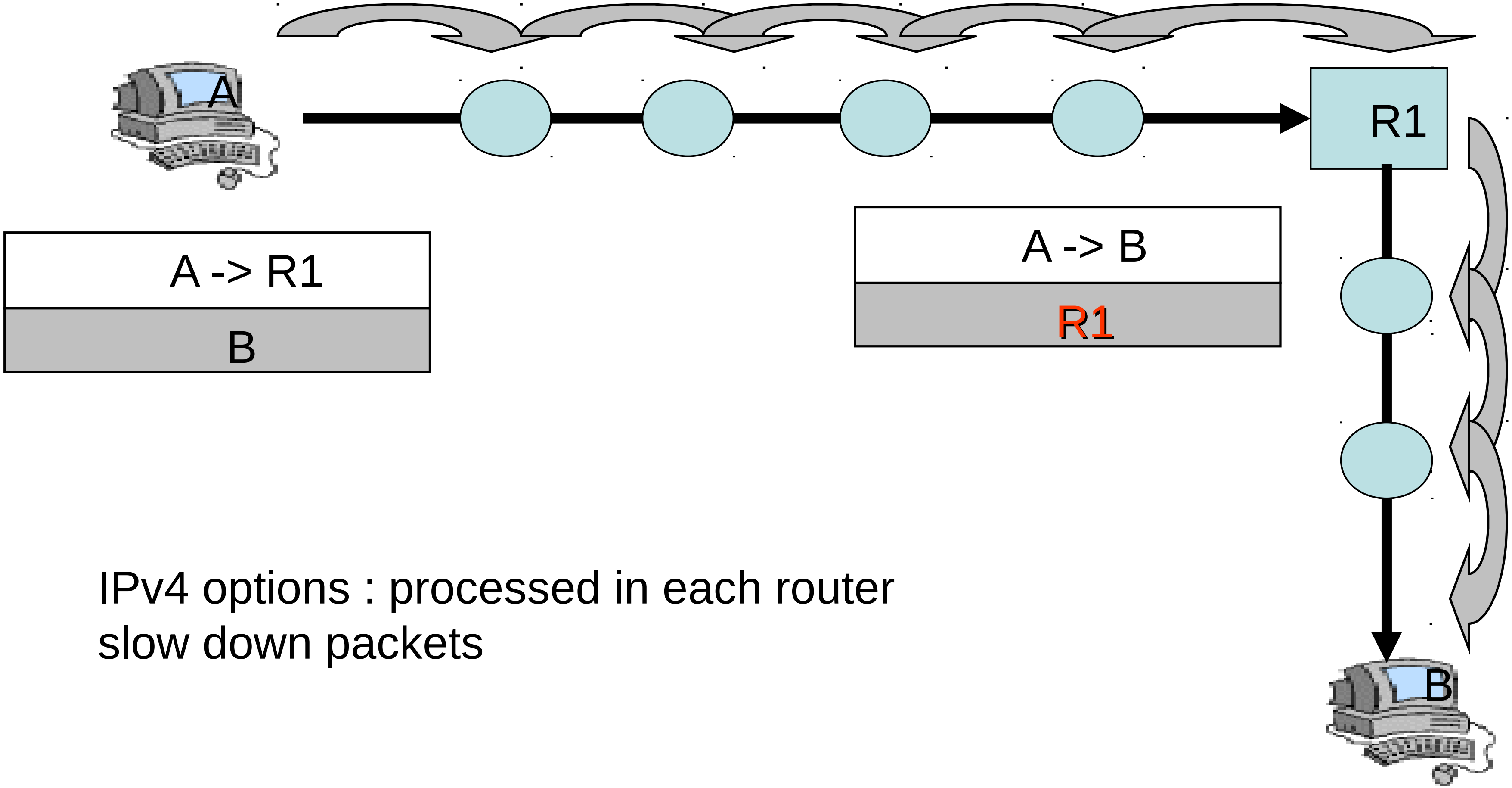
Data
(Protocol: TCP, UDP, ICMPv6, etc.)

IPv6 Extension Header

Next Header Value (Decimal)	Extension Header Name	Extension Header Description
0	Hop-by-Hop Options	Used to carry optional information, which must be examined by every router along the path of the packet.
43	Routing	Allows the source of the packet to specify the path to the destination.
44	Fragment	Used to fragment IPv6 packets.
50	Encapsulating Security Payload (ESP)	Used to provide authentication, integrity, and encryption.
51	Authentication Header (AH)	Used to provide authentication and integrity.
60	Destination Options	Used to carry optional information that only needs to be examined by a packet's destination node(s).

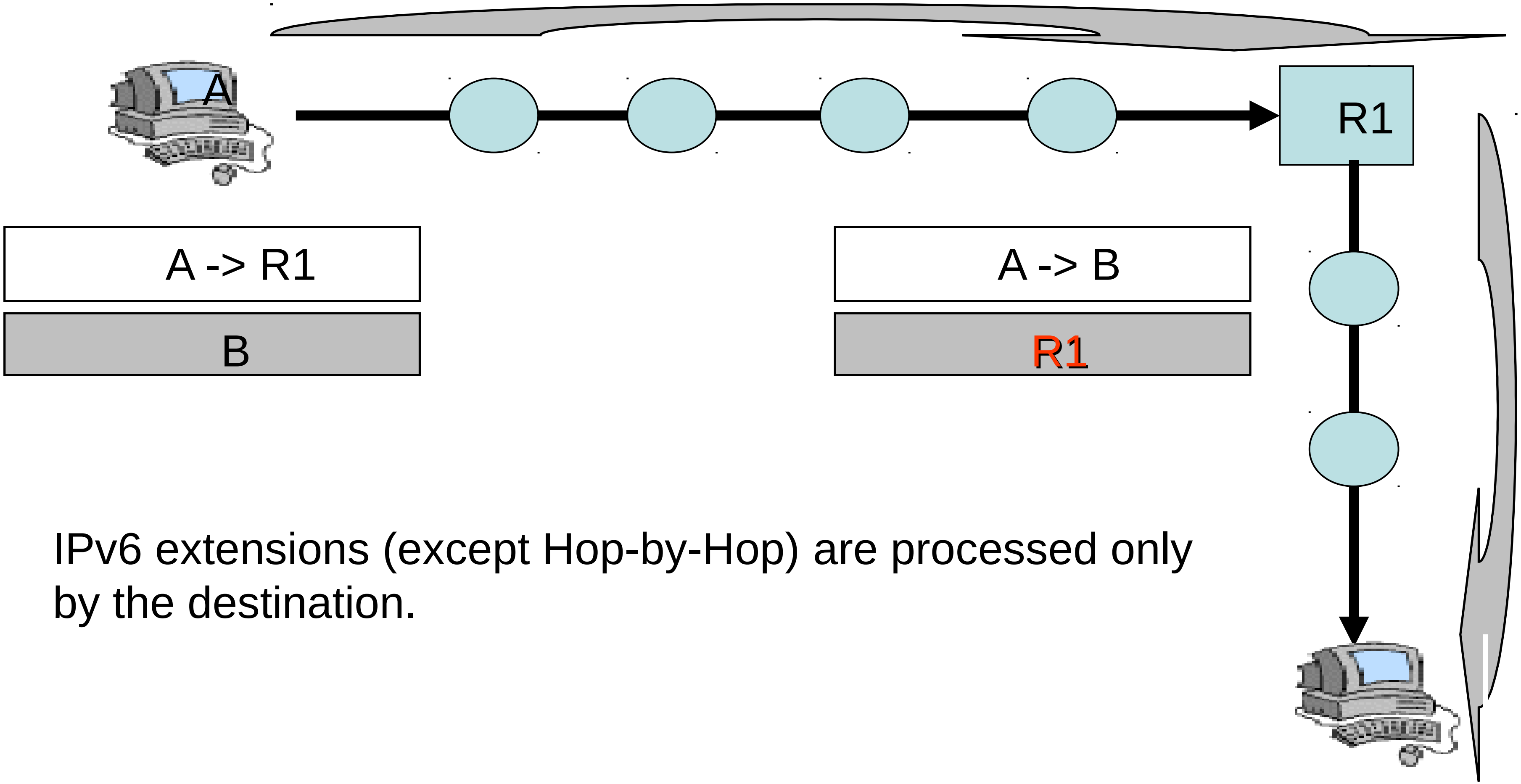


IPv4 options vs. IPv6 extensions



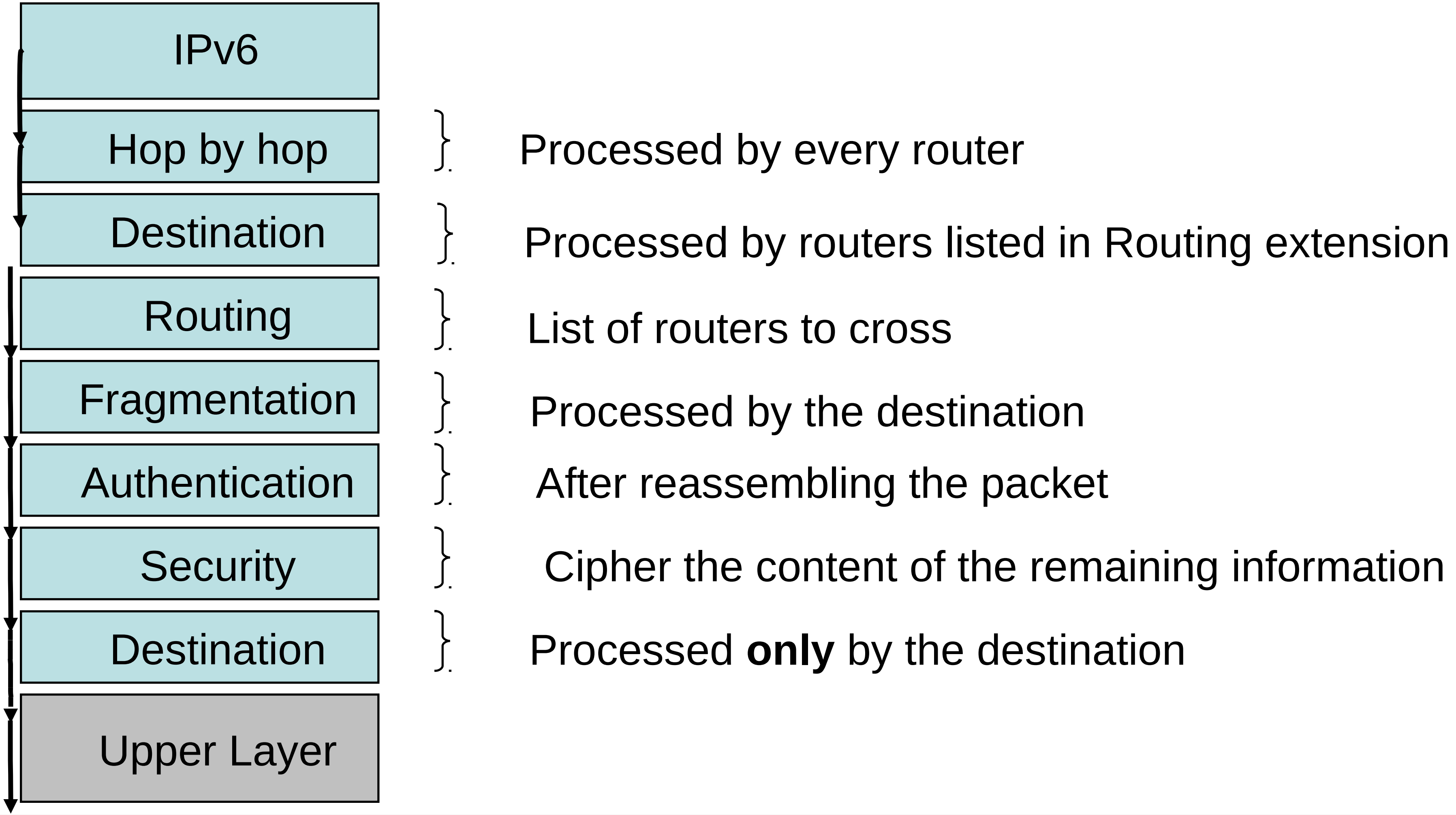
IPv4 options : processed in each router
slow down packets

IPv4 options vs. IPv6 extensions



IPv6 extensions (except Hop-by-Hop) are processed only by the destination.

Order is important (RFC 2460)



IPv6

Hop by hop — Processed by every router

Destination — Processed by routers listed in Routing extension

Routing — List of routers to cross

Fragmentation — Processed by the destination

Authentication — After reassembling the packet

Security — Cipher the content of the remaining information

Destination — Processed **only** by the destination

Upper Layer

Lab activity

Main tasks

- DHCPv6 with prefix delegation
- ICMPv6 MTU discovery
  - With ping and tracepath

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/1W6JQzWVyH5_LKLD20R6XH1ugPDP5LWP5/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 should work in both Virtualbox and VMware
- It should 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
  - DON'T FORGET TO UPDATE → `~/pnd-labs$ git pull`
- You can move in the directory and run lstart
  - **NOTE**: launch docker first or the first lstart attempt can (…will…) fail

Lab activity: ex4

Exercise 4: pnd-labs/lab2/ex4

DHCPv6 with prefix delegation

- One router with two lan, both with 2 pcs. The router is connected with an ISP router.
- TASK: configure the topology to use IPv6 addresses
  - The ISP makes use of a DHCPv6 server for address and prefix distribution
  - The router has to ask prefixes to its ISP and has to distribute addresses inside the two lans, using SLAAC.
    - At least two options:
      - dibbler DHCPv6 client + radvd
      - wide-dhcp + dnsmasq
- The ISP is already configured to provide prefixes, while the router and the pcs have to be configured.
  - the router has always 1 in the host part of its own link local address

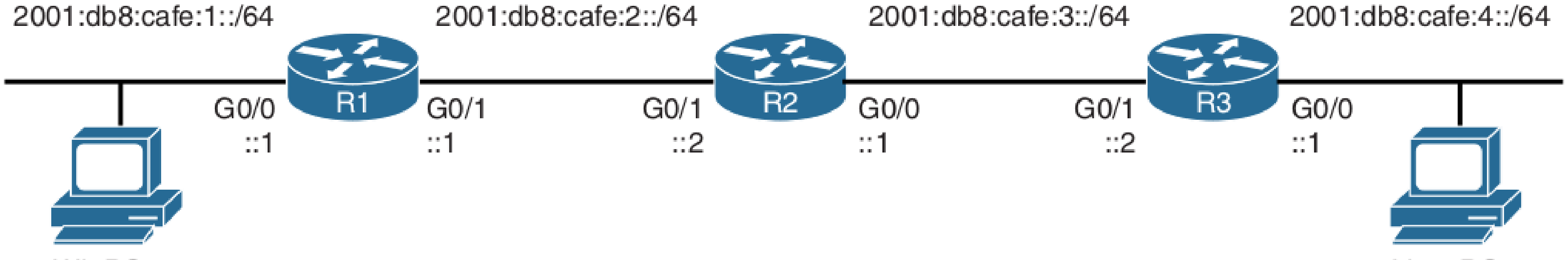
DHCPv6-PD: Reference links

- Linux ipv6 configuration: ipv6 sysctl
  - https://www.kernel.org/doc/Documentation/networking/ip-sysctl.txt
- DHCPv6:
  - dibbler+radvd:
    - useful guide: https://k3a.me/setting-up-ipv6-using-a-dhcp-client/
    - man pages:
      - https://manpages.debian.org/testing/radvd/radvd.conf.5.en.html
      - https://klub.com.pl/dhcpv6/doc/dibbler-user.pdf
  - wide-dhcp+dnsmasq:
    - useful guide: https://github.com/torhve/blag/blob/master/using-dnsmasq-for-dhcpv6.md
    - man pages:
      - https://thekelleys.org.uk/dnsmasq/docs/dnsmasq-man.html
      - https://manpages.debian.org/stretch/wide-dhcpv6-client/dhcp6c.8.en.html
      - https://manpages.debian.org/stretch/wide-dhcpv6-client/dhcp6c.conf.5

Lab activity: ex5

Exercise 5: pnd-labs/lab2/ex5

- Three routers connecting two lans with one pc each.
- Configure the topology to use static addressing for the routers and SLAAC IPv6 addresses for the two lans. See the README file for the details.
- Moreover, you have to play with the MTU of the links between the routers to generate and capture ICMPv6 packets (Packet too big or MTU discovery).
- You have to use tracepath and ping to test connectivity and MTU
- You can use the **`ip link set mtu XXXX dev YYY`** on both the ent points of a link to alter the MTU



Lab activity: ex6

Exercise 6: create an IPv6 capable connection

- The task is to create an virtual interface for providing capable IPv6 Internet connection
  - IPv6 native: the entire infrastructure supports IPv6
    - Namely, your ISP provides you IPv6 addresses
  - IPv6 capable: the infrastructure can support IPv6 services and technologies by taking advantage of IPv6 transition technologies
    - Namely, you use a Intra-Site Automatic Tunnel Addressing Protocol (ISATAP)
      - Tunnel IPv6 messages inside an IPv4 header
  - IPv4 only: the infrastructure can not support IPv6
- Reference: https://developers.redhat.com/blog/2019/05/17/an-introduction-to-linux-virtual-interfaces-tunnels/

ISATAP: howto, using hurricane-electric services

- Go to https://www.tunnelbroker.net/ and register
- On the left, select Create regular tunnel
- Setup everything following the form directions
  - You can also refer to https://ipv6.he.net/certification/faq.php or http://ipv6.he.net/presentations.php and http://tunnelbroker.net/forums/
  - Beware if you are in a NAT'd network (this is highly likely)
- Important: your host has to be reachable from outside using protocol 41 → IPv6 Encapsulation (RFC 2473)
  - Virtual server, forward or DMZ in your home router

Steps to follow (sketch)

```
ip tunnel add he-ipv6 mode sit remote 216.66.80.98\
       local 192.168.100.13 ttl 255
ip link set he-ipv6 up
ip addr add 2001:a23f:f25:14c9::2/64 dev he-ipv6
ip route add ::/0 dev he-ipv6
ip -f inet6 addr
```

That’s all for today

- **Questions?**
- References:
  - https://developers.redhat.com/blog/2019/05/17/an-introduction-to-linux-virtual-interfaces-tunnels/
  - http://www.tcpipguide.com/free/t_InternetProtocolVersion6IPv6IPNextGenerationIPng.htm
  - https://www.6diss.org/e-learning/
  - http://www.cabrillo.edu/~rgraziani/ipv6-presentations.html
  - Book chapter 11 (even if quite obsoleted)