

Practical Network Defense

Master's degree in Cybersecurity 2021-22

Course introduction

Angelo Spognardi
spognardi@di.uniroma1.it

*Dipartimento di Informatica
Sapienza Università di Roma*

Practical Info



The lecturer

- Angelo Spognardi
 - Associate professor, Dipartimento di Informatica
 - Tel: 06 4925 5164
 - <https://angelospognardi.site.uniroma1.it>
 - spognardi@di.uniroma1.it
- Student hours:
 - Until COVID emergency: on request by email
 - After: my office G29, v.le Regina Elena, 295, Edificio G
- My research interests:
 - Computer network security, security in social networks, privacy, applied cryptography, operating systems, programming





This course

- Website:
<https://sites.google.com/di.uniroma1.it/netdef2122>
 - Join the class in classroom
classroom.google.com
using this code: 6east2o
 - Use your @studenti.uniroma1.it account (or it won't work...)
- 6 CFUs
- Schedule:
 - Monday 17-19 Aula ??
 - Thursday 12-15 Tiburtina Labs (lab XV)



MASTER'S DEGREE IN CYBERSECURITY

Academic year 2021/2022

Teacher: [Angelo Spognardi](mailto:Angelo.Spognardi@di.uniroma1.it) (spognardi@di.uniroma1.it)



DIPARTIMENTO
DI INFORMATICA
SAPIENZA
UNIVERSITÀ DI ROMA

OVERVIEW

The course explains the fundamentals of the methods and tools for the protection of computer networks. Particular attention is paid to the practical application of the concepts learned: commonly-seen threats arising from the use of particular protocols in networked computer systems, mechanisms commonly used by intruders and designers of malware in order to compromise a computer system's security, the basic mechanisms used for the detection of intrusion attempts in computer systems.



Objectives

- Methods and tools for the protection of computer networks
- Focus on practical application of the concepts learned:
 - protocols in networked computer systems
 - mechanisms commonly used to compromise a computer system's security
 - mechanisms used for the detection of intrusion attempts in computer networks
- At the end of the course students will be able to:
 - monitor traffic in networks
 - apply a security policy
 - perform a network scan and search for vulnerabilities in a computer network
- Students will develop the ability to:
 - select the appropriate firewall rules to protect a network
 - select the most appropriate mechanisms to protect a networked computer system
 - make the most appropriate design choices to implement a "defense in depth" strategy, using isolated networks and dedicated tools (VPN, proxy and firewall)
- Students will learn how to document their choices, also through the use of automated reporting tools. They will also have acquired the ability to prepare presentations related to specific scientific topics



Topics covered (tentative)

- Network monitoring
- Network traffic analysis
- Network attacks (e.g., session hijacking, man-in-the-middle)
- Minimizing exposure (attack surface and vectors)
- Network hardening
- Network policy development and enforcement
- Defense in depth
- Perimeter networks (DMZs)/Proxy Servers
- Implementing firewalls and virtual private networks (VPNs)
- Implementing IDS/IPS
- Network access control (internal and external)



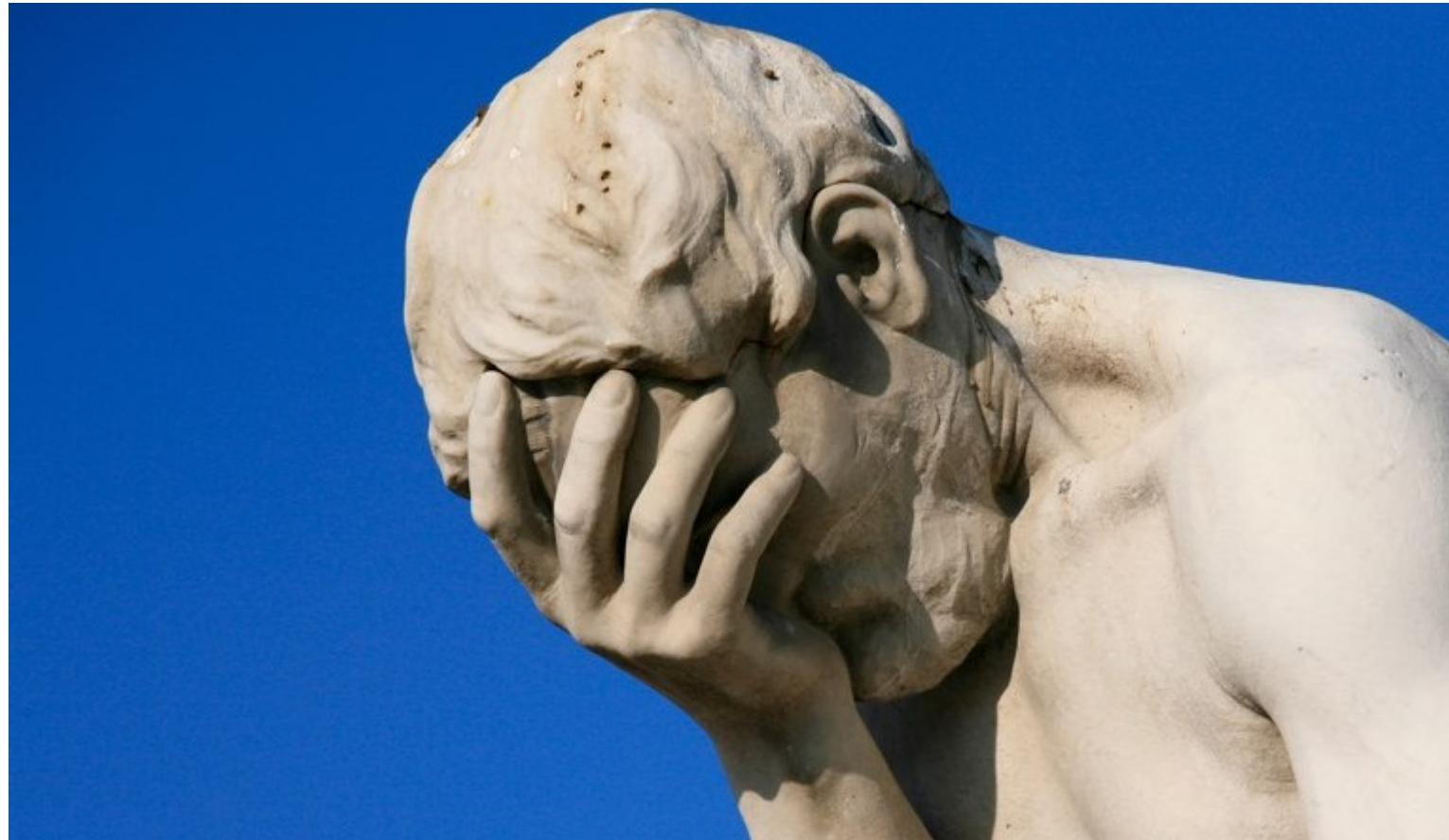


How the classes are structured

- Theoretical concepts presented mainly during classroom hours, but also during lab sessions
- Practical activity during lab hours
- Some topics will have an assignment with two parts:
 - 1) An implementing part in the virtual lab environment
 - 2) A reporting part
- Assignments are part of the exam
 - They are a pre-requisite for accessing the written exam



Remember that... things can go bad!!



Especially during labs... Please be tolerant with us 😊



To do the first 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: please download it **BEFORE** coming to the labs
 - https://drive.google.com/file/d/1W6JQzWVyH5_LKLD20R6XH1ugPDP5LWP5/view?usp=sharing
- After you will be granted the access to our virtual infrastructure (the ACME co.)



How to pass the exam

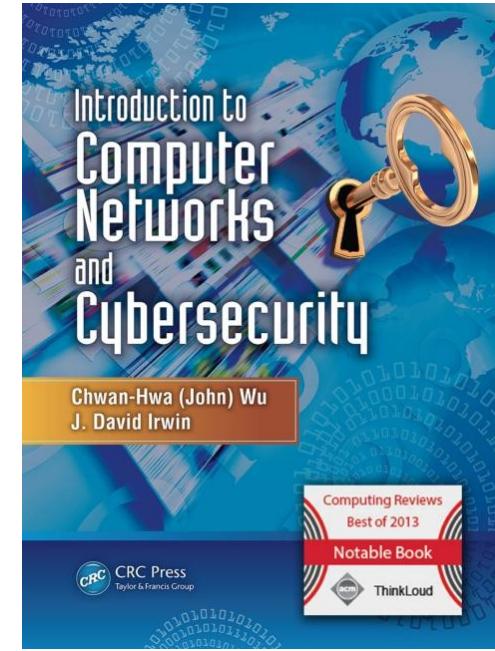
- All the rules are applied both to full-time and part-time students
 - 4 mandatory assignments
 - full written exam
 - (optional) on demand oral exam for students with a written exam ≥ 27
- Assignment rules
 - Assignments are individual or group activities to apply techniques and tools introduced during the classes
 - Assignments do not have an hand-in date, but must be handed-in before taking the full-exam
 - Assignments are evaluated A, B or C and provide an increment to the final grade of the written exam (up to +3 points)
- Written exam rules
 - A written exam is valid for one year and supersedes any other written exam done previously





Material

- Hand-notes
- Slides of the lectures
- Articles linked in the website
- Main textbook:
 - **Introduction to Computer Networks and Cybersecurity** (Chwan-Hwa (John) Wu, J. David Irwin, 1e) CRC Press
- Other books will be suggested during the lectures for each considered subject
 - Your contribution is welcome: don't hesitate to share useful resources via classroom





Feedback

- Always welcome
- Write me your suggestions



**YOUR FEEDBACK
MATTERS**



Hack the box!

The screenshot shows the official website of Sapienza Università di Roma. At the top, there is a dark header with the university's name and a graduation cap icon. Below the header, a banner features the university's logo and the text: "Here you can find information about Sapienza Università di Roma.". The main content area contains a section about the university's history and ranking, followed by a bio for the "Cybersecurity" page on the website, which includes a link to <https://cybersecurity.uniroma1.it/>, a joining date of 4 months ago, and 15 students and 18 respect.

<https://www.hackthebox.eu/>

- Website with CTF (capture the flag) challenges
- Join our team (and please do your job!)
- To be enrolled:
 - Register yourself in the website
 - Join our telegram group <https://t.me/htbsapienza>
 - Write in the group your hack-the-box username

Self-assessment!



What is this assessment about?

- If you can not understand some questions at all, you are definitely lacking the necessary prerequisites
- If you understand the questions, but just cannot remember the answers, you may think you would look in the book (or would google)...
- You can find the answers in most elementary books on computer security and using your computer (as a engineer would)
- But do not cheat yourself – we will be using the commands and the terms asked about in the test all through the course...
 - Don't get discouraged: simply you have to **fill the gap** to catch up!



Self assessment 1: linux





Self assessment 2: linux2





Self assessment 3: network





Self assessment 4: network2





Self assessment 5: cryptography





That's all for today

- **Questions?**
- See you next lecture!
- Bonus reference to get used to linux CLI and tools:
<http://overthewire.org/wargames/bandit/bandit0.html>
 - Go to bandit and try to reach level 34!!
 - 33 is also good :-)
 - Take notes of the passwords and how you obtained them
 - Try to learn as much as you can solving each level

Practical Network Defense

Master's degree in Cybersecurity 2021-22

Networking 101

Angelo Spognardi
[*spognardi@di.uniroma1.it*](mailto:spognardi@di.uniroma1.it)

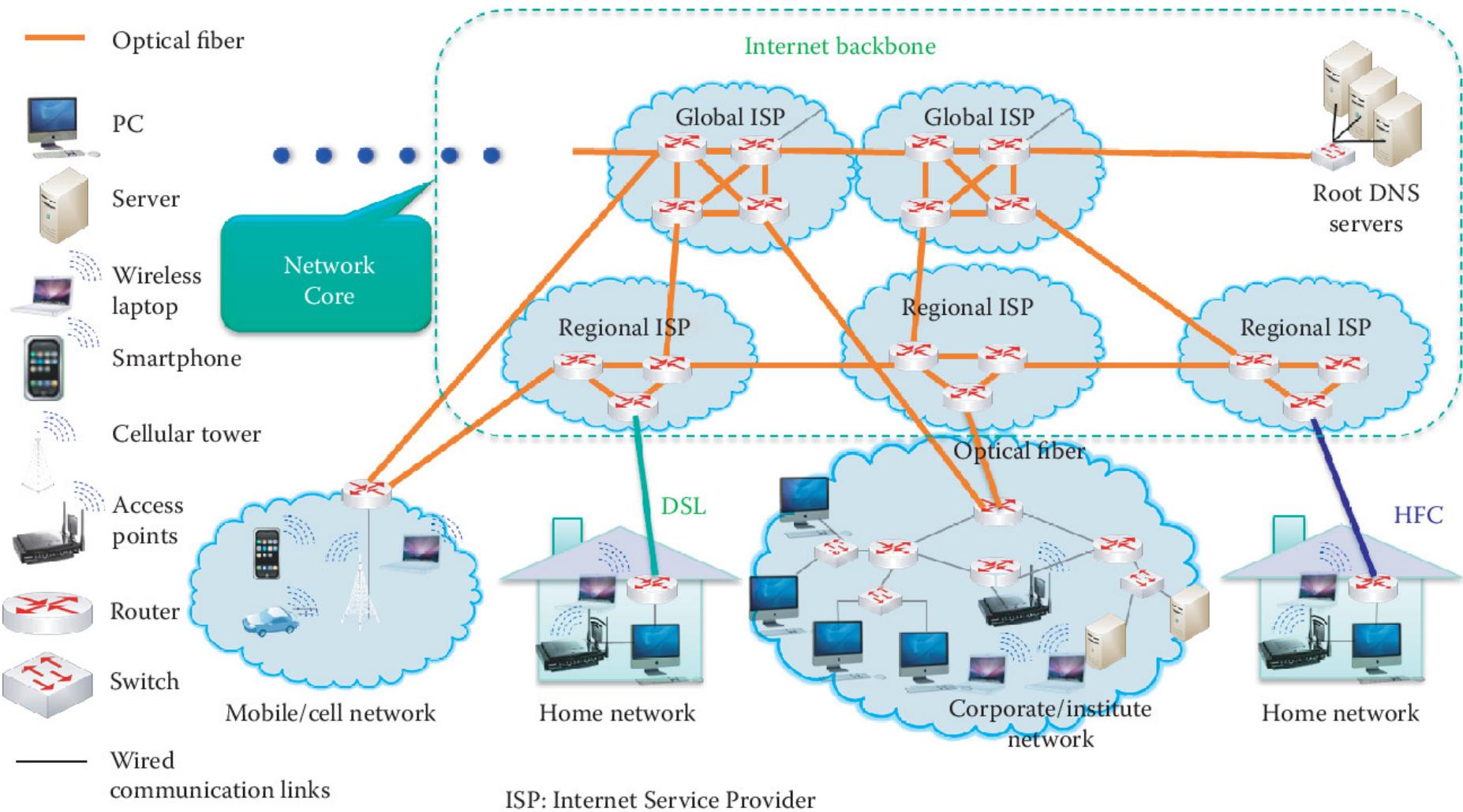
*Dipartimento di Informatica
Sapienza Università di Roma*



What is Internet

- Internet: an interconnected network of networks
 - Hierarchical networks:
 - Internet backbone: connecting the ISPs' backbones
 - ISP backbone: connecting organizations' backbones
 - Organization backbone connects local area networks (LANs)
 - LAN connects end systems
 - Public Internet versus private intranet
- Internet standards
 - RFC: Request for comments
 - IETF: Internet Engineering Task Force
 - Free download of RFCs at rfc-editor.org

Internet architecture

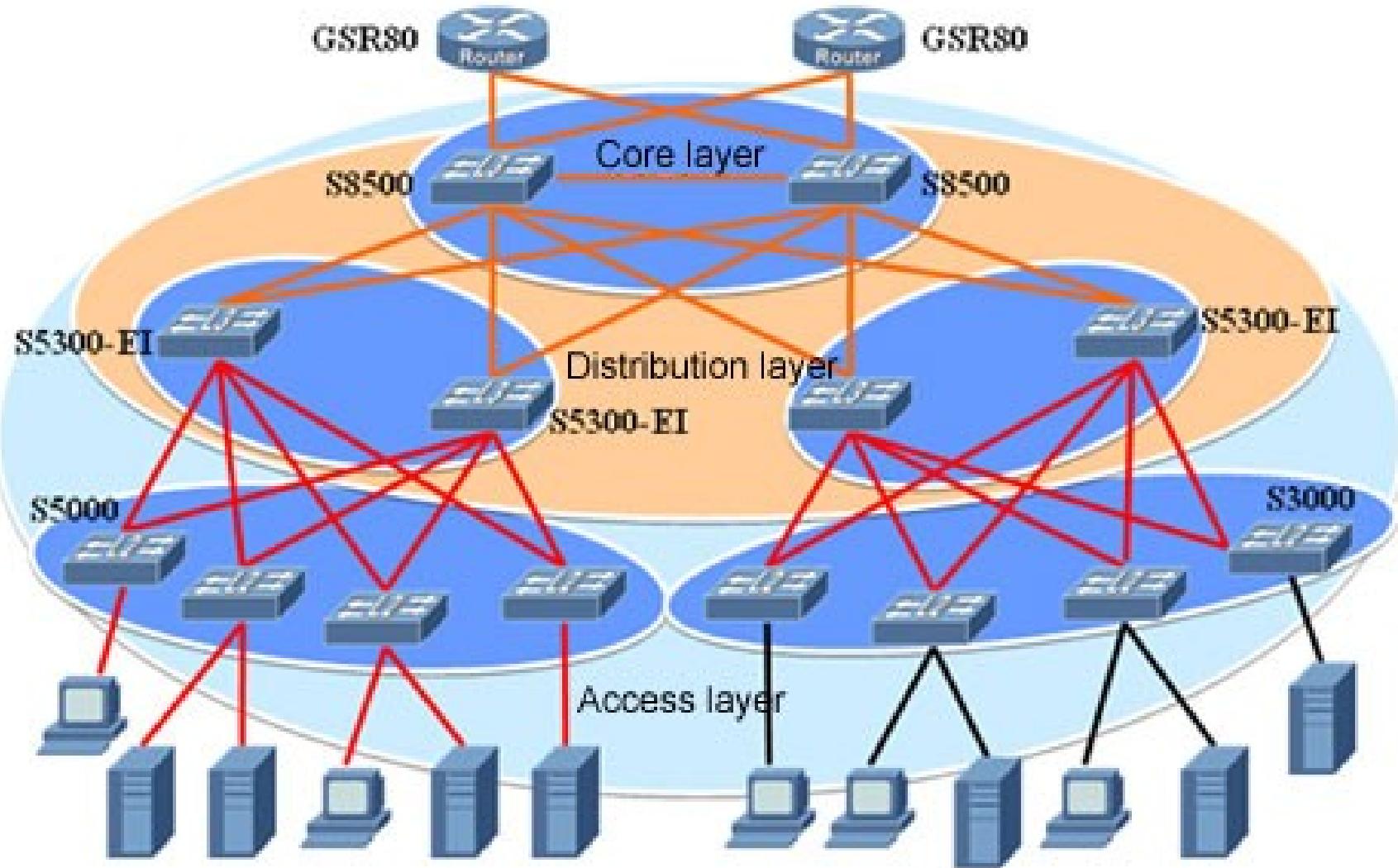




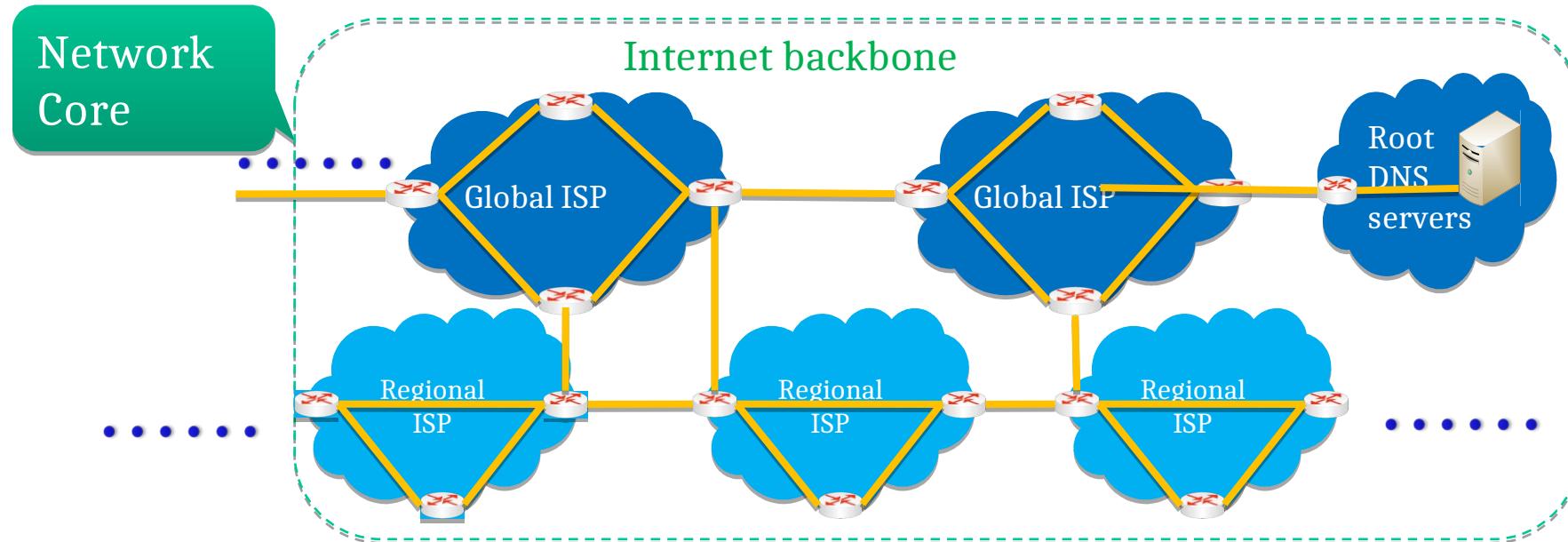
Internet hierarchy

- Network edge
 - Hosts: server, client, P2P
 - Applications: http, mail, Facebook, Twitter
- Network core
 - Edge router: connecting an organization/ISP to the Internet
 - Interconnection of routers using fiber
 - Naming services
- Access networks
 - Wired, or wireless communication links

Internet, hierarchical approach

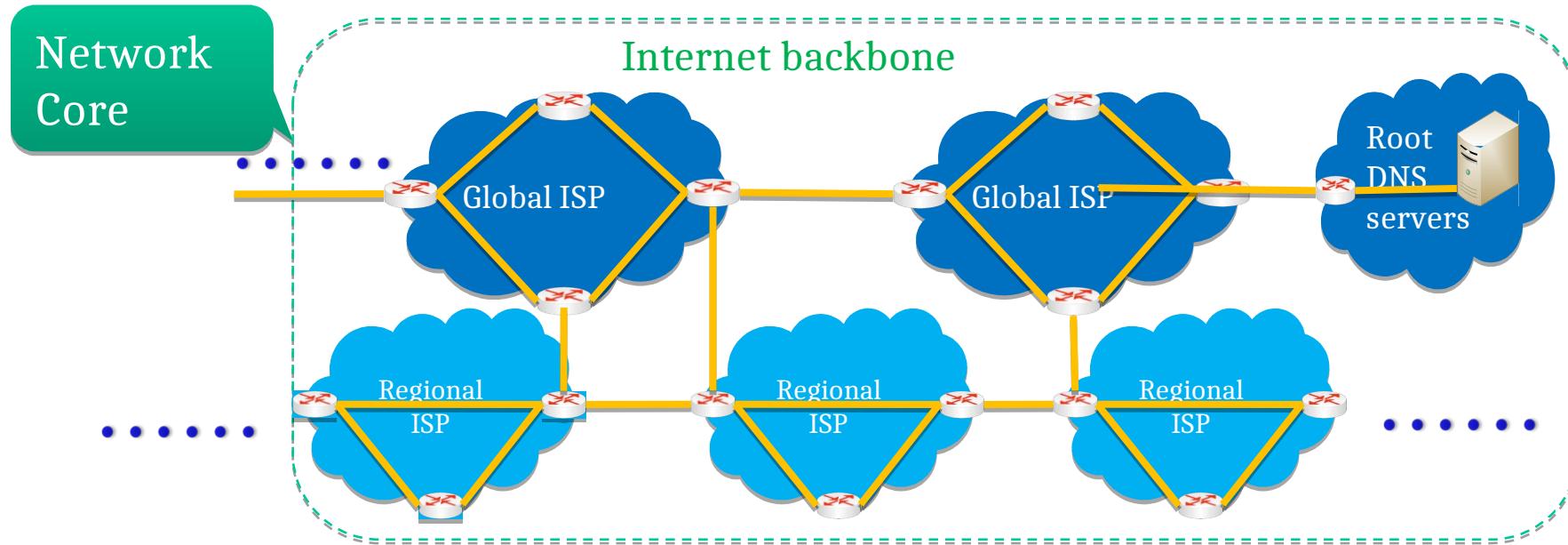


Internet Core



- Routers and fiber links (in orange) form the Internet core
- Routers work together to figure out the most efficient path for routing a packet from source to destination host
 - A distributed algorithm can adapt to changing Internet conditions
 - Great idea during the cold war
 - Routing tables are generated and maintained in real time

Internet core management: ISPs



- The core is provided by ISPs that interconnect multiple continents
- ISPs
 - Global ISPs or Tier-1 ISPs
 - Regional ISPs or Tier-2 ISPs

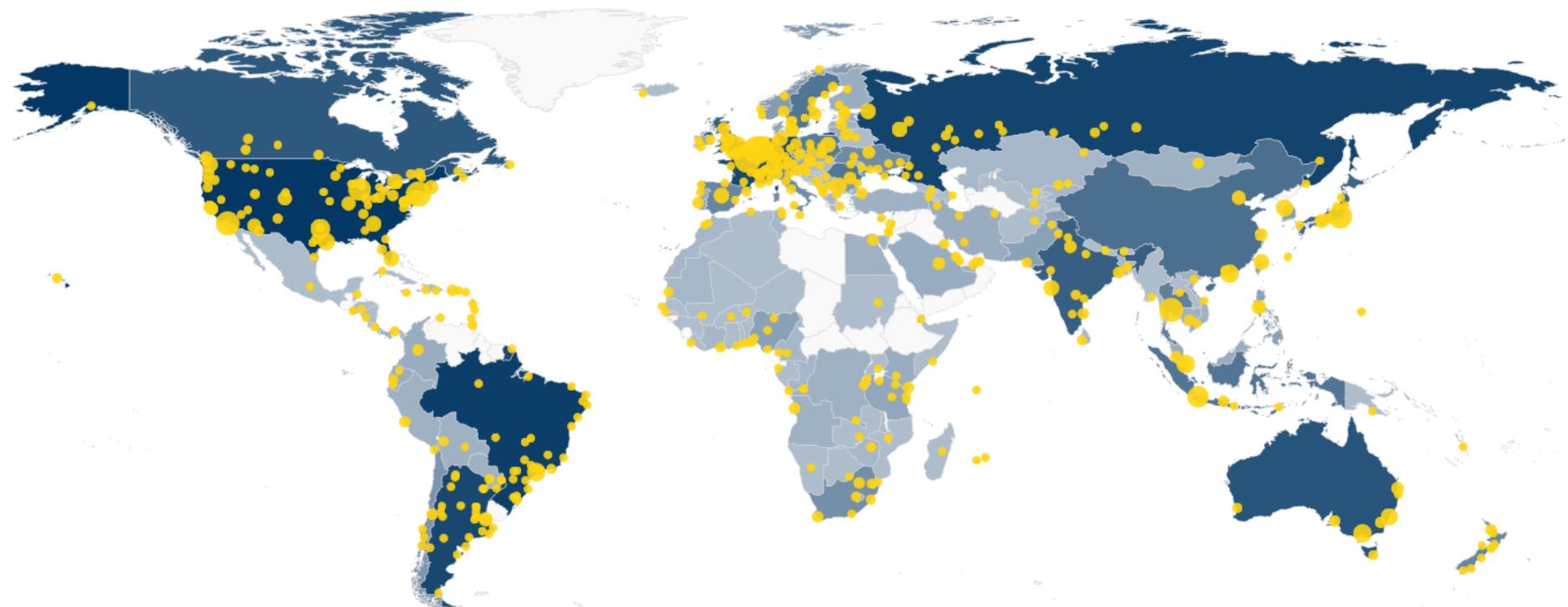


Internet: network of networks

- Internet Backbone connects tier-1 ISPs
 - e.g., Verizon, Sprint, AT&T, Qwest, Level 3 Communications
- The backbones of tier-1 ISPs are interconnected at various access points called Internet eXchange Points (IXP)
- The number of IXPs around the world is continually growing
 - to date more than 1000
- Interactive (probably not exhaustive) map:
<https://www.pch.net>



Global map of Internet eXchange Points



964 IXPs shown - Number of IXPs by Country

Source: pch.net



Protocols

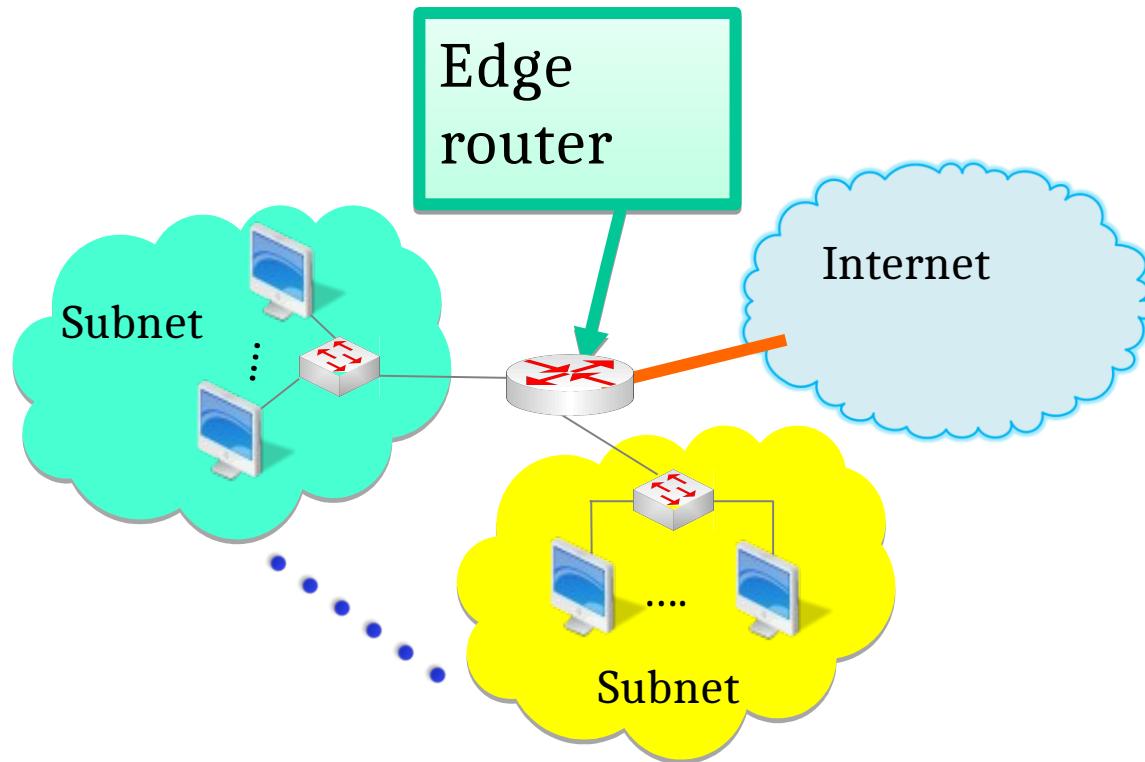
- Specify rules about the desired service
 - Procedure Rules
 - Types and sequences of messages exchanged
 - Syntax and semantics
 - Actions to take with respect to messages and events
 - Message Format: format, size and coding of messages.
 - Timing: the time to wait between any event.
 - Access to medium
 - Flow control
 - Timeouts



Protocol specification examples

- Modularization → Many protocols for each layer
 - Hides implementation details
 - Layers can change without disturbing other layers
 - Development (one company can tackle one module)
 - Maintenance
 - Updating the system
- Packet switching
 - Best effort delivery
 - Better for resource sharing
- Network congestion and flow control

Router and subnet



- Internet uses a gateway (edge router) to connect a Local Area Network (LAN) or a subnet to the hierarchical network

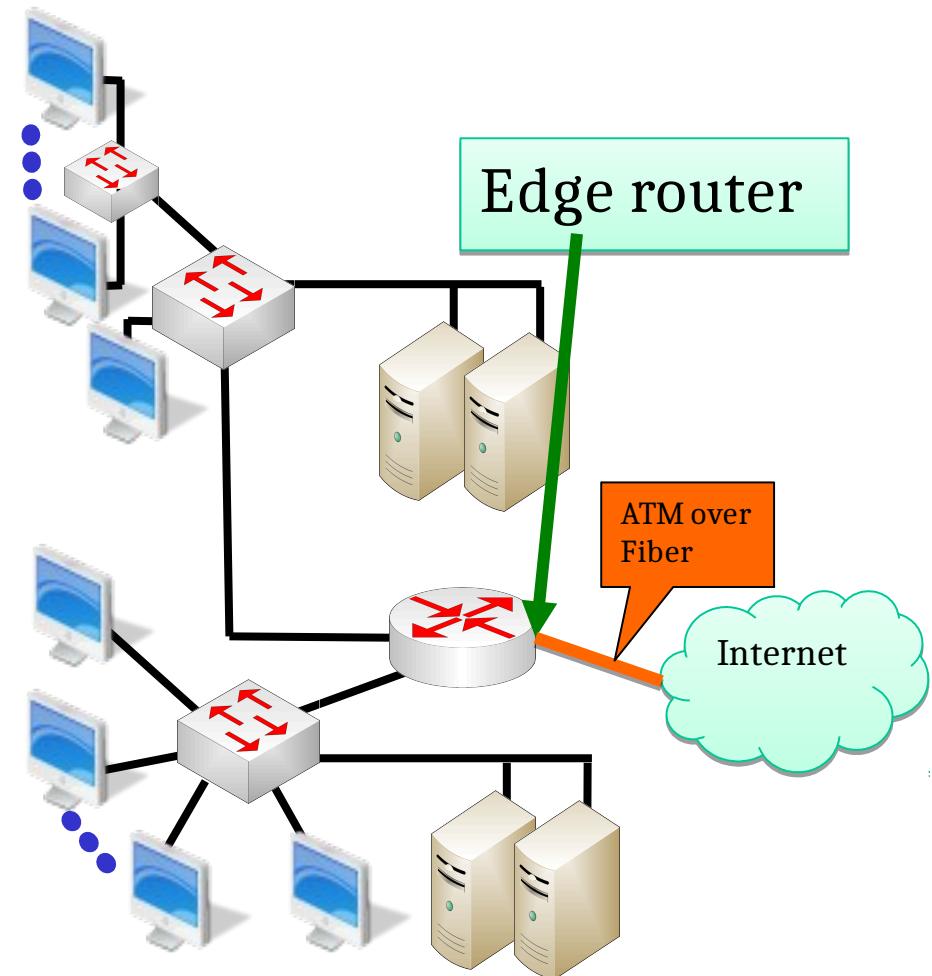


Residential Internet access overview

- Point to point protocol (PPP) for access to an ISP
- Dialup via modem
- DSL: digital subscriber line
- Cable modem
- Fiber In The Loop
- Broadband over a power line
- Broadband wireless: such as WiMAX
- Satellite

Local area network connected to Internet

- Organization/home local area network (LAN) or subnet connects hosts to edge router
- Edge router connects LANs to Internet
 - Telco uses ATM over fiber
- Ethernet LAN
 - Hosts connect into Ethernet switch
 - 10Mbps, 100Mbps, 1Gbps, 10Gbps Ethernet
- ATM: asynchronous transfer mode





Access layer

- Constituted by networks with end-points of the same local management
- Provides connectivity among stations on the same network
- Nodes in the same network can directly communicate among them
 - Used protocol: Ethernet family



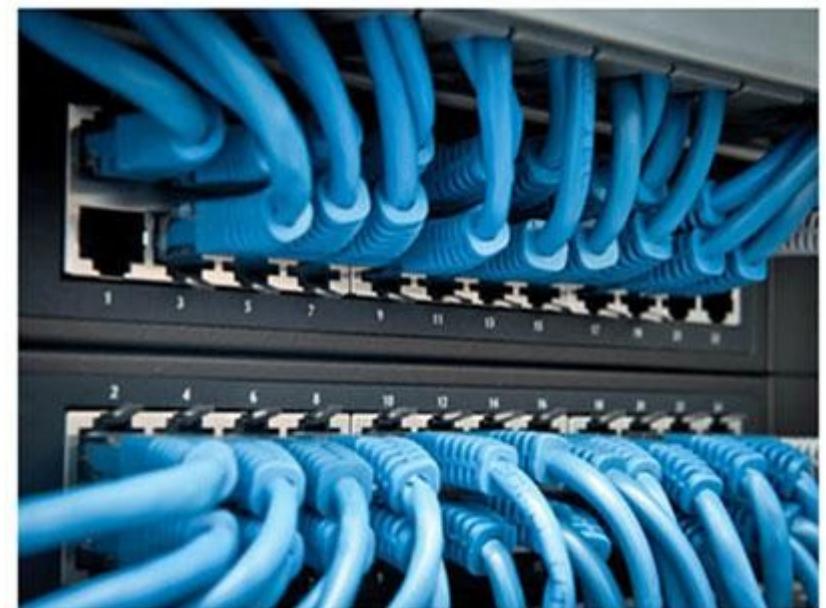
Ethernet (IEEE 802.3) networks

- Each host in a Ethernet network has a NIC (Network Internet Card) with a (generally) fixed address
- MAC addresses are 48 bits (6 bytes) long and UNIQUELY ideintify hosts in the network
- Each host only processes packets intended for it
- Each Ethernet packet ("frame") has a fixed format

Preamble (7 byte)	SFD (1 byte)	Dest. (6 byte)	Source (6 byte)	Type (2 byte)	Data (PDU livello 3) (46-1500 byte)	FCS (4 byte)
----------------------	-----------------	-------------------	--------------------	------------------	--	-----------------

How to build a Ethernet network

- All the hosts connected together with a shared “transmission system” based on Ethernet are a network, as if they were connected to the same medium
 - Two computer with a single Ethernet cable
 - Many computer connected with several Ethernet cables to a single device (generally a **switch**, but also **repeater**, **hubs** or **bridges**)
 - Many computer connected with several Ethernet cables to several devices (generally **switches**)



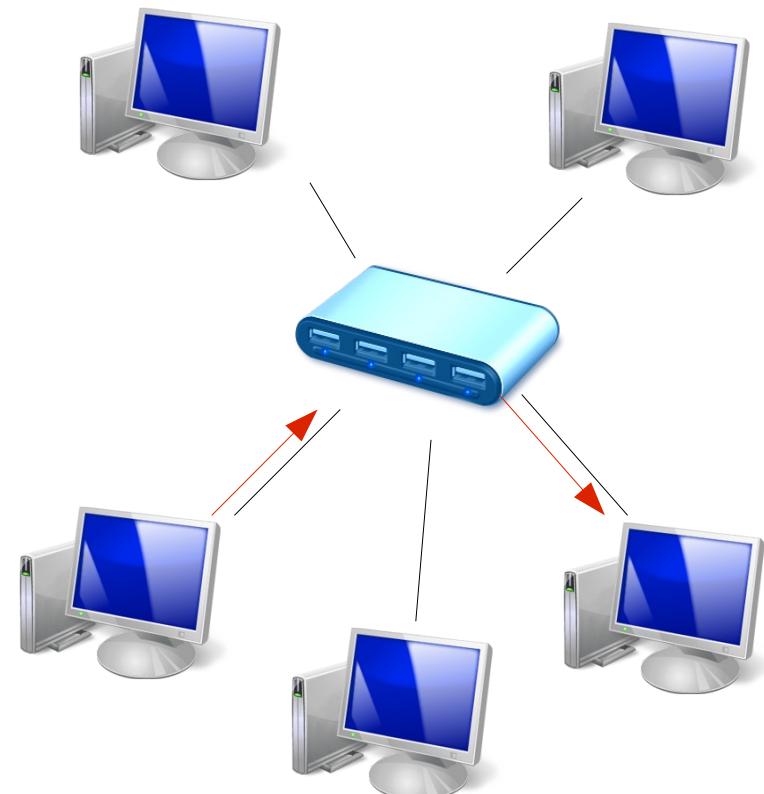


Ethernet and its broadcast domains

- An Ethernet network constitutes a **broadcast domain**
 - For historical reasons there also exist collision domains, but full-duplex and switches have made them obsolete
- Ideally frames sent in a broadcast domain are potentially received by all the hosts in the network
 - All the host receive all the frames and only read some
- Actually, switches segment the network to limit the explosion of packets in the network
- Only broadcast messages are “replicated”

How switches segment the network

- Switches remembers the source MAC addresses on the different ports
- They only replicate the frame on the segment where the destination MAC address replies
 - Tables of MAC are
 - ARP tables for hosts
 - CAM tables for switches





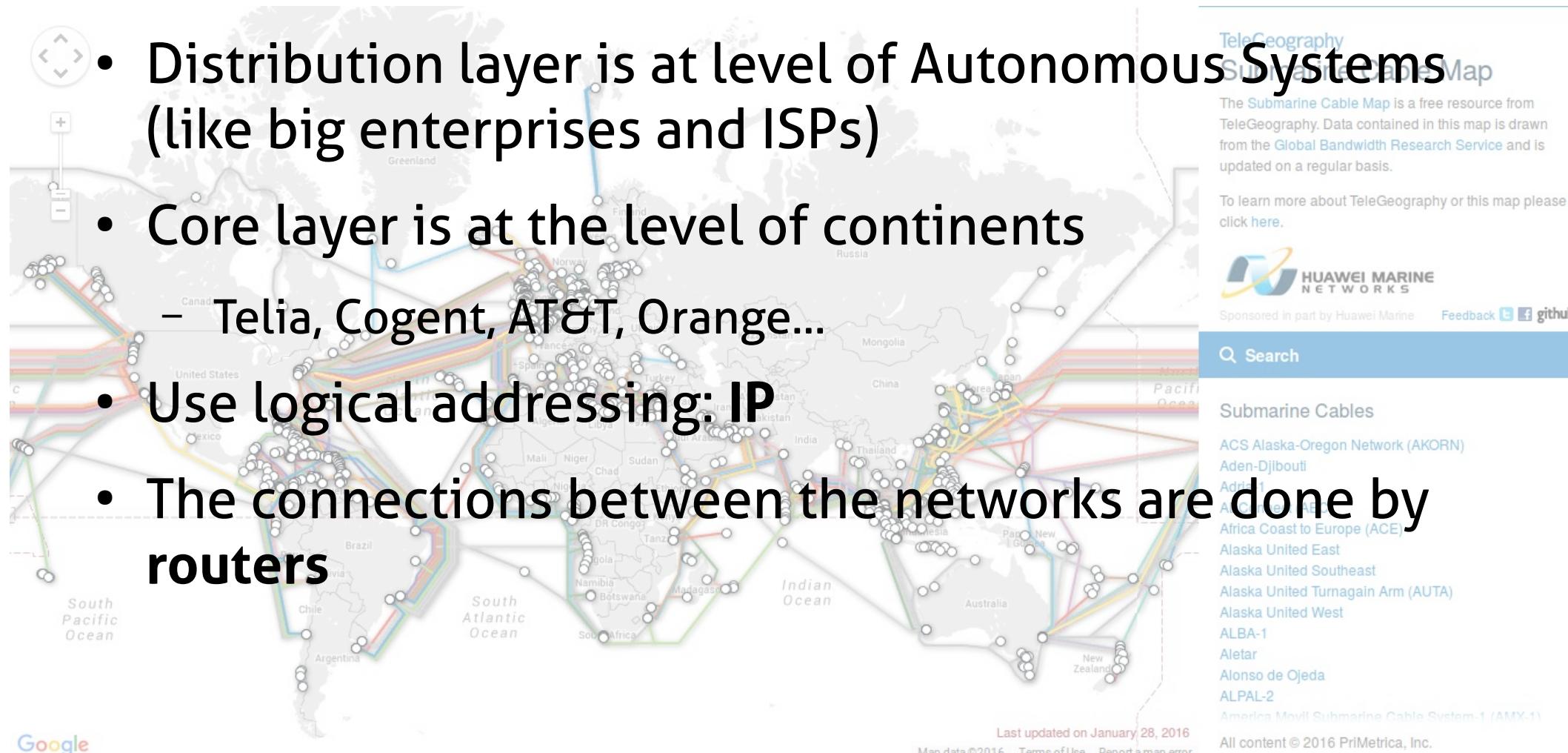
Why Internet is not a large Ethernet net?

- Ethernet makes high use of broadcast packets
 - Inefficient for large networks
- Large networks are split in order to reduce the broadcast domain
- There is the need of a LOGICAL division of the networks: Ethernet is the Access layer, but we need a Distribution layer
- Hosts in a local network use a Default Gateway to go out and have access to the Distribution layer
- Distribution layer is based on IP, the Internet Protocol



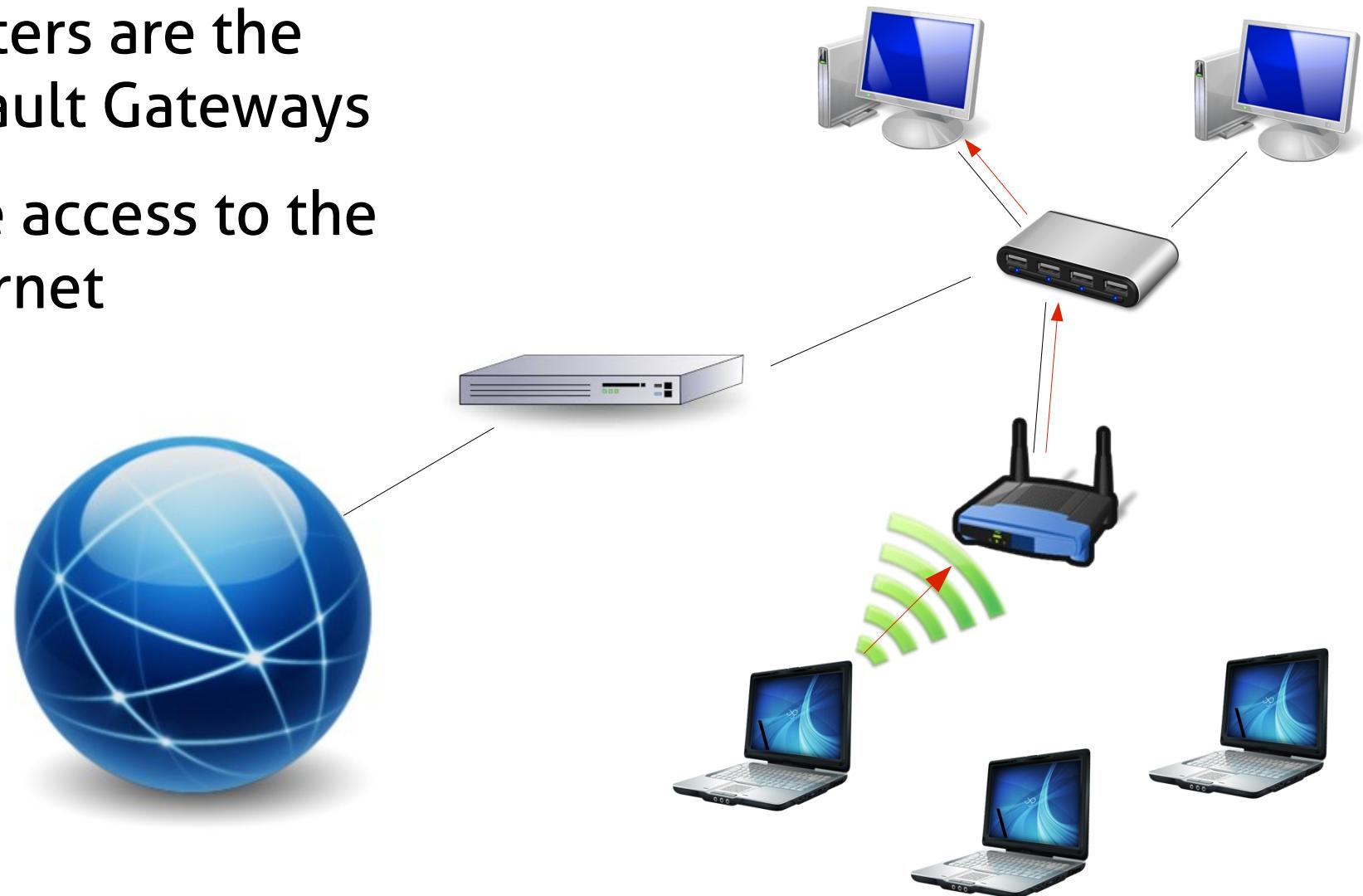
Distribution and core layers

- Interconnect local networks among them
- Distribution layer is at level of Autonomous Systems (like big enterprises and ISPs)
- Core layer is at the level of continents
 - Telia, Cogent, AT&T, Orange...
- Use logical addressing: IP
- The connections between the networks are done by routers



Router and switches

- Routers are the Default Gateways
- Give access to the Internet





Ethernet vs IP addresses

- Ethernet has **physical** addresses
 - You can not(*) change the MAC address of your NICs
 - It is like your **name**: it goes wherever you go
 - An Ethernet address tells WHO you are, but does not tell anything on WHERE you are
- IP has **logical** addresses
 - You can change IP address of your NIC
 - It is like your **home address**: it changes if you go somewhere
 - IP addresses are used to identify and reach networks and hosts



Local addresses and remote addresses

- Analogy: if you want to say something to somebody
 - If both of you are in the same room, you can simply call his/her name and he/she will answer
 - Directly connected → Local address
 - If you are NOT in the same room, you have to know where he/she is, before sending the message AND the message has to LEAVE the room through the door
 - Remote address
- How to know if one IP is the same network than you?
 - Subnet mask

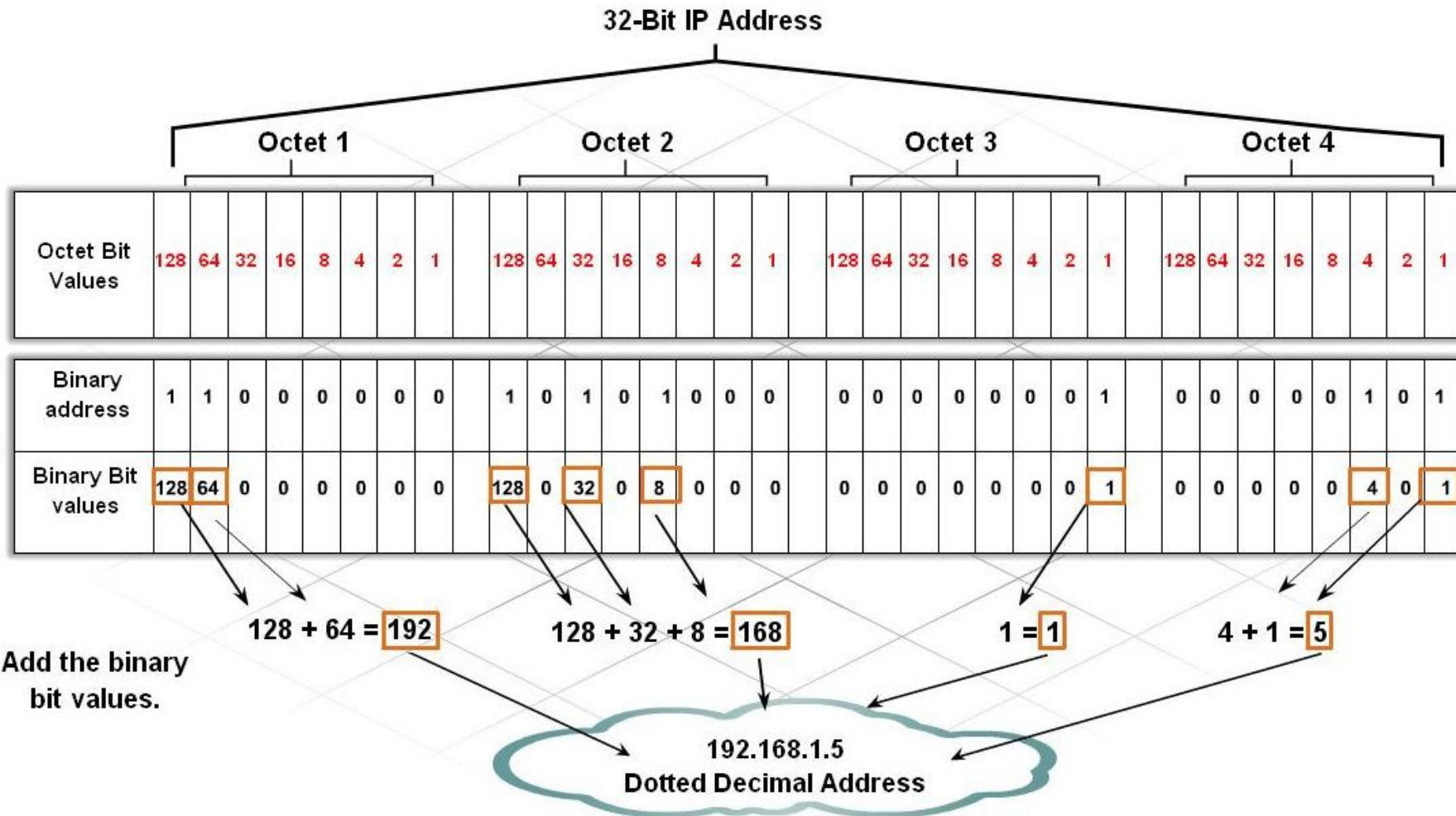


IP Addresses

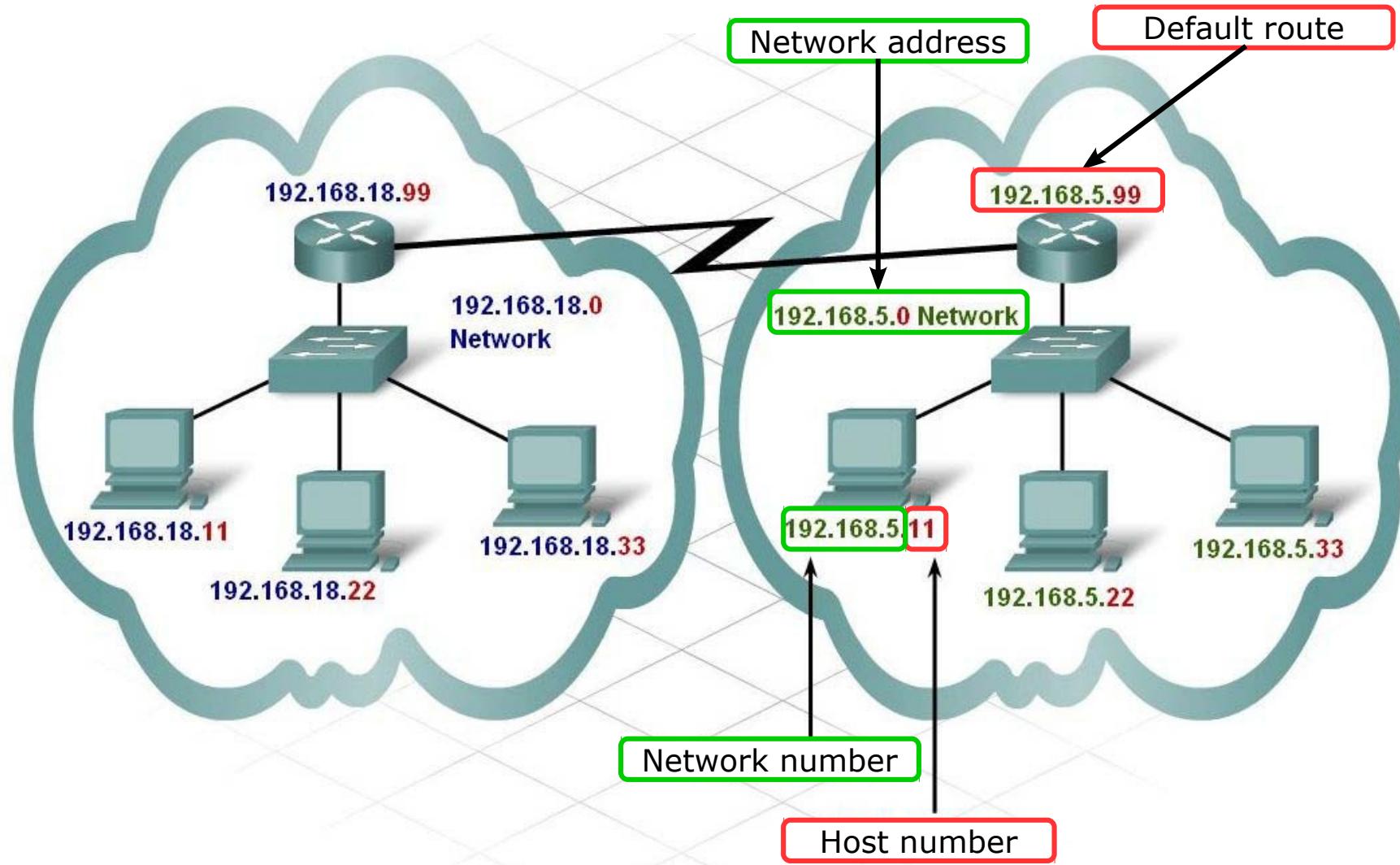
Two versions of IP addresses: IPv4 and IPv6.

- IPv4 defines IP address with 32 bits organized in four octets (8 bits in each).
- IPv6 (version 6) has 128 bits.
- For human readability, the bits in each octet are separated by dots while writing an IPv4 address (colons in IPv6).
 - E.g. **69.58.201.25** and **fe80::250:56ff:fea0:1**
- Certain bits from the left correspond to the network address (**69.58.201**) and the remaining correspond to define the computer (host) on the network (**25**).
- Subnet mask defines boundary between network portion and the host portion of the IP address.

Dotted decimal IP Address



Network address and Host address





Types of IP Addressing

There are three types of IP addresses

- **Unicast (one to one)**
 - These refer to a single destination host
- **Broadcast (one to all)**
 - These refer to every host on a network or subnet
- **Multicast (one to many)**
 - Refers to a group of IP addresses in a network, not necessarily all of them
 - <http://www.firewall.cx/networking-topics/general-networking/107-network-multicast.html>



IP Addressing, Classful

Allocation classes of IP addresses

- **Class A** (24 bits for host addresses, or /8)
 - 0.0.0.0 to 127.255.255.255
- **Class B** (16 bits for host addresses, or /16)
 - 128.0.0.0 to 191.255.255.255
- **Class C** (8 bits for host addresses, or /24)
 - 192.0.0.0 to 223.255.255.255
- **Class D** (Multicast)
 - 224.0.0.0 to 239.255.255.255
- **Class E** (Reserved)
 - 240.0.0.0 to 255.255.255.255



IP Addressing

- There are routable and non-routable address ranges
- Routable addresses need to be unique on the Internet
- Non-routable address ranges are defined in RFC1918
 - Distinction between “private” and “public” IP
 - 10.0.0.0 - 10.255.255.255 (10/8 prefix)
 - 172.16.0.0 - 172.31.255.255 (172.16/12 prefix)
 - 192.168.0.0 - 192.168.255.255 (192.168/16 prefix)



IP Addressing, example

192.168.8.0/24

- The last eight (32 minus 24) bits of 32 total will be used for host addresses
- The first address reserved for the **network address**
- The last address reserved for the **broadcast address**
 - Then, we have $2^{(32-\text{netmask})} - 2$ hosts in any CIDR specified network
- So, if we are given **192.168.8.0/24**, 192.168.8.0 is the network address, 192.168.8.255 is the broadcast address, and .1 to .254 are host addresses



IP Addressing, Classless

- Each set of 8-bits (octet) can hold values from 0-255
 - Poor flexibility!
- Idea: let's use a Variable Length Subnet Mask (VLSM)
- Introduced by CIDR (Classless Inter-Domain Routing), a new notation for the netmask:
 - specify how many bits of the 32-bit total will specify the network address
 - The remaining bits specify the host addresses
- Ex: 10.10.10.0/26
 - the netmask can also be specified in dotted-quad notation, as in 10.10.10.0/255.255.255.192



IP Addressing, other example

192.168.1.248/30

- $2^{(32-30)} - 2 = 2^2 - 2 = 4 - 2 = 2$ hosts (2 usable addresses)
 - 192.168.1.248 is the network address
 - 192.168.1.251 is the broadcast address
- Large networks can be subnetted:
 - We say things like “There are 64 /30 subnets in a /24 network”
- Many smaller networks can be “supernetted” for routing reasons → “summarization”



Notes about IP addresses

- In Point-to-Point links, using a 30 bit netmask is a waste...
 - If A sends a broadcast, only B will receive it...
- There is the proposal of RFC 3021:
 - *Using 31-Bit prefixes on IPv4 Point-to-Point Links*
 - <https://tools.ietf.org/rfc/rfc3021.txt>
- Reduce the waste of IP addresses in a subnet
 - Other ways to reduce it?
 - NAT
 - IPv6



Exercises

- Determine the network part, the host part, the network size (number of hosts), the network address, the broadcast address and the type of the following IP addresses:
 - 10.11.12.1 netmask 255.255.255.128
 - 192.168.4.32 netmask 255.255.255.224
 - 172.17.17.17 netmask 255.255.240.0
 - 10.11.12.0/21
 - 192.168.4.32/27
 - 172.17.17.17/29



Exercises 2

- Determine whether the destination IP address is local or remote
 - Namely, if it belongs to the same network than the Host
 - It is plenty of tools online for this job, but try to put your pen to paper

Host IP address	Host subnet mask	Destination IP address
210.145.149.123	255.255.255.0	210.145.253.199
192.168.4.189	255.255.255.224	192.168.1.107
10.154.187.89	255.192.0.0	10.152.179.88
132.100.45.5	255.255.252.0	132.100.45.45
151.251.100.101	255.255.0.0	166.200.110.10
172.32.9.82	255.255.255.240	172.32.9.79



Summary

- Internet: a mess!
 - Networks connected by other networks that meet in IXP
- Hierarchical structure: to make Internet admins to survive!
 - Core and distribution layers managed by ISPs
- Protocols: the Internet manuals!
 - Describe rules and formats to exchange data and make services to be well defined
- Ethernet and IP: who you are and where you are
 - Addresses with two different meanings, for two different purposes
- Networks and subnet masks: to train with math!



That's all for today

- **Questions?**
- See you next lecture!
- Bonus reference to get used to Linux CLI and tools:
<http://overthewire.org/wargames/bandit/bandit0.html>
 - Go to bandit and try to reach level 34!!
 - 33 is also good :-)
 - Take notes of the passwords and how you obtained them
 - Try to learn as much as you can solving each level

Practical Network Defense

Master's degree in Cybersecurity 2021-22

Networking 101 lab

Angelo Spognardi
[*spognardi@di.uniroma1.it*](mailto:spognardi@di.uniroma1.it)

*Dipartimento di Informatica
Sapienza Università di Roma*



Main tasks

- Properly configure the topology provided in the lab packages
- Manual configuration
 - Via ip and via interfaces file
- Automatic configuration
 - Via DHCP
- Network debug
 - Fix configuration errors
- Reference link:
<https://www.debian.org/doc/manuals/debian-reference/ch05.en.html>

Assigning IP addresses

- Blocks of public addresses are allocated by IANA (Internet Assigned Numbers Authority) and RIRs (regional Internet registries)
 - To companies, institutions, universities and so on
- Private addresses can be used by anyone
 - Manually
 - Automatically (via DHCP Dynamic Host Configuration Protocol)





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>
- You can move in the directory and run **lstart**
 - **NOTE:** the first **lstart** attempt can (...will...) fail



Katharà main commands (aliases)

- All the commands should be used in the lab directory
- Start (restart) a lab exercise:
 - `lstart/lrestart`
- Stop a running lab exercise
 - `lclean`
- Wipe the kathara environment (when labs do not restart after a failure)
 - `kwipe`
- List virtual networks and virtual interfaces of VMs
 - `docker network list`



Lab activity: ex1



Exercise 1: pnd-labs/lab1/ex1

- Manually configure pc1, pc2 and pc3 in order to be in the same network than the r1 host, with IP 192.168.100.30/29
- Configure pc1 using the **interfaces** file (before starting the lab), within the pc1/etc/network/ directory (remember to add the **ifup** command in the startup file)
- Configure pc2 using the **ip** command
- Configure pc3 using the **ifconfig** command
- The DNS server can be the server used by the host machine
 - This should be used also in the **r1.startup** file
- The default gateway must be the r1 host
- Verify connectivity within the network and with the Internet (ex: **ping www.google.com**)



Properly configure a host

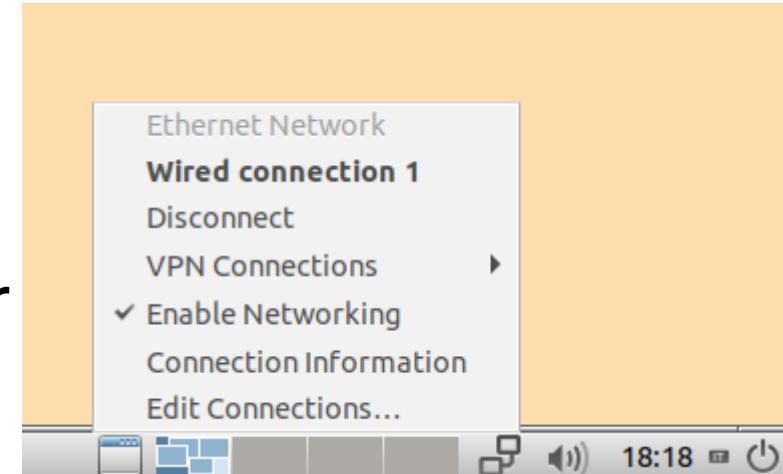
- In order to (properly) use Internet a host has to receive 4 main pieces of information
 - The IP address
 - The netmask
 - The IP address of its default gateway
 - Namely the host of its local network able to access to the distribution layer
 - The IP address of a DNS (Domain Name Server)
 - Namely a remote host able to translate human intelligible names to IP addresses



Systems with a GUI

- Use the related tool
- Most used one: **network-manager**
 - Counterpart from command line:
`nmcli`

```
angelo@lakr:~/teaching/Netdef2020$ nmcli connection show -a
NAME           UUID                                  TYPE   DEVICE
TIM-30962259  8397b8a4-ab2e-45e2-ae07-990a2652cd03  wifi   wlp2s0
tun0          20cb14f6-bcc0-405a-b4c6-887514e31eec  tun    tun0
angelo@lakr:~/teaching/Netdef2020$
```



- Quite extensible with plugins
 - Example: for managing additional VPN types



Manual network configuration (linux) (old school – legacy)

- **ifconfig** to assign the IP address
 - This command is used to configure network interfaces, or to display their current configuration. In addition to activating and deactivating interfaces with the “up” and “down” settings
- **route** to define the default gateway
 - The route command is the tool used to display or modify the routing table
- **/etc/resolv.conf** to specify the DNS server(s)



Manual network configuration using ip (preferred)

- **ip addr** to assign the IP address
 - This command is used to configure network interfaces, or to display their current configuration. In addition to activating and deactivating interfaces with the “up” and “down” settings
- **ip route** to define the default gateway
 - The route command is the tool used to display or modify the routing table
- **/etc/resolv.conf** to specify the DNS server(s)



Other details about ip command

- Show interfaces
 - `ip link show`
- Bringing interface up/down
 - `ip link set eth0 (up|down)`
- Set MAC address
 - `ip link set eth0 address 00:11:22:33:44:55`
- Show IP address
 - `ip address show [dev eth0]`
- Add/remove IP address
 - `ip address (add|del) 10.0.0.1/8 dev eth0`
- Flush any IP address (remove the assigned address/es)
 - `ip address flush [dev eth0]`



ip for routing purposes

- List/flush routing table
 - ip route (list|flush)
- Add/del routes
 - next hop
 - ip route (add|del) 10.0.0.0/8 via 10.0.0.1
 - default
 - ip route (add|del) default via 10.0.0.1
 - direct forwarding
 - ip route (add|del) 10.0.0.0/24 dev eth0



ip for ARP and more

- Show ARP cache
 - `ip neigh show [dev eth0]`
- Flush ARP cache
 - `ip neigh flush dev eth0`
- Add/del/change/replace ARP cache entry
 - `ip neigh (add|del|change|replace) to 10.0.0.2 lladdr 00:11:22:33:44:55 dev eth0 nud "state_name"`
 - (state_name: permanent, stale, noarp, reachable...)
- IP tunneling (IPinIP, IPinGRE, IPv6 tunneling)
 - `ip tunnel`



Lab activity: ex2



Exercise 2: pnd-labs/lab1/ex2

- Configure r1, pc1 and pc2 in order to receive their networking configuration from a DHCP server (r1)
 - The DNS server can be the server used by the host machine
 - The default gateway must be the r1 machine, with IP address 192.168.100.30/29
- Configure r1 in order to operate as a DHCP server on the eth1 interface
 - You can install **udhcpd** or any other server
 - (apt install udhcpd)
- Configure pc1 using the **interfaces** file
- Configure pc2 using the **dhclient** command (after run)
- Verify connectivity within the network and with the Internet (ex: ping www.google.com)



DHCP

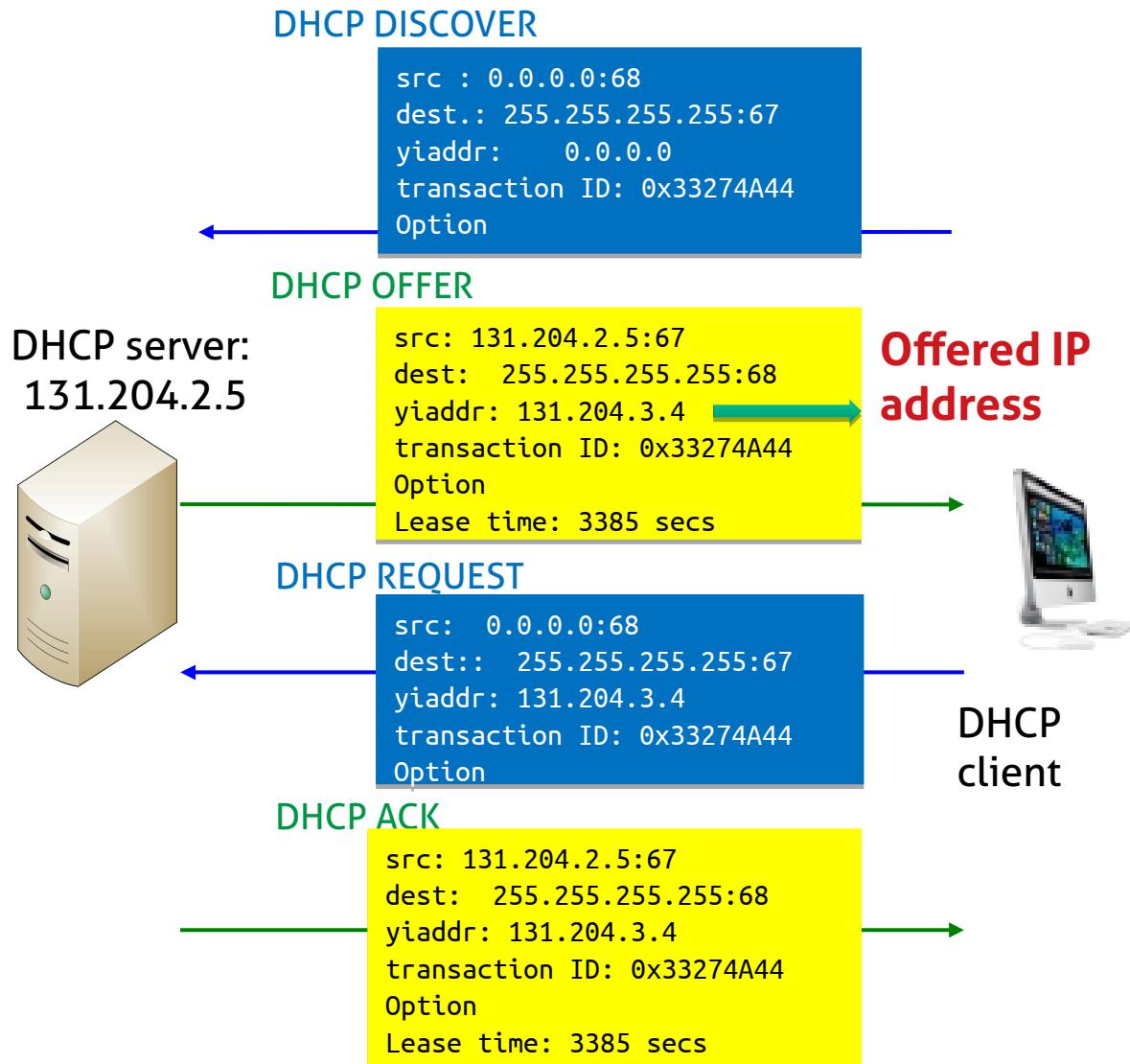
- Client-server mechanism
- Server has a pool of IP addresses to distribute, together with the network configuration
- Client requesting a new IP address receive a proposal and accept it
- Once accepted, the IP is reserved for a “leasing time”
- Observations?



DHCP Client/Server

DHCP procedure:

1. Host broadcasts "DHCP Discover"
2. DHCP server responds with "DHCP Offer"
3. Host requests IP address: "DHCP Request"
4. DHCP server sends address: "DHCP ACK"

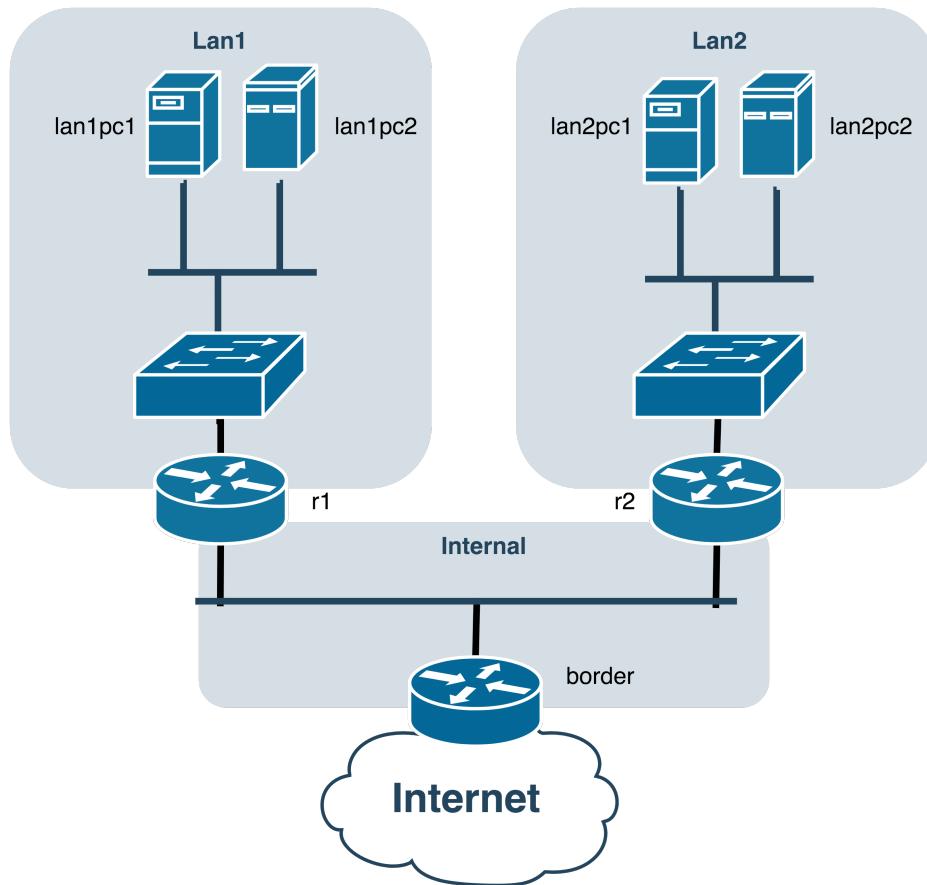


yiaddr: (offered) your IP address



Lab activity: ex3

Exercise 3: pnd-labs/lab1/ex3



- The border r0 is already configured to act as the gateway.
- eth0 on r0 has not to be configured.
- PC from the two lans have to be able to reach each other and to reach internet



ex3 activity

- Configure the 4 pc and the three routers so that the two lans are reachable and all can reach the Internet
 - You have to use the 172.16.0.0/16 network and assign subnetworks to all the LANs in the topology. Think about the most suitable approach.
 - r0 has to be the default gateway of the whole network. It is already set up to act as the default gateway. It is connected to the internet via eth0.
 - r1 and r2 have to be the default gateways for "lan1" and "lan2", respectively. They have to have a default route towards r0 and static routes to reach lan1 or lan2
 - you can use the **ip route** command (man ip-route)
 - the DNS server can be the server used by the host machine (this has to be set in all the pcs of the lab)
 - the PCs can be configured as you prefer



That's all for today

- **Questions?**
- See you next on Monday!

Practical Network Defense

Master's degree in Cybersecurity 2021-22

Network traffic monitoring

Angelo Spognardi
[*spognardi@di.uniroma1.it*](mailto:spognardi@di.uniroma1.it)

*Dipartimento di Informatica
Sapienza Università di Roma*



Layering Concepts

- The communication between the hosts in the network is organized in tasks, each assigned to a **layer**
- Each layer:
 - offers a service (a host of facilities) to the "Users" in the layer above
 - exploits the services offered by the layer below
- The task of a level involves the exchange of messages that follow a set of rules defined by a **protocol**.
- Example:
 - Layer ($N - 1$) provides an insecure service in which data can be overheard by unauthorized persons.
 - Protocol of level N specifies that messages sent via $(N - 1)$ -service are encrypted with symmetric encryption.
 - Layer N offers a secure, confidential service.



Encapsulation/decapsulation

- The data to be transferred from the application layer to application layer over a network.
- Each layer adds some protocol information and provides data to the layer below.
- The physical layer (bottom) sends data over the physical medium to the destination.
- The physical layer in the destination sends the data up the "stack".
- Each protocol in the destination reads the appropriate protocol information and forwards the data to the layer above.



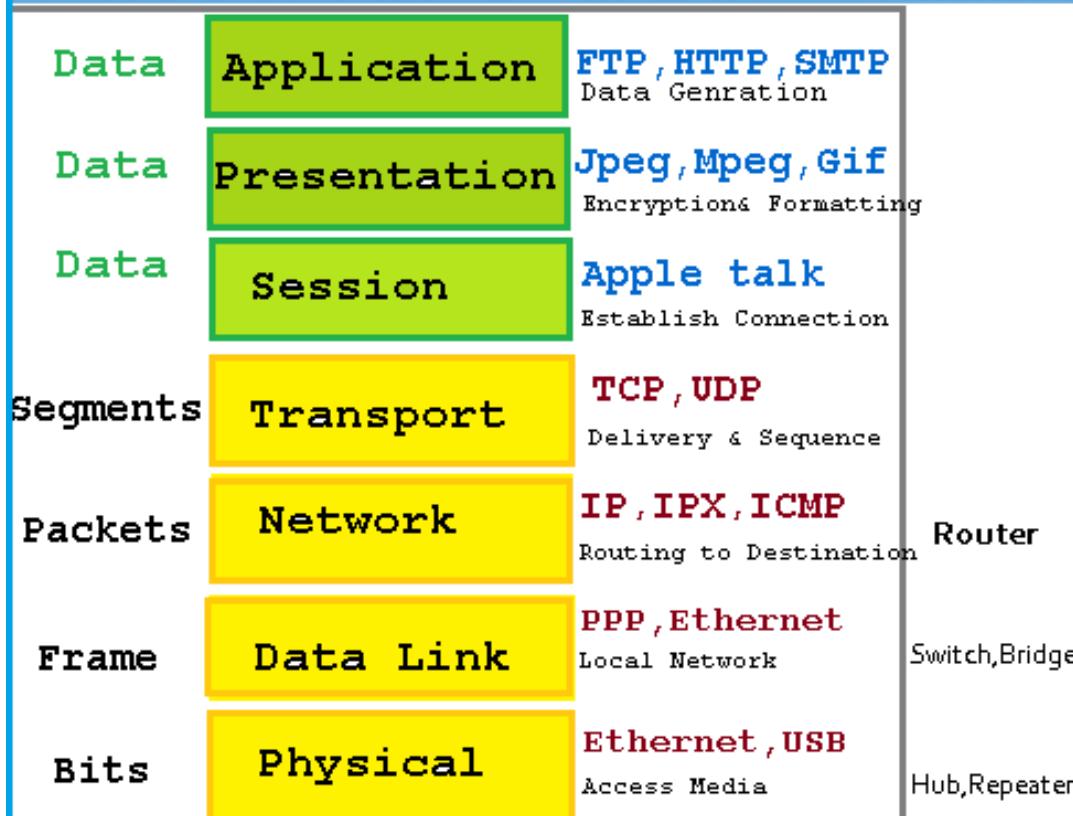
2 layered architectures

- ISO/OSI model: based on a reference model with 7 layer.
- TCP/IP model: created by the IETF, based on a reference model with 4 layers.
 - The lower TCP/IP layer is often split in 2 layers.
- Common idea: **packet switched network**

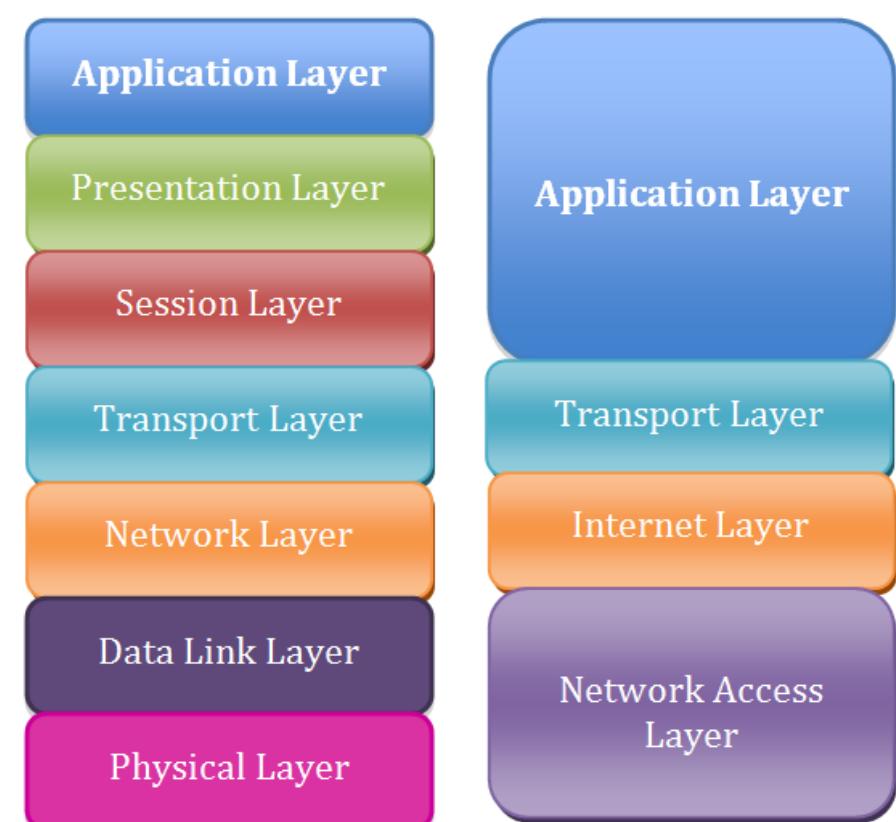


Architecture comparison

What is OSI Model



OSI Model



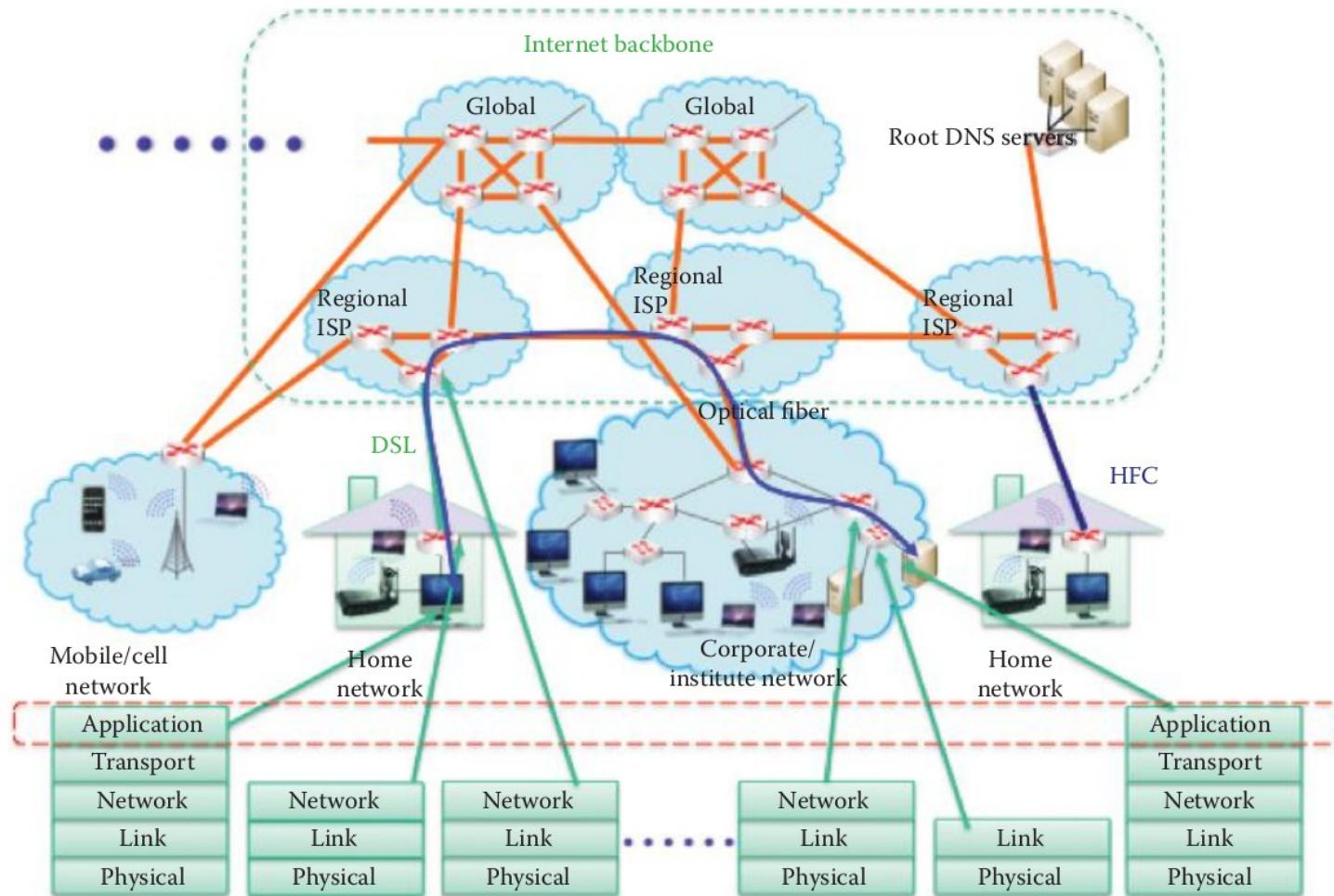
TCP/IP



TCP / IP model

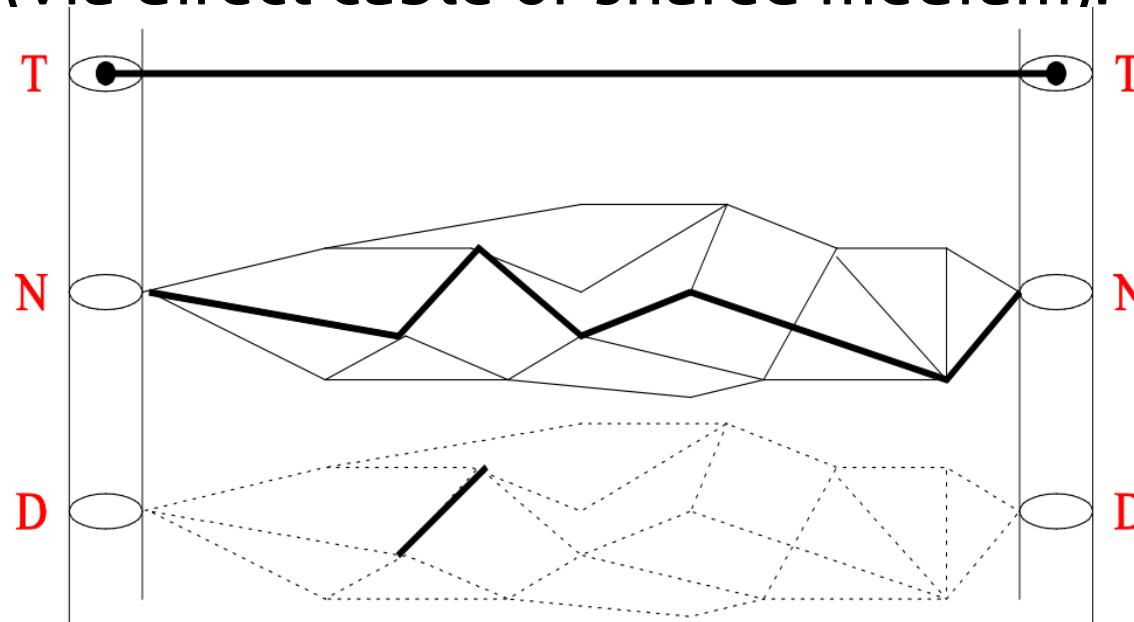
- Application layer: Corresponds to the top three layers of the OSI model.
 - Protocols: SMTP (sending e-mail), HTTP (web), FTP (file transfer), and others
- Transport layer: Equivalent to Layer 4 (Transport) of the OSI model
 - Protocols: TCP, UDP
- Internet: Equivalent to layer 3 (network) of the OSI model.
 - Protocols: IP, ICMP, IPSec
- Datalink: Equivalent to layer 2 (data link) of the OSI model.
 - Protocols: Ethernet, WiFi, ARP, etc.
- Physical layer: Equivalent to Layer 1 (Physical) of the OSI model.
 - NOTE: Datalink + physical layers are known as Network access layer.

Client-server communication example



Layer ideal representation

- **Transport:** the illusion of direct end-to-end connection between processes in arbitrary systems.
- **Network:** transferring data between arbitrary nodes.
- **Data Link:** transferring data between directly connected systems (via direct cable or shared medium).

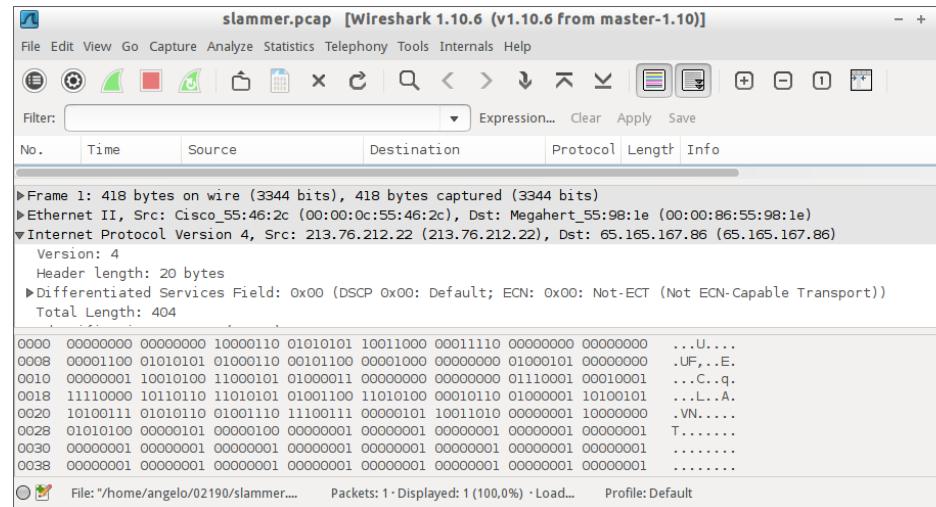
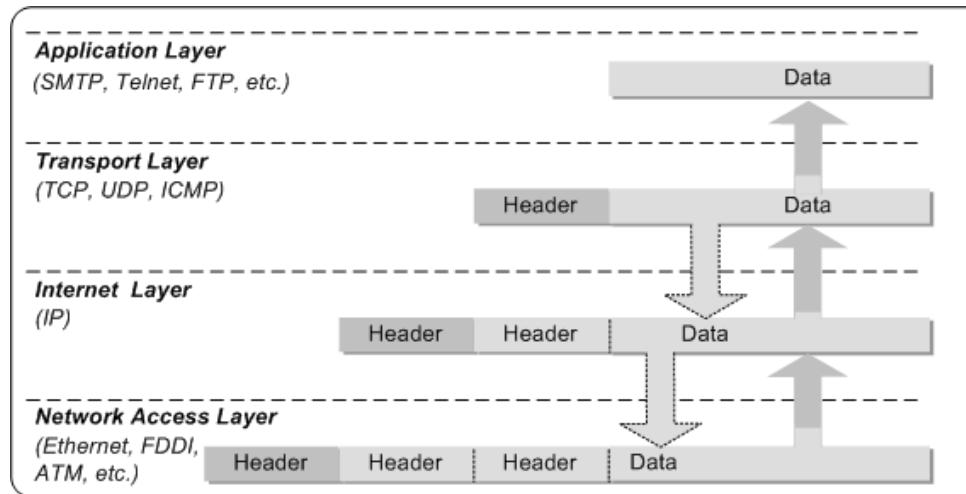




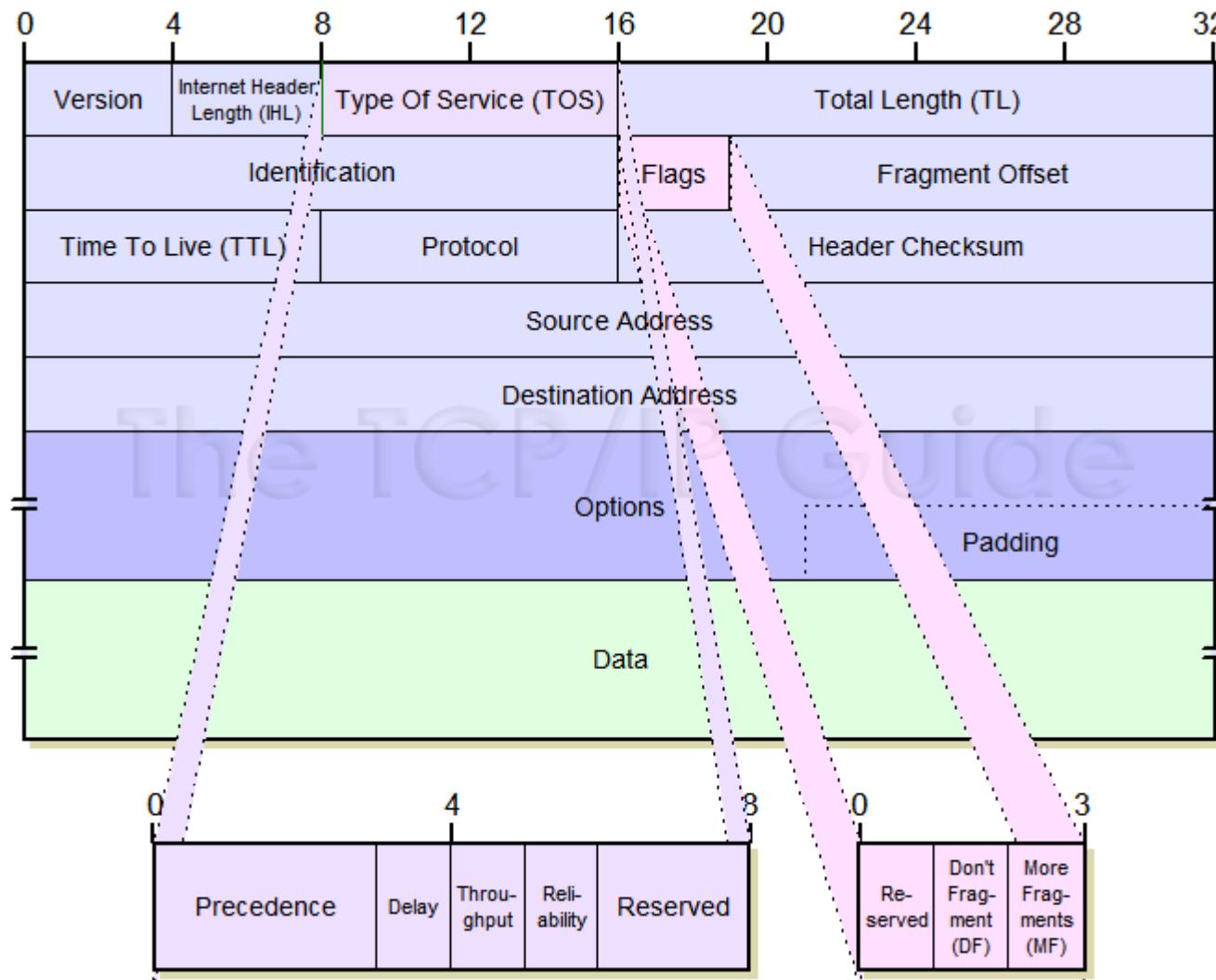
Addresses in the architectures

- Each layer has a type of address:
 - Application layer: Internet name, eg. `www.sapienza.it`
 - Transport layer: Port number, in the range [0..65535] that identifies the client or server. For example 80 for HTTP server.
 - Internet layer: IP address that identifies a network card, for example `151.100.17.4`
 - Datalink layer: MAC address, also identifies a network cards, for example `49:bd:d2:c7:56:2a`

Encapsulation in TCP/IP



IP packets



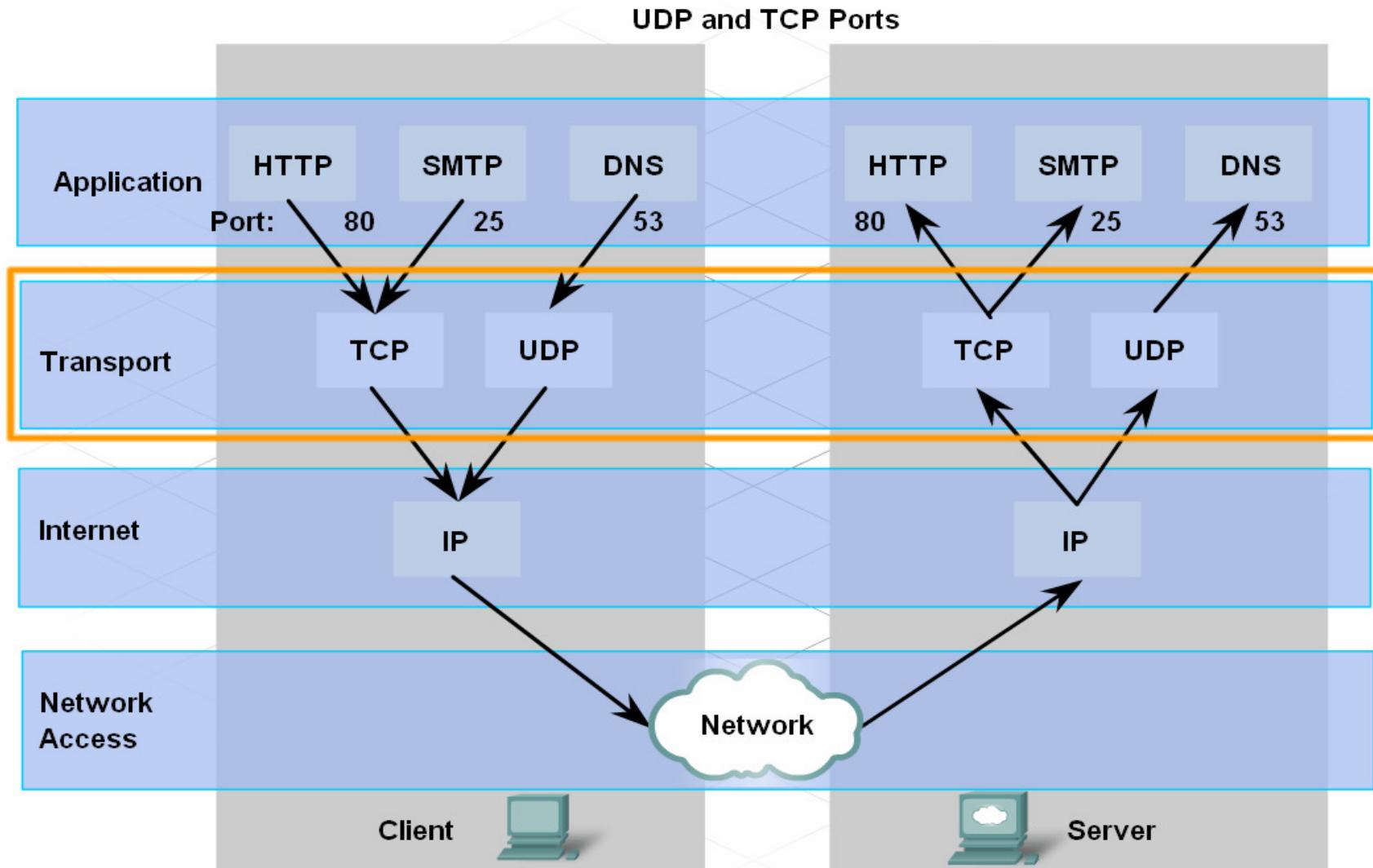
http://www.tcpipguide.com/free/t_IPDatagramGeneralFormat.htm



Ports

- Range [0..65535]
- Source port: randomly chosen by the OS
- Destination port determines the required service (application)
 - Assigned Ports [0..1023] are said “well-known ports” and used by servers for standard Internet applications:
 - 25: SMTP (sending mail)
 - 80: HTTP (web)
 - 143: IMAP (pick-up of mail)
 - Ports [1024..49151] can be registered with Internet Application Naming Authority (IANA)
 - Ports [49152..65535] ephemeral ports

Transport layer: TCP and UDP

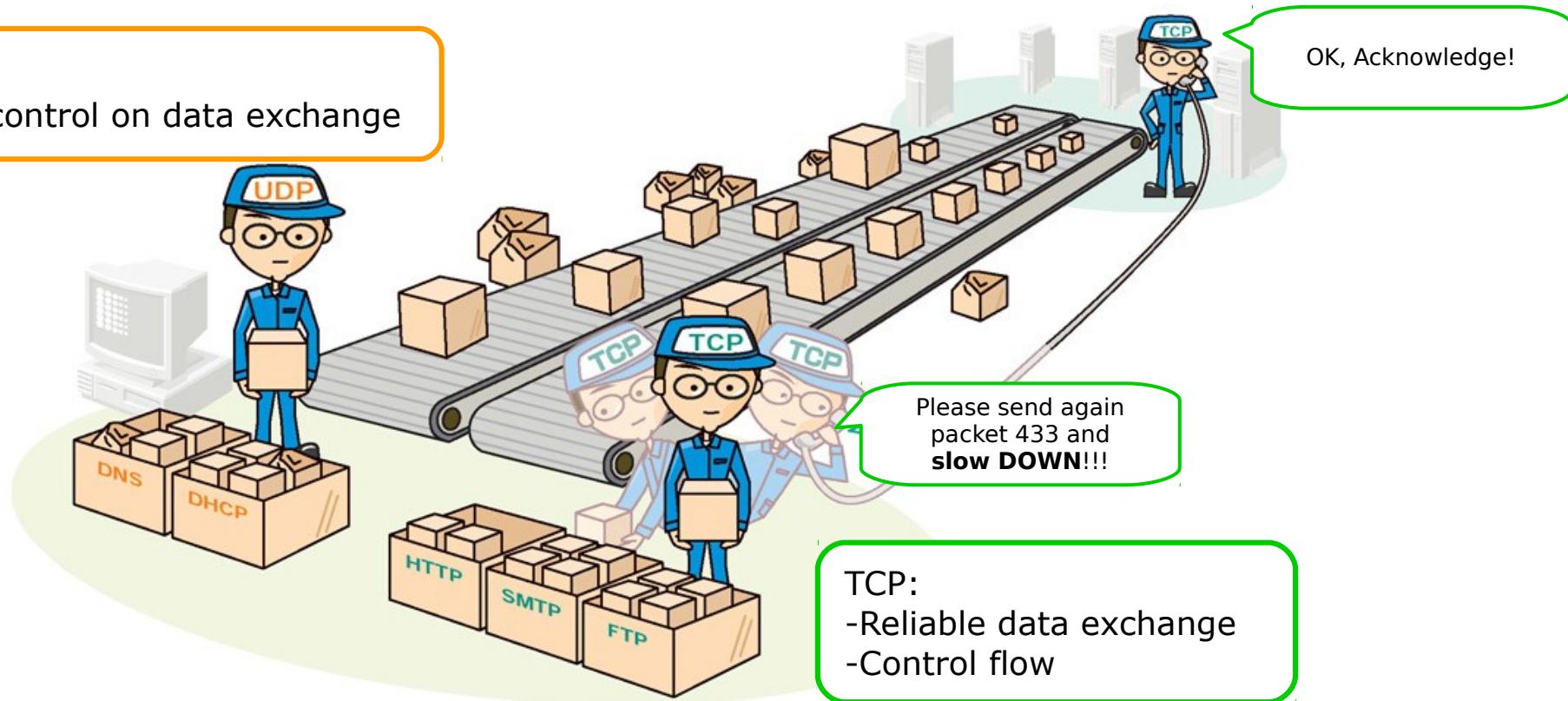


TCP vs UDP

- Connection vs Connection-less

UDP:

-No control on data exchange



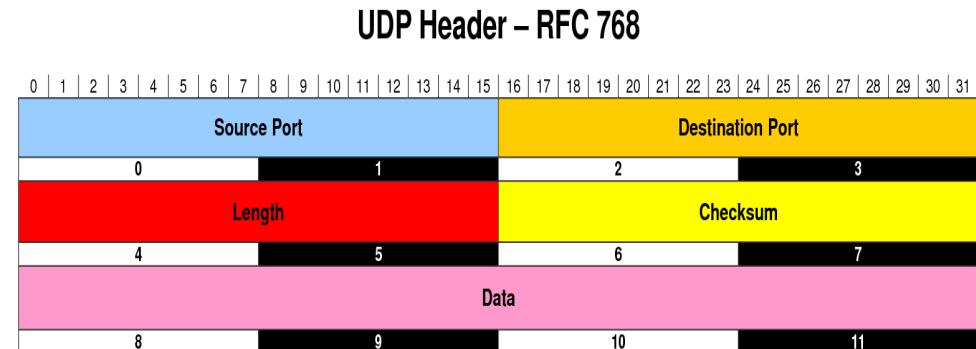
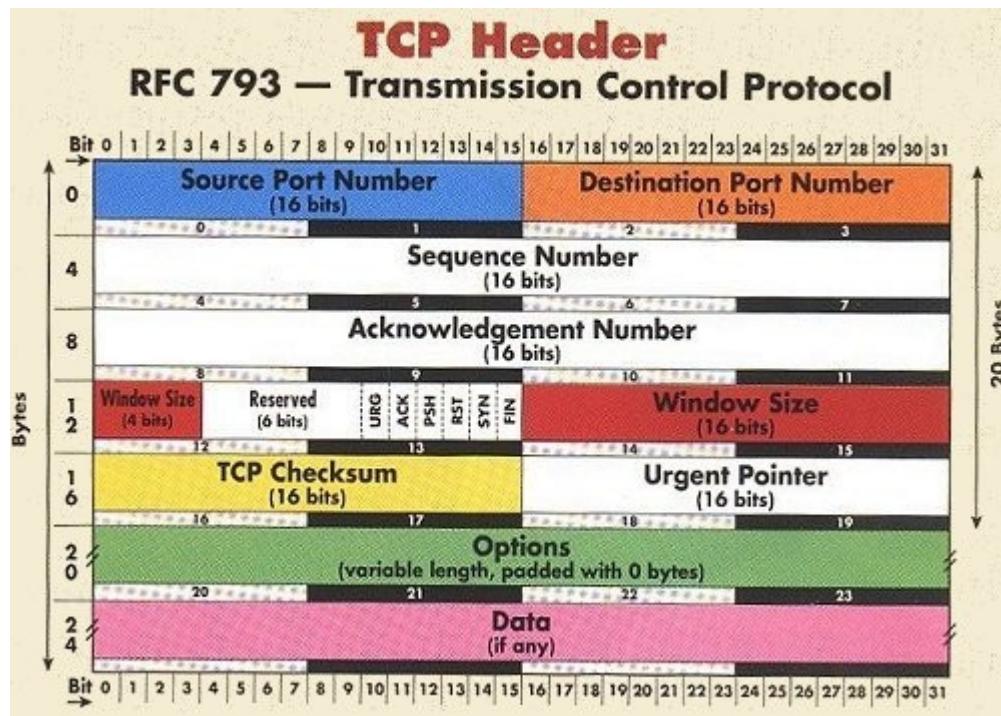
TCP:

-Reliable data exchange
-Control flow

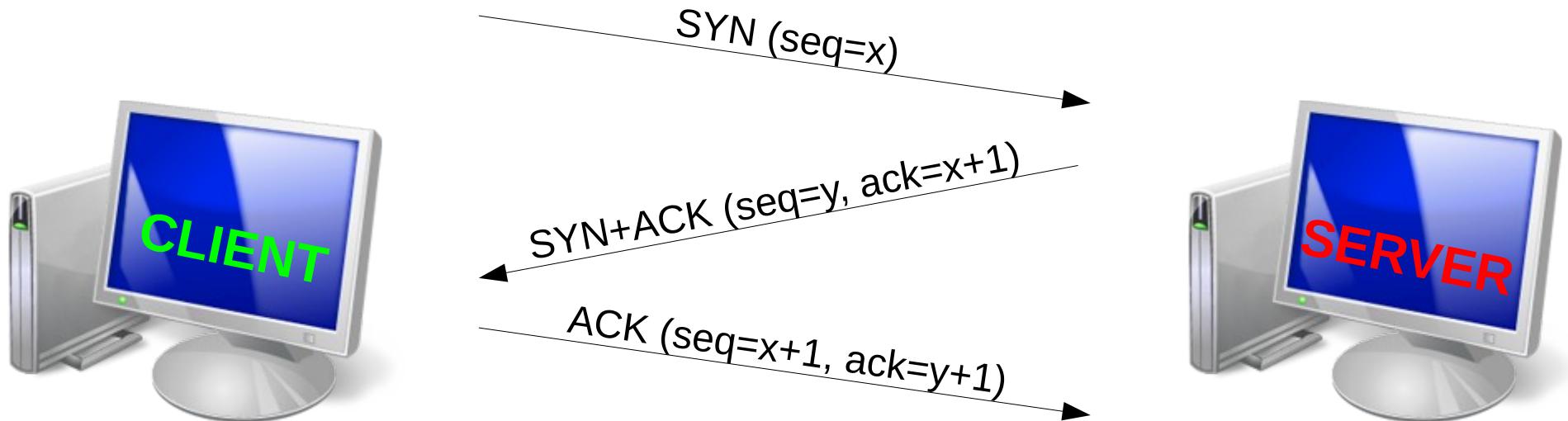
<http://itpro.nikkeibp.co.jp/article/lecture/20070305/263897/>



TCP header vs UDP header



TCP connection handshake





Services relying on TCP

- FTP on port 20 and 21
- SSH on port 22
- Telnet on port 23
- SMTP on port 25
- HTTP on port 80
- IMAP on port 143
- SSL on port 443

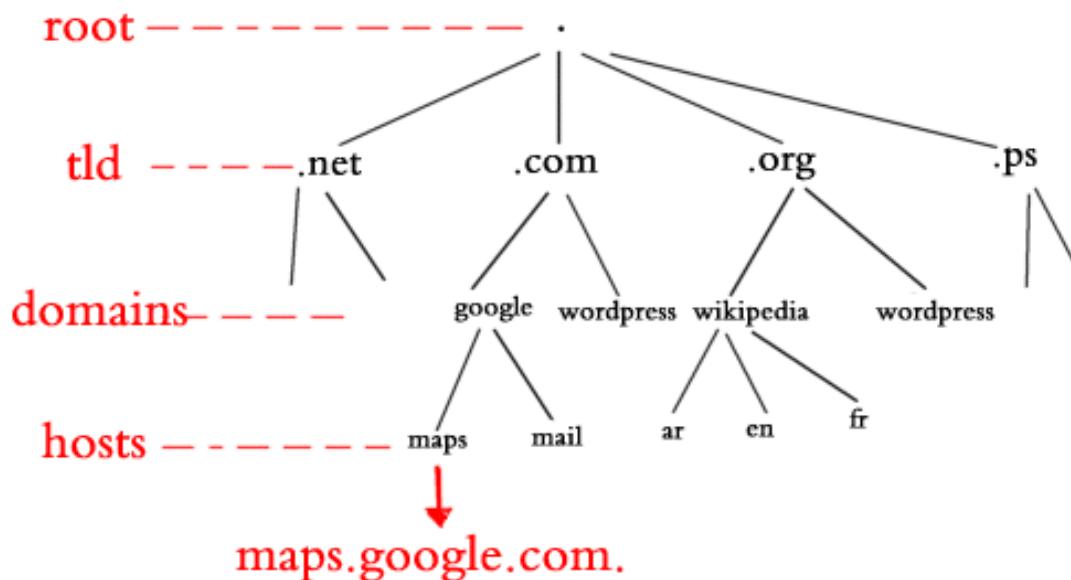


Services relying on UDP

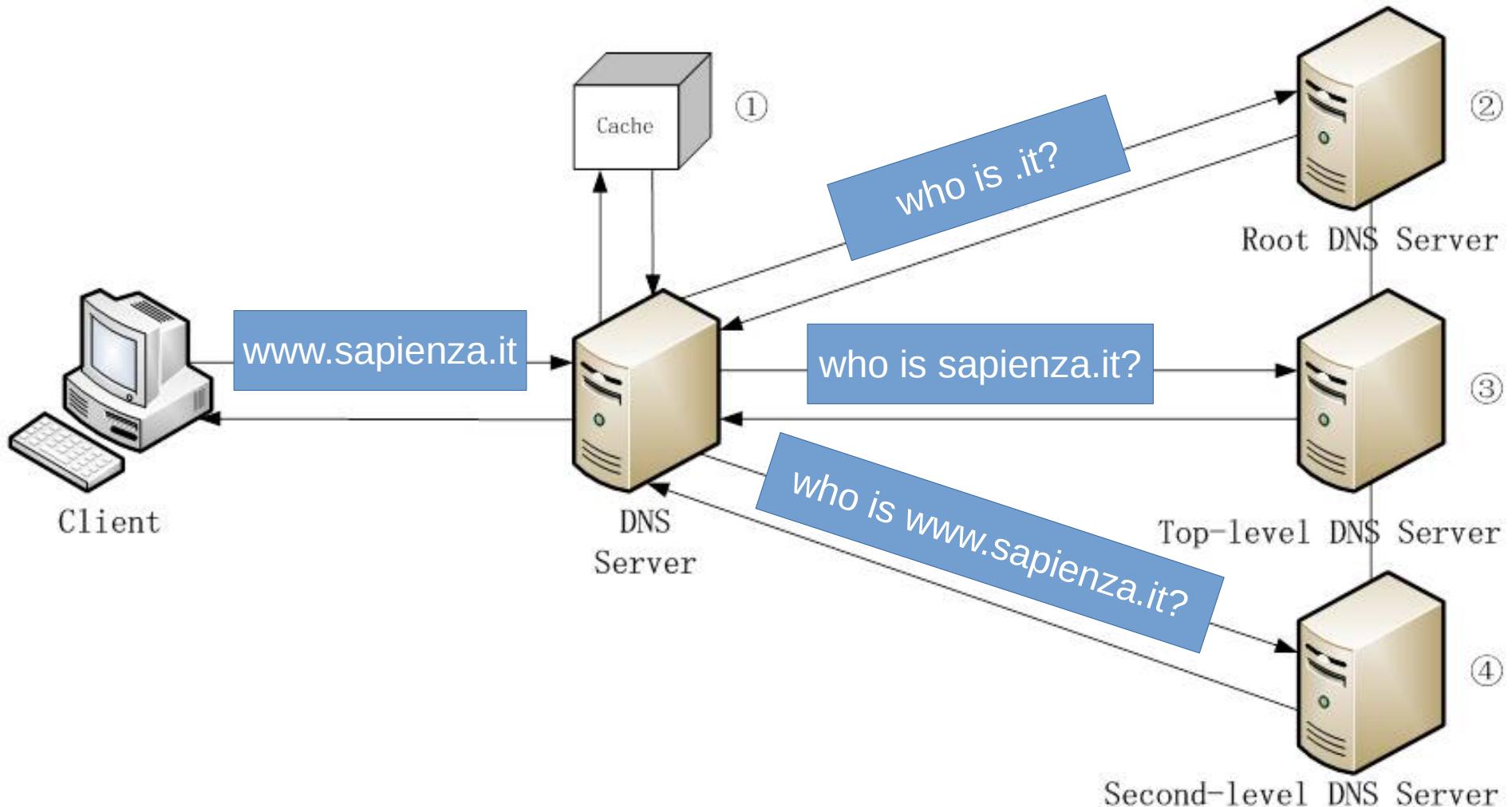
- DNS on port 53
- DHCP on ports 67 and 68
- TFTP on port 69
- SNMP on port 161
- RIP on port 520

DNS

- A service to get the IP address from an human-friendly domain name, like `www.sapienza.it`
- Hierarchy of entities responsible for domain names



DNS query example





Dive into packets



Capture packets

- Packets flow in the network, to capture them use a network traffic dump tool, like:
 - **dumpcap**
 - **wireshark/tshark** (<https://www.wireshark.org/docs/>)
 - **tcpdump**
- All based on the *pcap* (*winpcap* in Windows) library
- All of them can visualize and save the captured data
- Wireshark and tcpdump can also **analyze (decode)** the captured packets



Wireshark

- Data from a network interface are “dissected” in frames, segments, and packets, understanding where they begin and end
- Then, they are interpreted and visualized in the context of the recognized protocol
- **Promiscuous mode** (also called monitor mode) is required to capture packets not intended for the capturing host
- Best suited for
 - Looking for the root cause of a known problem
 - Searching for a certain protocol or stream between devices
 - Analyzing specific timing, protocol flags, or bits on the wire
 - Following a conversation between two devices
- It shouldn't be the first tool thought of early on in discovering a problem, but solving a problem...



Logic of wireshark

- Frames are collected from the interface and passed to several, consecutive, “dissectors”, one for each layer
- Frames pass from bottom layer to upper layer
- Protocols can be detected in two ways:
 - directly, if a frame (e.g. Ethernet) has the field that states which protocol it is encapsulating
 - indirectly, with tables of protocol/port combinations and heuristics
 - Usually working, troubles when protocols are used in nonstandard ports



Alternative way to capture traffic info

- Traffic represented as “connections”
- Netflow
 - For statistics and monitoring
 - Netflow v9 <https://www.ietf.org/rfc/rfc3954.txt>
- Zeek (formerly known as Bro)
 - Framework for traffic inspection and monitoring
 - Scripting engine to enable immediate processing

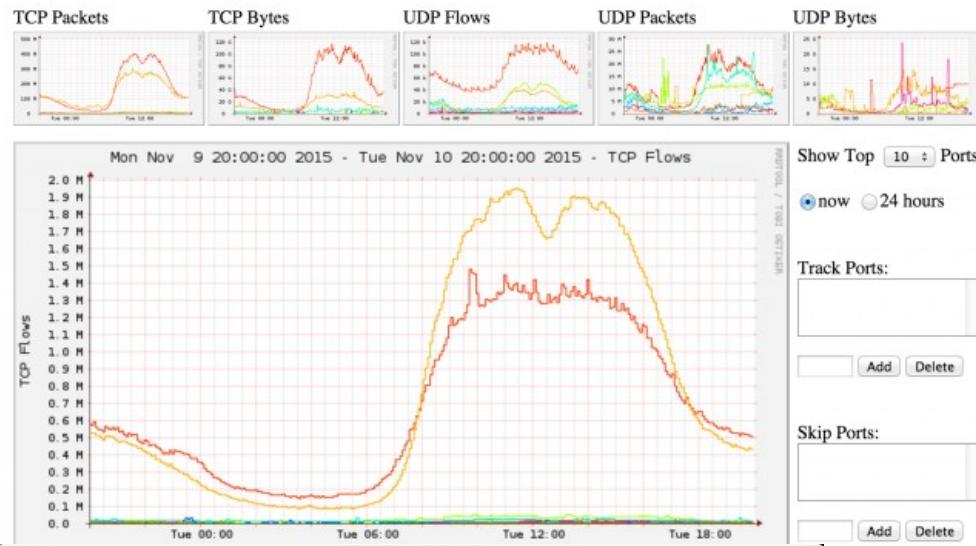


Netflow

- Suite of tools:
 - nfcapd
 - Capture and save netflows
 - nfdump
 - Analyze netflow files (tcpdump-style)
 - nfsen
 - Graphical tool to access captured netflows
 - It uses nfdump as back end

NfSEN

Port Tracker



Conditions based on individual Top 1 statistics:
 Conditions based on plugin:

Trigger:
 Each time after x condition = true, and block next trigger for cycles

Action:
 No action
 Send alert email To: haag@switch.ch Subject: DoSflows ix1 alert triggered
 Call plugin:

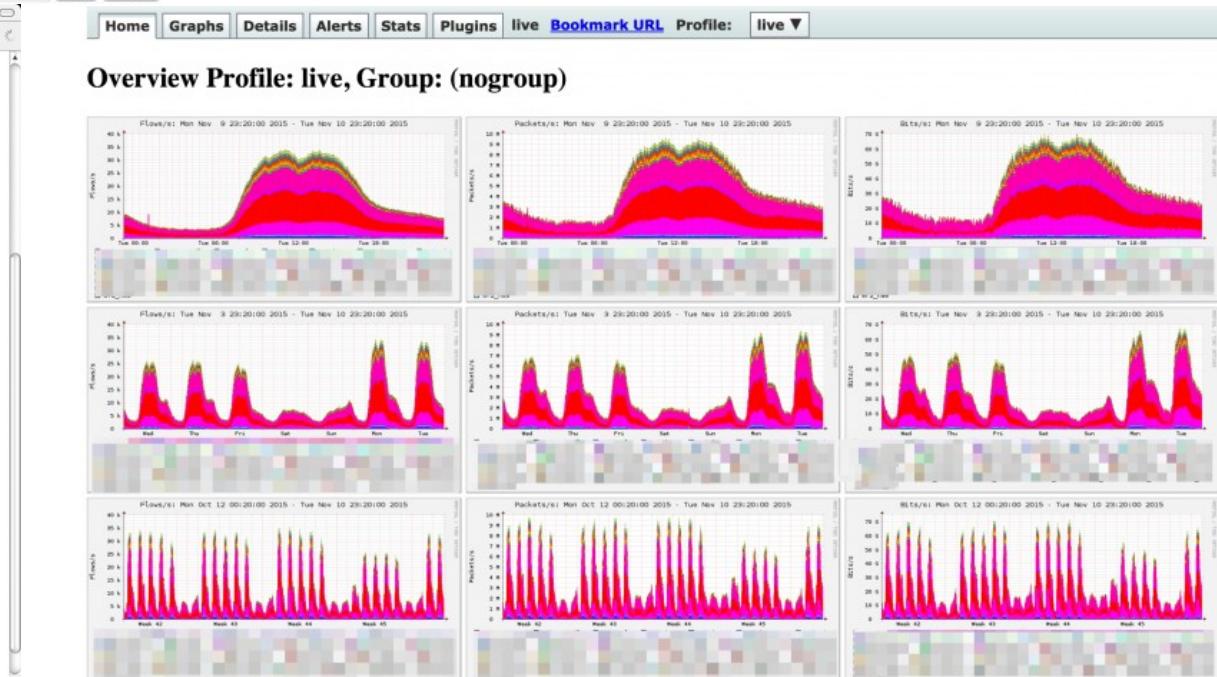
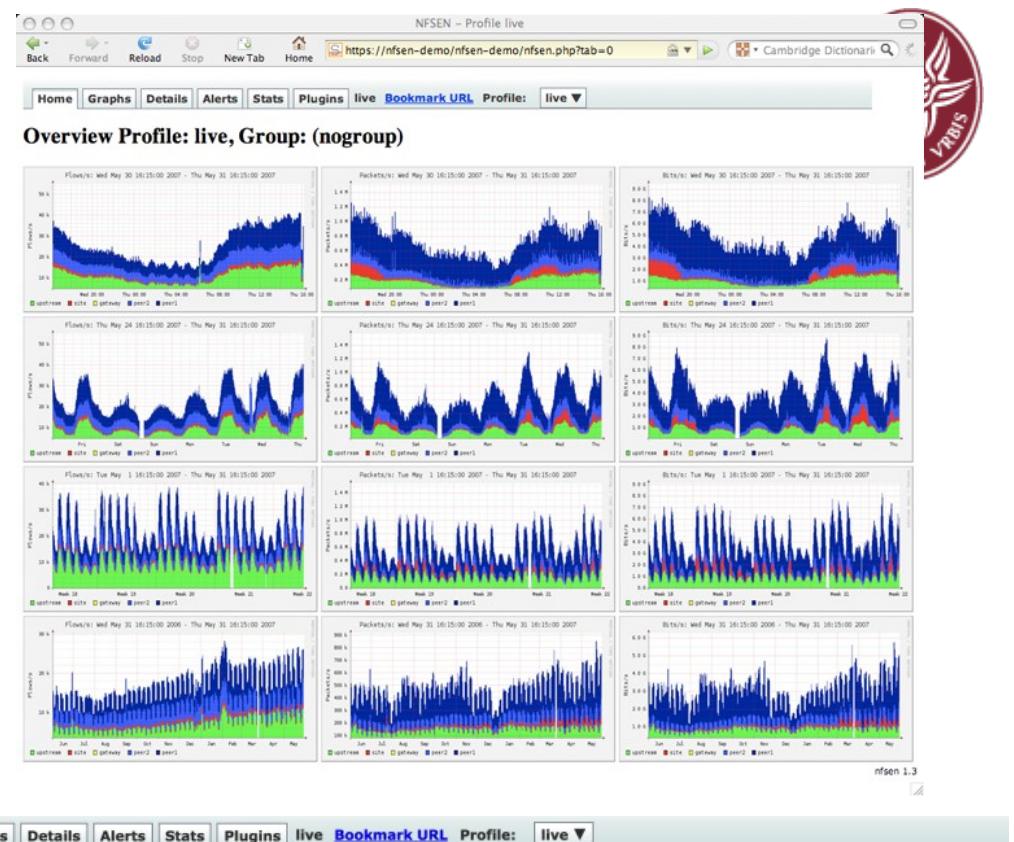
Alert Infos:
 Last cycle: 2007-05-31 16:45:45

The graph shows Flows/s over time from Wednesday 20:00 to Thursday 16:00. It includes several colored lines representing different average time periods (avg10m, avg30m, avg1h, avg5h, avg12h, avg24h) and a blue line for the last hour. A vertical red line marks a significant peak around 08:00 on Thursday.

	Last	Avg 10m	Avg 30m	Avg 1h	Avg 5h	Avg 12h	Avg 24h
Flows	4.2 M	4.4 M	4.4 M	4.6 M	4.6 M	3.8 M	3.2 M
Packets	14.0 k/s	14.5 k/s	14.7 k/s	15.5 k/s	15.2 k/s	12.5 k/s	10.8 k/s
Bytes	78.0 M	82.2 M	83.7 M	85.2 M	83.2 M	65.0 M	56.9 M

	Last	Avg 10m	Avg 30m	Avg 1h	Avg 5h	Avg 12h	Avg 24h
Flows	260.1 k/s	274.1 k/s	278.9 k/s	284.0 k/s	277.4 k/s	216.8 k/s	189.5 k/s
Packets	53.6 GB	56.5 GB	58.0 GB	58.8 GB	57.7 GB	45.9 GB	40.3 GB
Bytes	1.4 GB/s	1.5 Gb/s	1.5 Gb/s	1.6 Gb/s	1.5 Gb/s	1.2 Gb/s	1.1 Gb/s

Conditions: 0 1 2 Final: State: False False False False





Summary

- Packets: made of stacked layers
 - Each layer has its own role in the communication
- Encapsulation is the rule
 - Protocol encapsulates other protocols as their data
- How to capture packets?
 - It depends on the final goal: monitoring, statistics, debugging, security
- Wireshark: dissecting packets
 - Very powerful tool to explore and dive into packets



Wireshark activity



Using wireshark

- Capturing is way too easy... Too many packets!
 - <https://wiki.wireshark.org/CaptureSetup/CapturePrivileges>
- To survive, use filters!
 - They allow to only focus on requested packets or certain activity by network devices
- Two kinds of filters: **display filters** and **capture filters**
 - Capture filters to limit the amount of network data that goes into processing and is getting saved
 - Display filters to inspect only the packets you want to analyze once the data has been processed



Capture filters – wireshark/tcpdump

- Limit the traffic captured and, optionally, analyzed
 - Packets not captured are lost...
- Berkeley Packet Filter (BPF) syntax (`man pcap-filter`)

protocol direction type

 - Protocol: ether, tcp, udp, ip, ip6, arp
 - Direction: src, dst
 - Type: host, port, net, portrange
 - Other primitives: less, greater, gateway, broadcast
 - Combinations with operators: and (&&), or (||), not (!)



Display filters – wireshark

- Display only captured packets matching the filters
 - Packets are not discarded or lost
- Easy but refined syntax: only packets evaluating true are displayed
 - Comparison operators
 - Filters use types (strings where numbers are required return errors)
 - Common logical operators
- Filters can be built interacting with the packets

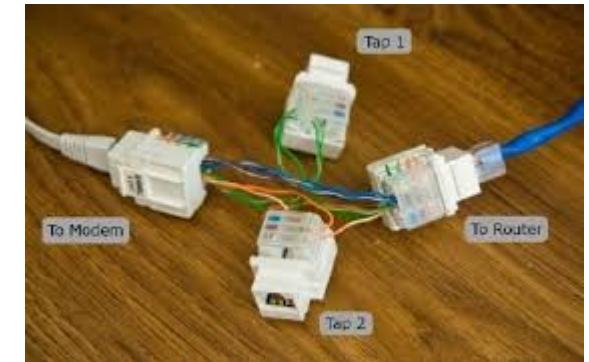


Logic of wireshark

- Frames are collected from the interface and passed to several, consecutive, “dissectors”, one for each layer
- Frames pass from bottom layer to upper layer
- Protocols can be detected in two ways:
 - directly, if a frame (e.g. Ethernet) has the field that states which protocol it is encapsulating
 - indirectly, with tables of protocol/port combinations and heuristics
 - Usually working, troubles when protocols are used in nonstandard ports

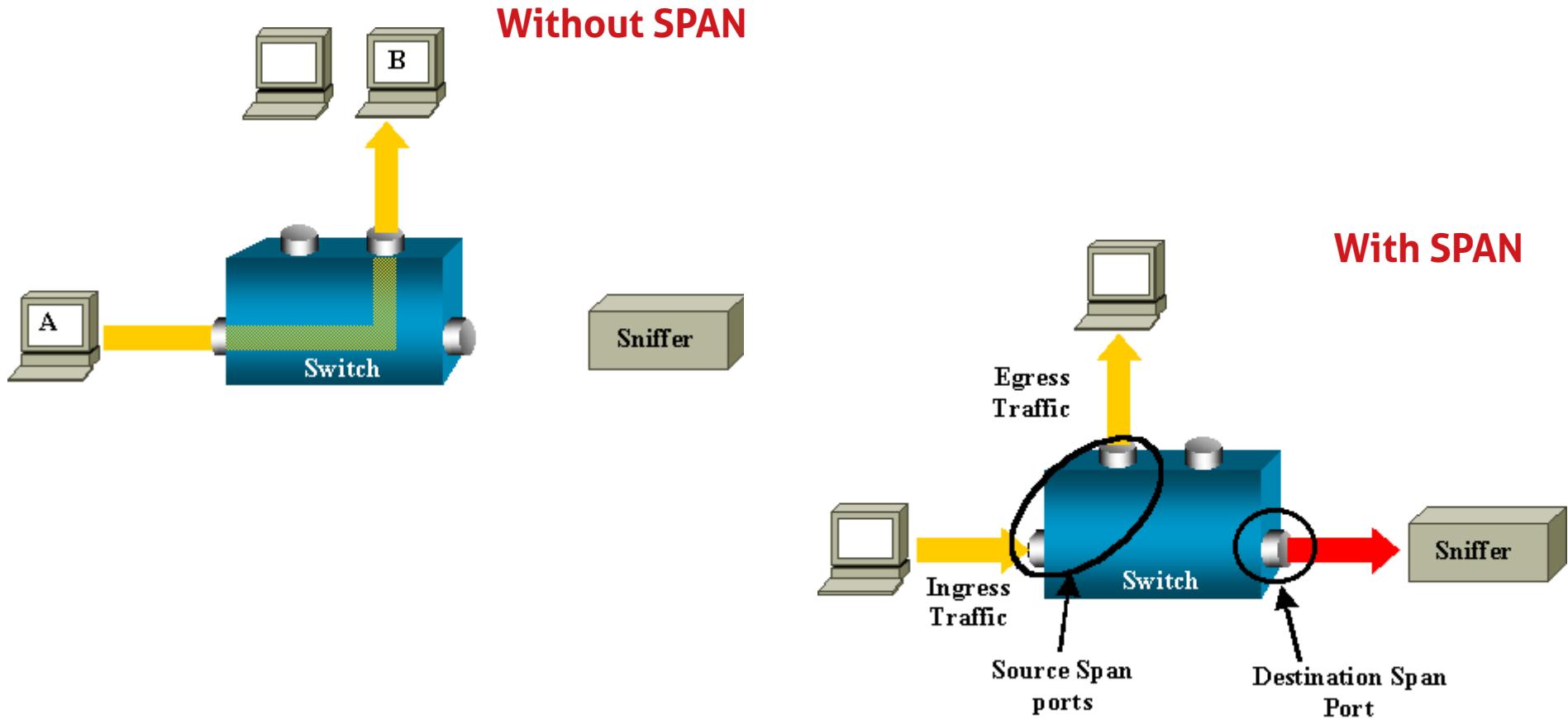
How to capture network traffic

- Promiscuous mode
 - Limitations?
 - Remember the difference between hubs and switches!
- Physical tap
- Port mirroring on a managed switch
- More “aggressive” approaches:
 - ARP cache poisoning
 - MAC flooding
 - DHCP redirection
 - Redirection and interception with ICMP
- NOTICE: on virtualized environments and SDN, this can be easier or harder



Port mirroring

- Switched Port Analyzer (**SPAN**) or Roving Analysis Port (**RAP**)





Less conventional approaches for sniffing

- ARP cache poisoning (or spoofing)
 - Unsolicited ARP replies to steal IP addresses (ettercap, Cain&Abel)
- MAC flooding
 - Fill the CAM of the switch to make it act as a hub (macof)
- DHCP redirection
 - Rogue DHCP server: it exhausts the IP addresses of the pool
 - Then pretends to be the default gateway of the network with new DHCP requests (Gobbler, DHCPstarv, Yersinia)
- Redirection and interception with ICMP
 - ICMP type 5 (redirect) used to indicate a better route (ettercap)



How to prevent packet capture

- **Dynamic address inspection**
 - Implemented in switches: Dynamic Address Resolution Inspection (DAI) validates ARP packets
 - IP-to-MAC address binding inspection, drop invalid packets
- **DHCP snooping**
 - Implemented in switches: distinguishes between trusted and untrusted ports and uses a database of IP-to-MAC
 - Ports that show rogue activity can also be automatically placed in a disabled state



Additional setup

- Configure the GeolP resolver
 - <https://wiki.wireshark.org/HowToUseGeolP>
 - Sign and download the GeoLite2 MaxMind free database(s)
 - Unzip the files in a directory
- In wireshark:
 - Edit→Preferences→Name Resolution
 - Select MaxMind database directories
- Now you can use filters like

```
ip.geoip.country eq "China"
```



Activity 1: pnd-labs/lab1/ex4

- Download the package <https://github.com/vitome/pnd-labs.git>
- Run tcpdump and save the captured traffic in an output file
 - Best option: into the `/hosthome/` directory
- Use the browser for connecting to the webserver in `pnd-labs/lab1/ex4/pc1`
 - Browse page `ba.php`
user=angelo psw=angsp
- Stop tcpdump
- Repeat the procedure with another outfile
 - Browse page `da.php`
user=angelo psw=angsp
- Use wireshark to analyze and compare the captured files



Try to use also virtual interfaces

- add (or del) a virtual interface (pair veth0@veth1):
 - `ip link add dev veth0 type veth peer name veth1`
- connect one veth end to the virtual bridge:
 - `ip link set master br0 dev veth1`
- assign an IP address to the other end (not enslaved):
 - `ip addr add x.x.x.x/y dev veth0`
- enable both the ends of the virtual interface
 - `ip link set veth0 up`
 - `ip link set veth1 up`
- A script can be found in the pnd-labs folder



Activity 2

- Run tcpdump and save the captured traffic in an output file
- Connect via ftp to an open ftp server
 - e.g.: **test.rebex.net** (demo:password)
- Stop tcpdump
- Repeat the procedure with another outfile, connecting with sftp to:
 - **test.rebex.net** (demo:password)
- Use wireshark to analyze and compare the captured files
- Use wireshark FILTERS of ftp to look for user/password



Activity 3: pnd-labs/lab1/ex2

- Capture the DHCP exchange of pnd-labs/lab1/ex2
- Run tcpdump and save the captured traffic in an output file
- Then use wireshark from the host machine to explore the captured traffic



Activity 4: pnd-labs/lab1/ex3

- Capture the traffic exchange of pnd-labs/lab1/ex3 between the hosts of the two different lans from different positions
 - lan1, lan2 and internal (between r1 and r2)
- Use tcpdump to save the captured traffic in an output files into the /hosthome/ directory
- Then use wireshark from the host machine to explore the captured traffic
- Pay attention to the layering approach and how packets change when moving from one network to the other



Activity 5

- Try to solve with wireshark the CTF of Hack3rCon 3 conference (2012)
- <http://sickbits.net/other/hc3.pcap-04.cap>



References

- Wireshark for Security Professionals: Using Wireshark and the Metasploit Framework
 - Bullok, Parker, Wiley ed.
- The Network Security Test Lab: A Step-by-Step Guide
 - Gregg, Wiley e.



That's all for today

- **Questions?**
- See you next lecture!
- References:
 - http://www.tcpipguide.com/free/t_IPDatagramGeneralFormat.htm
 - <https://www.wiley.com/en-us/Wireshark+for+Security+Professionals%3A+Using+Wireshark+and+the+Metasploit+Framework-p-9781118918210>

Practical Network Defense

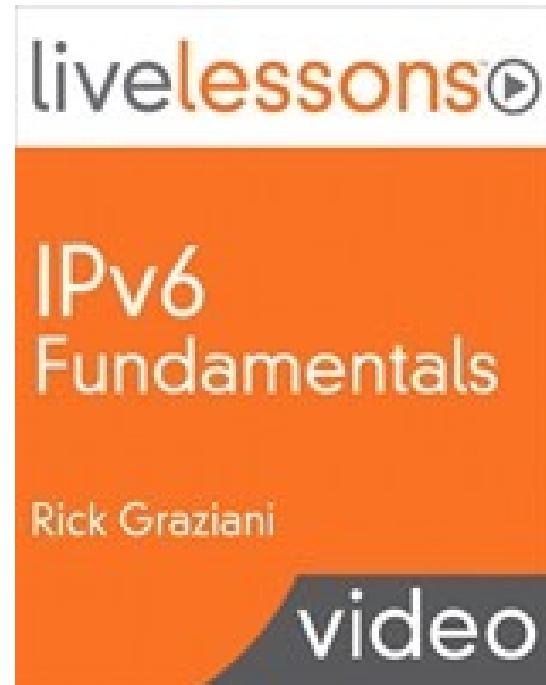
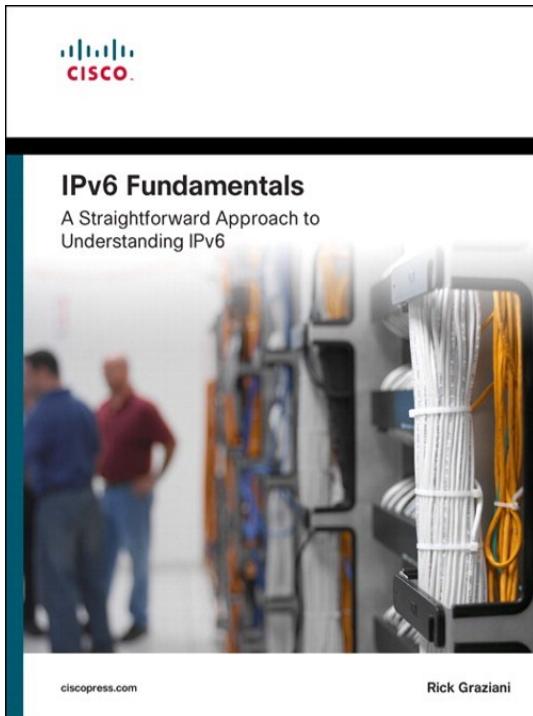
Master's degree in Cybersecurity 2021-22

IPv6: addressing

Angelo Spognardi
[*spognardi@di.uniroma1.it*](mailto:spognardi@di.uniroma1.it)

*Dipartimento di Informatica
Sapienza Università di Roma*

Material taken from Rick Graziani IPv6 courses



IPv6 Fundamentals: A Straightforward Approach to Understanding IPv6

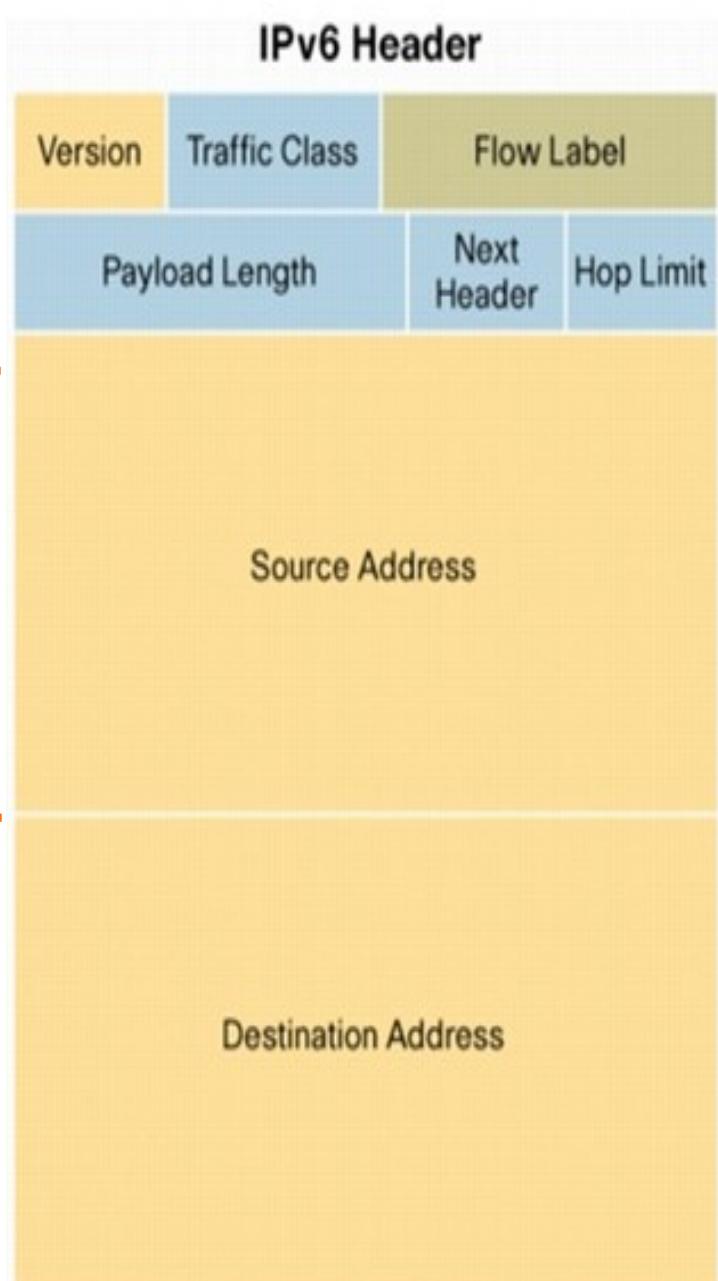
- By Rick Graziani
- ISBN-10: 1-58714-313-5

IPv6 Fundamentals LiveLessons: A Straightforward Approach to Understanding IPv6

- By Rick Graziani
- ISBN-10: 1-58720-457-6

Introducing IPv6

- Not a “new” protocol.
- Developed mid to late 1990s.
- Much learned from IPv4.
- 128-bit address space, written in hexadecimal.
- This gives us 340 undecillion addresses!



2001:DB8:CAFE:0001::100

340 undecillion

= 340,282,366,920,938,463,463,374,607,431,768,211,456

IPv6

- How many is 340 undecillion?
- 340 undecillion addresses is 10 nonillion addresses per person!
- Internet is a much different place and will continue to evolve:
 - Mobile devices
 - Video on demand
 - Internet of Everything
 - A critical part in how we “live, work, play, and learn”.

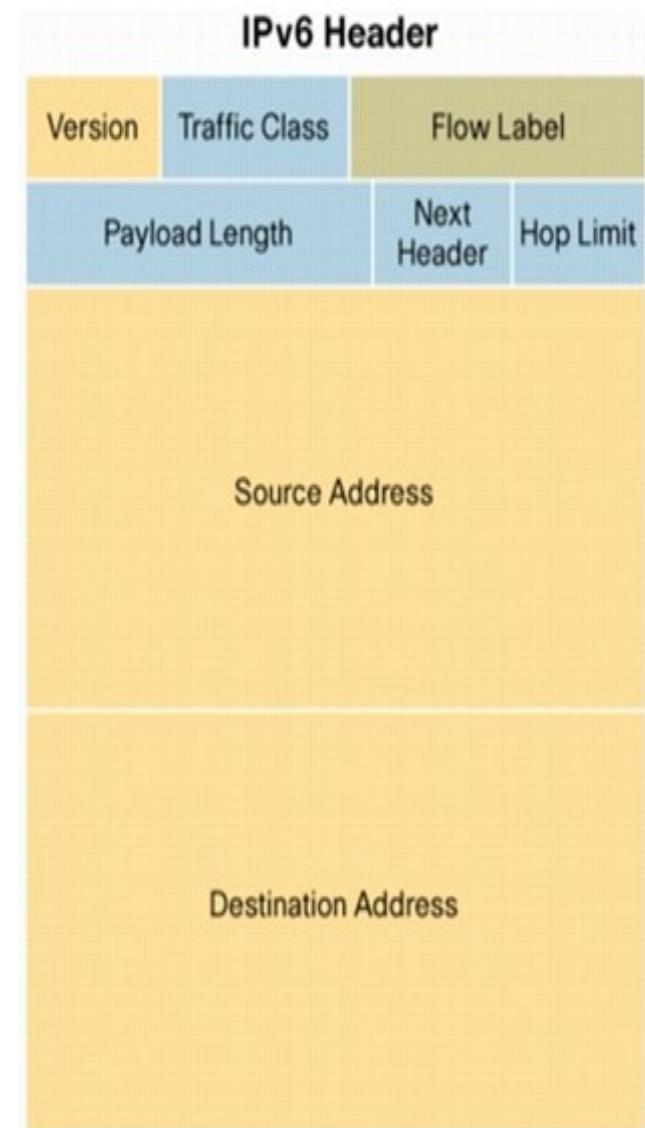


10 nonillion
 $= 10,000,000,000,000,000,000,000,000,000,000$

IPv6



- IPv6 is not just about more addresses:
 - Stateless autoconfiguration
 - End-to-end reachability without private addresses and NAT
 - Better support for mobility
 - Peer-to-peer networking easier to create and maintain, and services such as VoIP and Quality of Service (QoS) become more robust.



IPv6: A Brief History

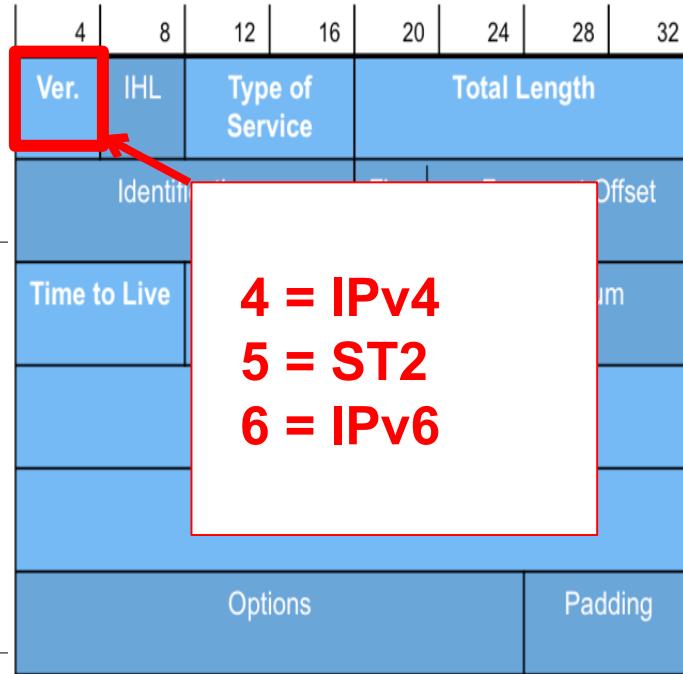


Network Working Group Request for Comments: 1883 Category: Standards Track	S. Deering, Xerox PARC R. Hinden, Ipsilon Networks December 1995
<p style="text-align: center;">Internet Protocol, Version 6 (IPv6) Specification</p>	
<p>Status of this Memo</p> <p>This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.</p>	

Network Working Group Request for Comments: 2460 Obsoletes: 1883 Category: Standards Track	S. Deering Cisco R. Hinden Nokia December 1998
<p style="text-align: center;">Internet Protocol, Version 6 (IPv6) Specification</p>	
<p>Status of this Memo</p> <p>This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.</p>	

- 1993, IETF announced a call for white papers with RFC 1550 *IP: Next Generation (IPng) White Paper Solicitation*.
- IETF chose Simple Internet Protocol Plus (SIPP) written by Steve Deering, Paul Francis, and Bob Hinden but changed the address size from 64 bits to 128 bits.
- 1995, IETF published RFC 1883 Internet Protocol, Version 6 (IPv6) Specification - later obsoleted by RFC 2460 in 1998.

What About IPv5?

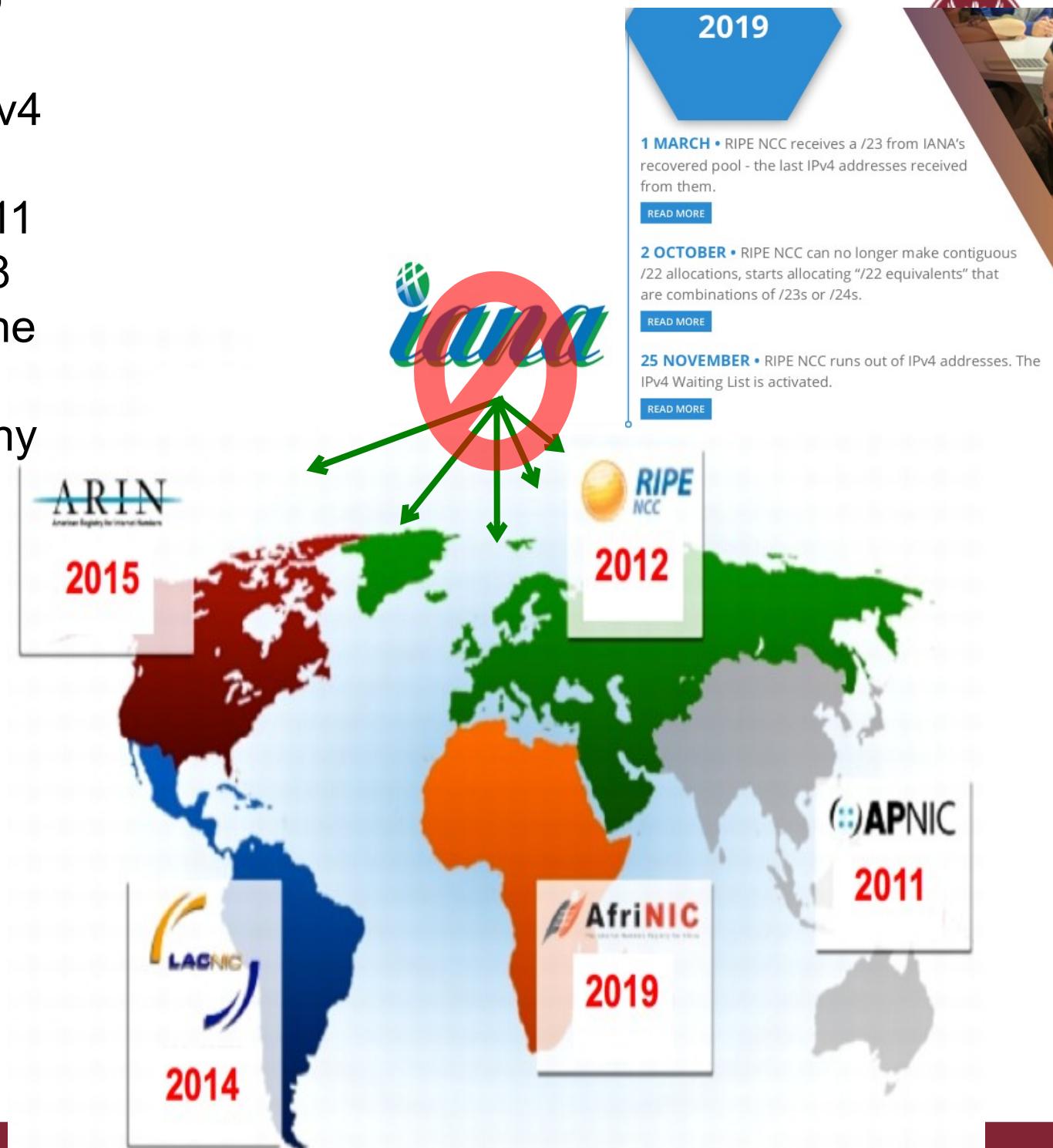


The ST packet header is not constrained to be compatible with the IP packet header, except for the IP Version Number (the first four bits) that is used to distinguish ST packets (IP Version 5) from IP packets (IP Version 4). The ST packets, or protocol data units (PDUs), can be encapsulated in IP either to provide connectivity (possibly with degraded service) across portions of an internet that do not provide support for ST, or to allow access to services such as security that are not provided directly by ST.

- In the late 1970s, a family of experimental protocols was developed intended to provide quality of service (QoS) for real-time multimedia applications such video and voice.
- Known as Internet Stream Protocol (ST) and later ST2 – (RFC 1190 and RFC 1819).
- Although it was never known as IPv5, when encapsulated in IP, ST uses IP Protocol version 5.

The Need for IPv6

- We are running out of IPv4 address space.
- Monday, January 31, 2011 IANA allocated the last /8 IPv4 address blocks to the RIRs.
- RIR's have very few, if any IPv4 address left.
- Many ISPs are severely limited and some have already run out.



Source: www.potaroo.net/tools/ipv4



Running Out of IPv4

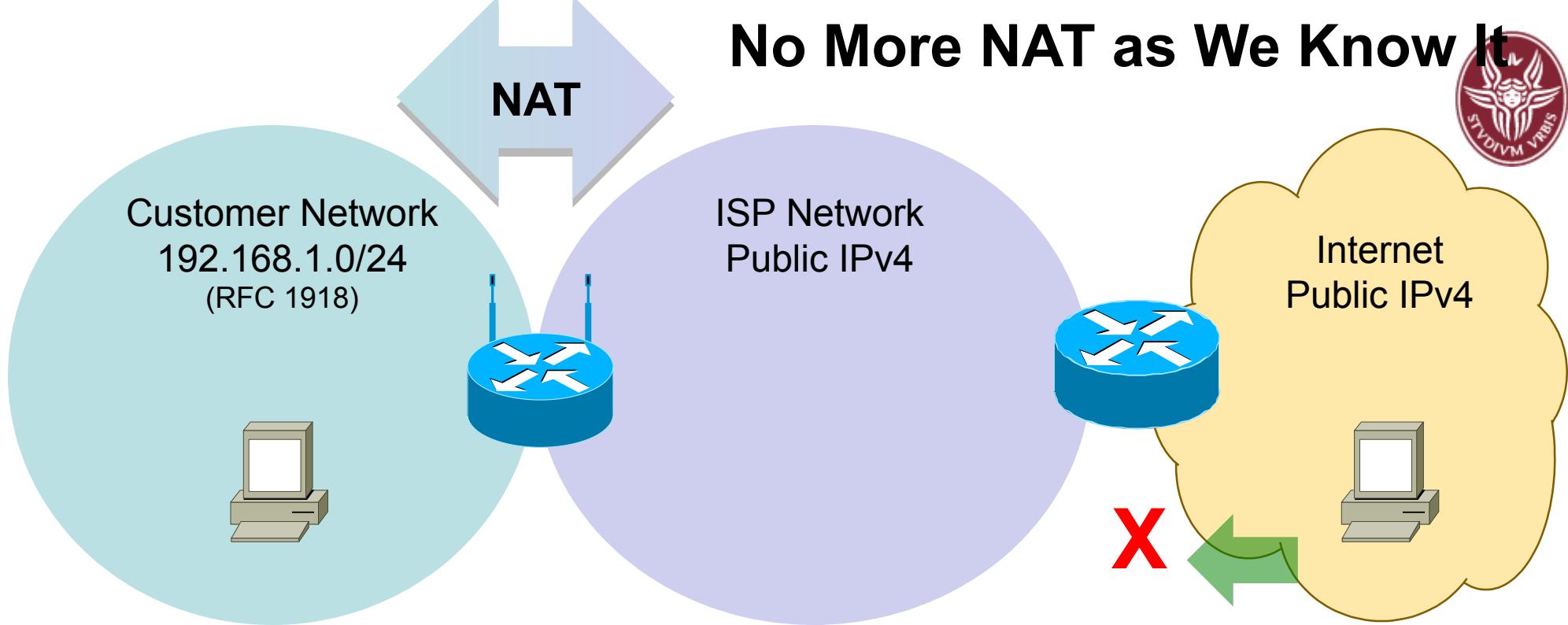
WORLD INTERNET USAGE AND POPULATION STATISTICS JUNE 30, 2014 - Mid-Year Update

World Regions	Population (2014 Est.)	Internet Users Dec. 31, 2000	Internet Users Latest Data	Penetration (% Population)	Growth 2000-2014	Users % of Table
Africa	1,125,721,038	4,514,400	297,885,898	26.5 %	6,498.6 %	9.8 %
Asia	3,996,408,007	114,304,000	1,386,188,112	34.7 %	1,112.7 %	45.7 %
Europe	825,824,883	105,096,093	582,441,059	70.5 %	454.2 %	19.2 %
Middle East	231,588,580	3,284,800	111,809,510	48.3 %	3,303.8 %	3.7 %
North America	353,860,227	108,096,800	310,322,257	87.7 %	187.1 %	10.2 %
Latin America / Caribbean	612,279,181	18,068,919	320,312,562	52.3 %	1,672.7 %	10.5 %
Oceania / Australia	36,724,649	7,620,480	26,789,942	72.9 %	251.6 %	0.9 %
WORLD TOTAL	7,182,406,565	360,985,492	3,035,749,340	42.3 %	741.0 %	100.0 %

- The regions with the largest populations have the lowest percentages of people connected to the Internet

Graphic from Internet World Stats, www.internetworkworldstats.com/stats.htm

No More NAT as We Know It



- NAT has been used to help “hide” customers and works for many client-initiated applications.
- However, NAT also creates some issues, like peer-to-peer networking and accessing our “hidden” systems from other networks.
- Using NAT to “hide” IPv6 networks has been the source of some debate.
- IETF continues to state that NAT is not a security feature.



Benefits of IPv6

The benefits of IPv6 include:

- Larger address space
- Stateless autoconfiguration
- End-to-end reachability without private addresses and NAT
- Better mobility support
- Peer-to-peer networking easier to create and maintain, and services such as VoIP and Quality of Service (QoS) become more robust.
- The “killer application” for the Internet is the Internet itself.



Graphic from IPv6 Forum, www.ipv6ready.org

Hex and IPv6 Address Representation

The Beauty of Hexadecimal: 4 bits = 1 hex digit

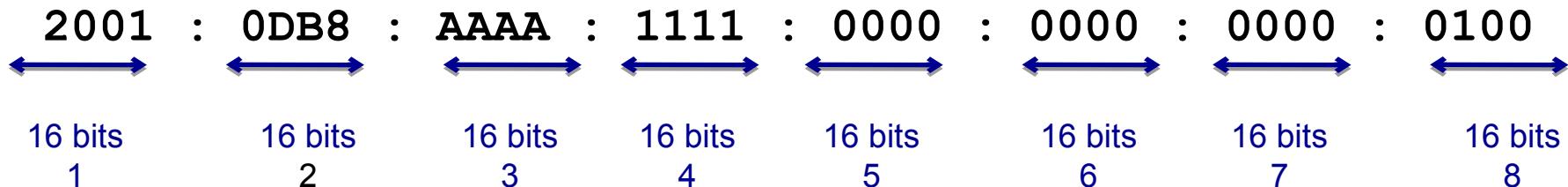


Binary			Binary		
<u>Dec</u>	<u>Hex</u>	<u>8421</u>	<u>Dec</u>	<u>Hex</u>	<u>8421</u>
0	0	0000	8	8	1000
1	1	0001	9	9	1001
2	2	0010	10	A	1010
3	3	0011	11	B	1011
4	4	0100	12	C	1100
5	5	0101	13	D	1101
6	6	0110	14	E	1110
7	7	0111	15	F	1111

IPv6 Address Notation

Dec.	Hex.	Binary	Dec.	Hex.	Binary
0	0	0000	8	8	1000
1	1	0001	9	9	1001
2	2	0010	10	A	1010
3	3	0011	11	B	1011
4	4	0100	12	C	1100
5	5	0101	13	D	1101
6	6	0110	14	E	1110
7	7	0111	15	F	1111

2001:0DB8:AAAA:1111:0000:0000:0000:0100



IPv6 addresses are 128-bit addresses represented in:

- Hexadecimal: 1 hex digit = 4 bits
- Eight 16-bit segments or “hextets” (not a formal term) between 0000 and FFFF
- Separated by colons
- Reading and subnetting IPv6 is easier than IPv4.... Almost always!

Number of IPv6 Addresses



Number name	Scientific Notation	Number of zeros
1 Thousand	10^3	1,000
1 Million	10^6	1,000,000
1 Billion	10^9	1,000,000,000
1 Trillion	10^{12}	1,000,000,000,000
1 Quadrillion	10^{15}	1,000,000,000,000,000
1 Quintillion	10^{18}	1,000,000,000,000,000,000
1 Sextillion	10^{21}	1,000,000,000,000,000,000,000
1 Septillion	10^{24}	1,000,000,000,000,000,000,000,000
1 Octillion	10^{27}	1,000,000,000,000,000,000,000,000,000
1 Nonillion	10^{30}	1,000,000,000,000,000,000,000,000,000,000
1 Decillion	10^{33}	1,000,000,000,000,000,000,000,000,000,000,000
1 Undecillion	10^{36}	1,000,000,000,000,000,000,000,000,000,000,000,000
340,282,366,920,938,463,463,374,607,431,768,211,456		

IPv4
4.3 billion

IPv4 addresses:

- 4.3 billion

IPv6 addresses:

- 340 undecillion

IPv6
340 undecillion

Two Rules for Compressing IPv6 Addresses

Rule 1: Omitting Leading 0s



- Two rules for reducing the size of written IPv6 addresses.
- **First rule:** Leading zeroes in any 16-bit segment do not have to be written.
- **Only** leading 0s can be excluded, trailing 0s must be included.

2001 : 0DB8 : 0001 : 1000 : 0000 : 0000 : 0ef0 : bc00

2001 : 0DB8 : 010d : 000a : 00dd : c000 : e000 : 0001

2001 : 0DB8 : 0000 : 0000 : 0000 : 0000 : 0000 : 0500

Two Rules for Compressing IPv6 Addresses



Rule 1: Omitting Leading 0s

- Two rules for reducing the size of written IPv6 addresses.
- **First rule:** Leading zeroes in any 16-bit segment do not have to be written.
- **Only** leading 0s can be excluded, trailing 0s must be included.

2001 : 0DB8 : 0001 : 1000 : 0000 : 0000 : 0ef0 : bc00
2001 : DB8 : 1 : 1000 : 0 : 0 : ef0 : bc00

2001 : 0DB8 : 010d : 000a : 00dd : c000 : e000 : 0001
2001 : DB8 : 10d : a : dd : c000 : e000 : 1

2001 : 0DB8 : 0000 : 0000 : 0000 : 0000 : 0000 : 0500
2001 : DB8 : 0 : 0 : 0 : 0 : 0 : 500

Two Rules for Compressing IPv6 Addresses

Rule 2: Double Colon ::



- The second rule can reduce this address even further:
- **Second rule:** Any single, contiguous string of one or more 16-bit segments consisting of all zeroes can be represented with a double colon (::).

2001 : 0DB8 : 1000 : 0000 : 0000 : 0000 : 0000 : 0001

Two Rules for Compressing IPv6 Addresses



Rule 2: Double Colon ::

- The second rule can reduce this address even further:
- **Second rule:** Any single, contiguous string of one or more 16-bit segments consisting of all zeroes can be represented with a double colon (::).

Second rule

2001 : 0DB8 : 1000 : 0000 : 0000 : 0000 : 0000 : 0001
2001 : DB8 : 1000 : : : 1

Two Rules for Compressing IPv6 Addresses



Rule 2: Double Colon ::

- The second rule can reduce this address even further:
- Second rule:** Any single, contiguous string of one or more 16-bit segments consisting of all zeroes can be represented with a double colon (::).

First rule

2001 : 0DB8 : 1000 : 0000 : 0000 : 0000 : 0000 : 0001
2001 : DB8 : 1000 :

Second rule

: 1

First rule

Two Rules for Compressing IPv6 Addresses



Rule 2: Double Colon ::

- **Second rule:** Any single, contiguous string of one or more 16-bit segments consisting of all zeroes can be represented with a double colon (::).

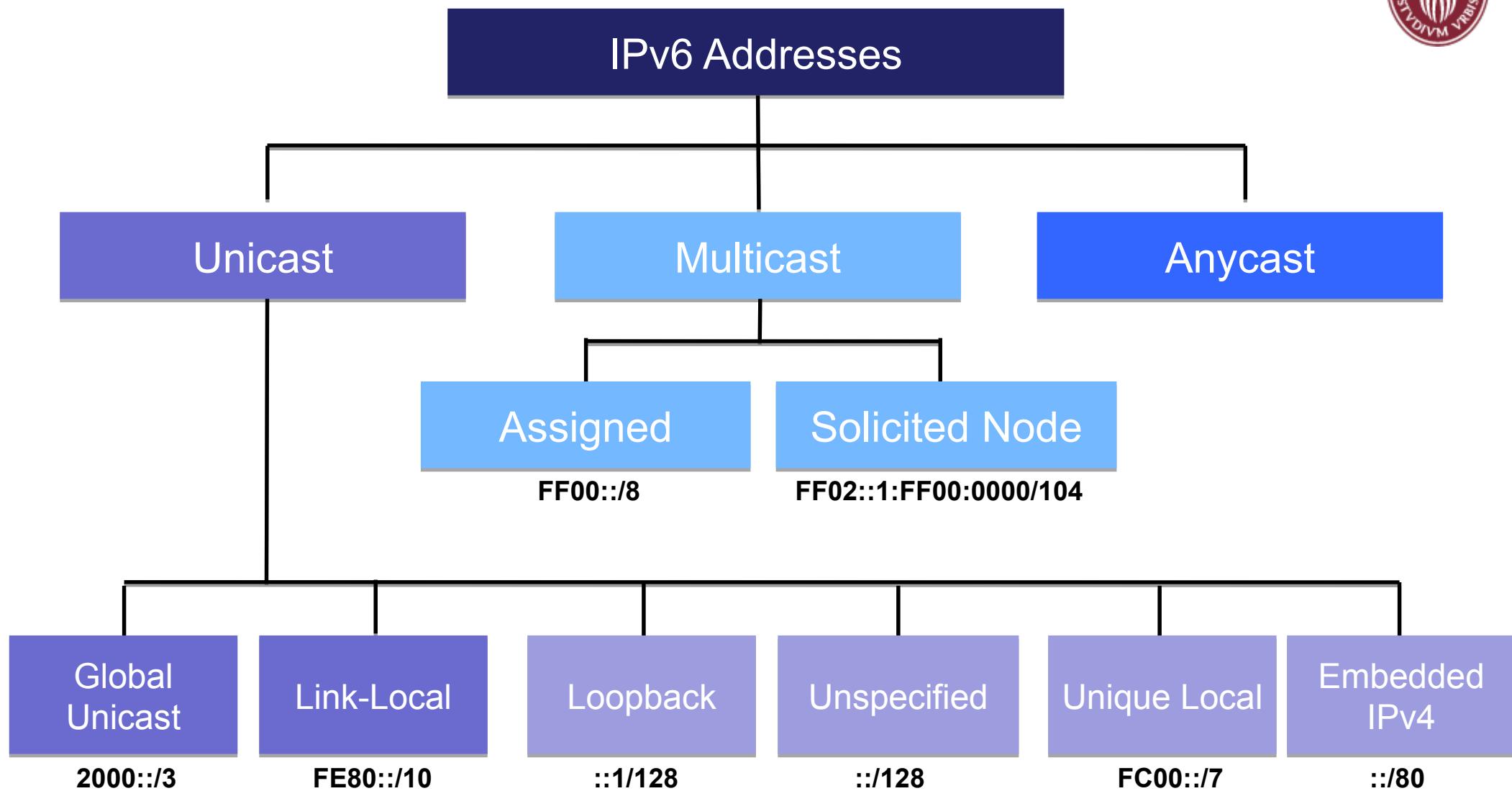
First rule	Second rule	First rule
2001 : 0DB8 : 1000 : 0000 : 0000 : 0000 : 0000 : 0001		
2001 : DB8 : 1000 :		: 1
 2001:DB8:1000::1		

If there are multiple possible reductions, RFC 5952 states that **the longest** string of zeroes must be replaced with the :: and if they are equal then only **the first string of 0's** should use the :: representation.

IPv6 Global Unicast Address

The equivalent of public IPv4 address

IPv6 Address Types

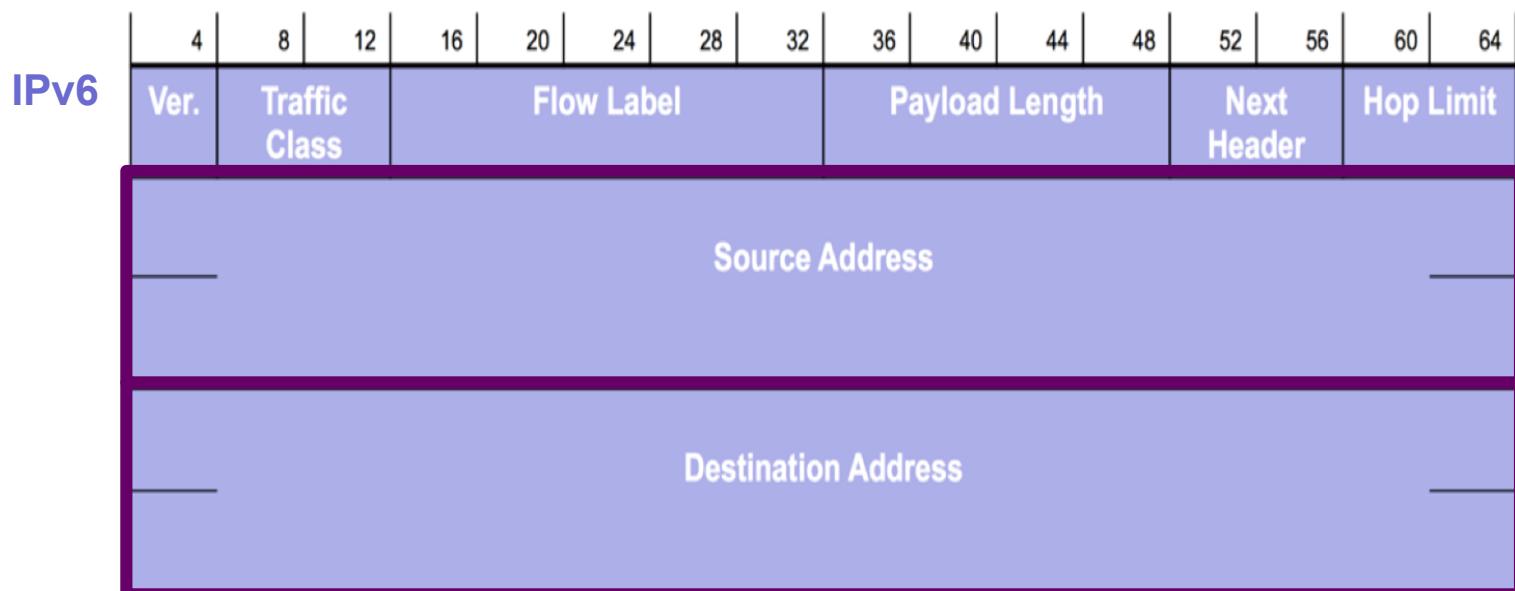
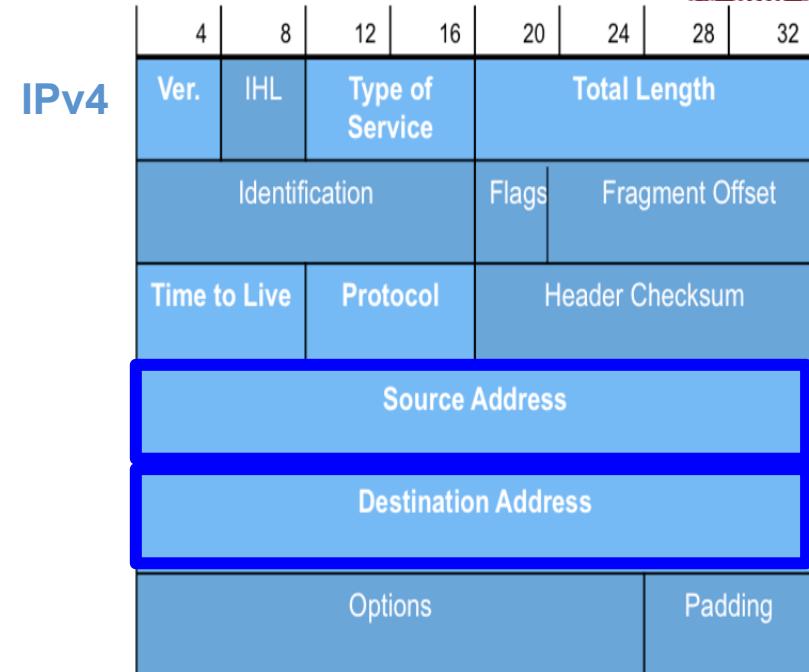


IPv6 does not have a “broadcast” address.

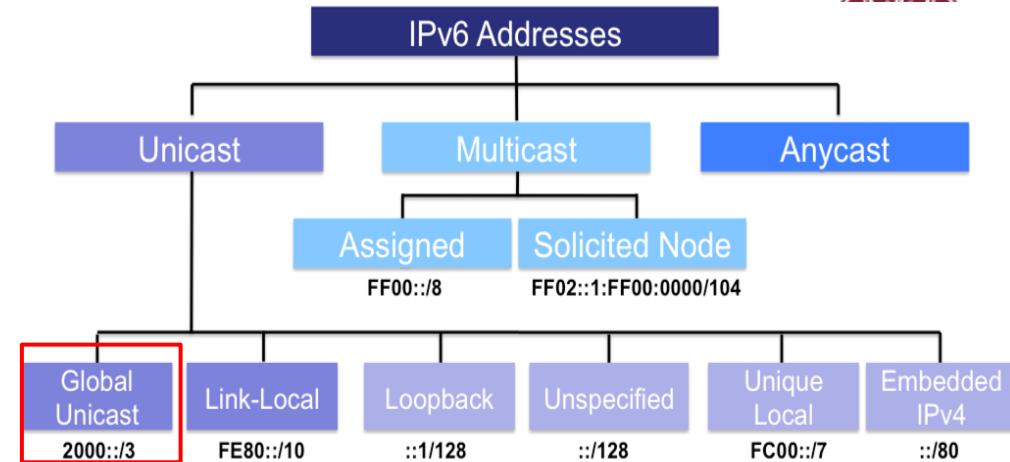
IPv6 Source and Destination Addresses



- **IPv6 Source** – Always a unicast (link-local or GUA)
- **IPv6 Destination** – Unicast, multicast, or anycast.



Global Unicast Address

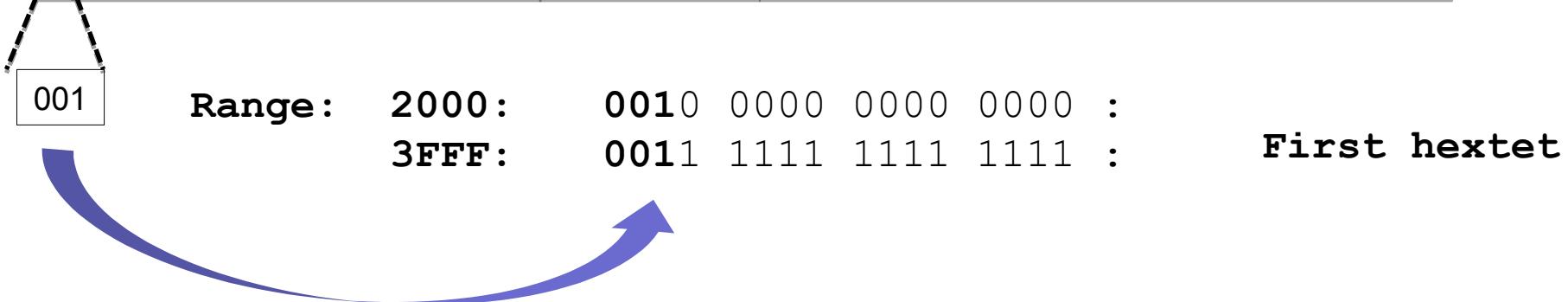


- **Global Unicast Address (GUA)**
 - 2000::/3 (First hextet: 2000::/3 to 3FFF::/3)
 - Globally unique and routable
 - Similar to public IPv4 addresses
 - 2001:DB8::/32 - RFC 2839 and RFC 6890 reserves this range of addresses for documentation
 - These are the addresses we will be referring to the most.

Global Unicast Address Range

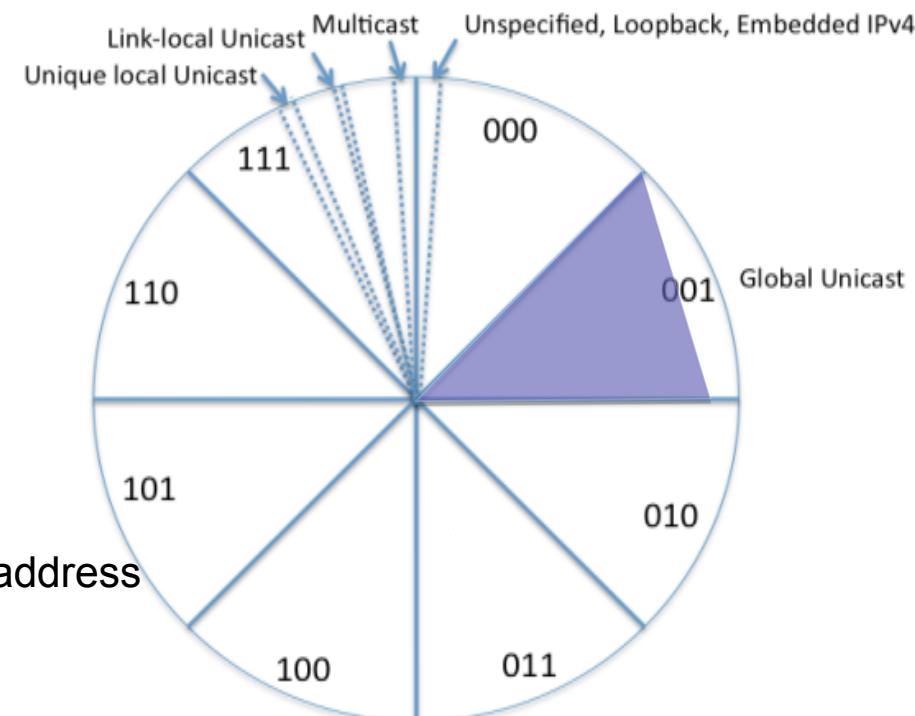


Global Routing Prefix	Subnet ID	Interface ID
-----------------------	-----------	--------------



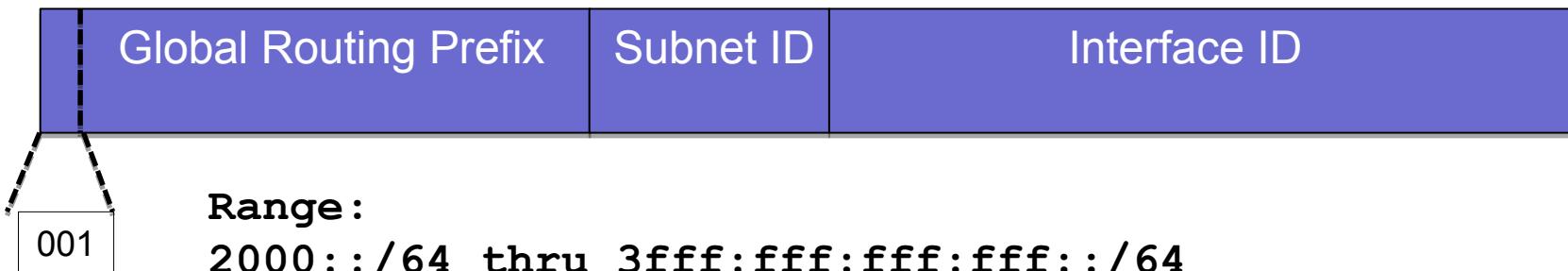
- **Global Unicast Address (GUA)**
 - 2000::/3
 - Range 2000::/64 thru 3fff:ffff:ffff:ffff::/64
 - 1/8th of IPv6 address space

IANA's allocation of IPv6 address space in 1/8th sections



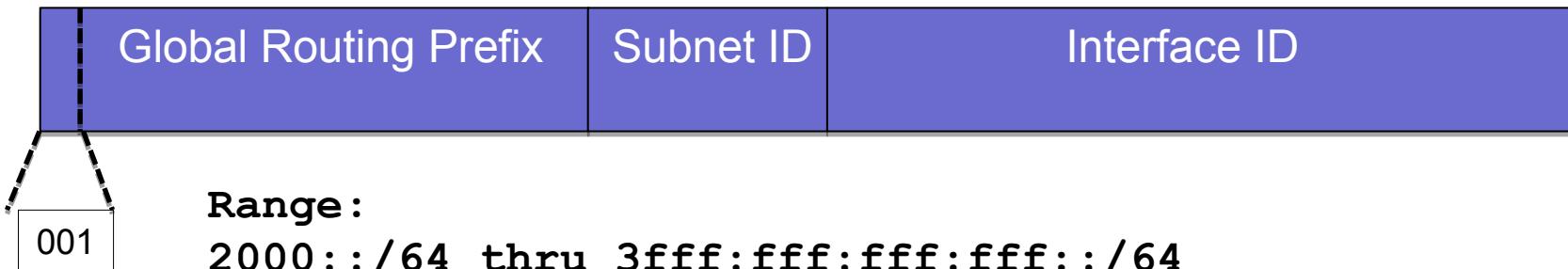
The remaining portion of IPv6 address space are reserved by IETF for future use.

Global Unicast Address Range



- Except under very specific circumstances, all end users will have a global unicast address.

Global Unicast Address Range



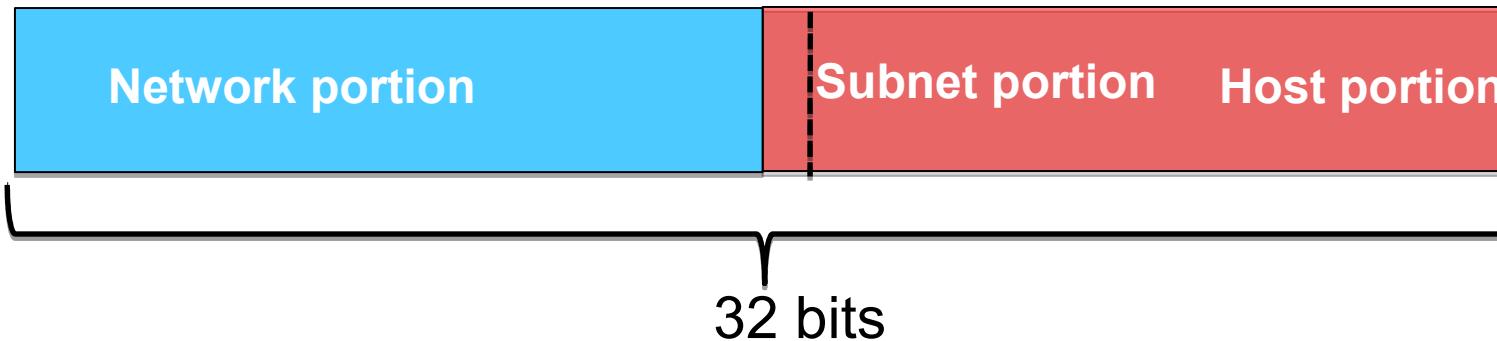
- Except under very specific circumstances, all end users will have a global unicast address.
 - **Note:** A host (an interface) can potentially have multiple IPv6 addresses on the same or different networks.
- Terminology:
 - **Prefix** equivalent to the *network address of an IPv4 address*
 - **Prefix length** equivalent to *subnet mask in IPv4*
 - **Interface ID** equivalent to *host portion of an IPv4 address*

Parts of a Global Unicast Address



IPv4 Unicast Address

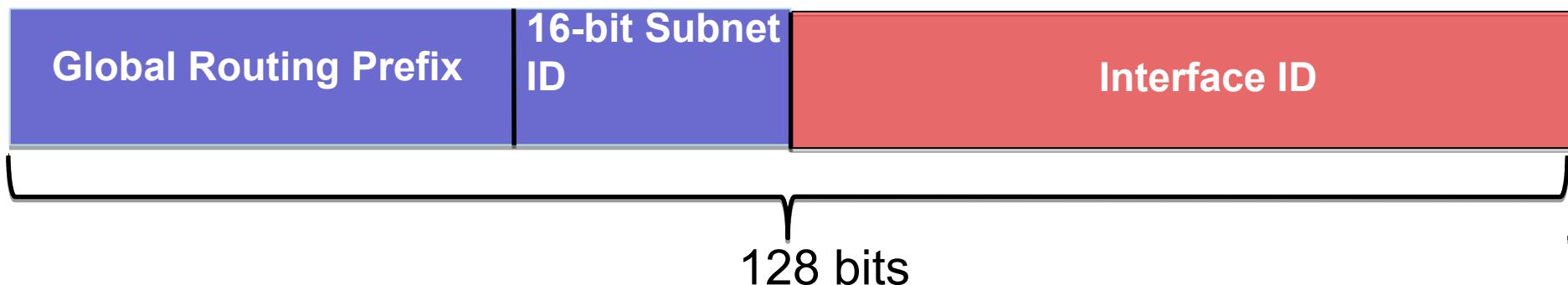
/?



IPv6 Global Unicast Address

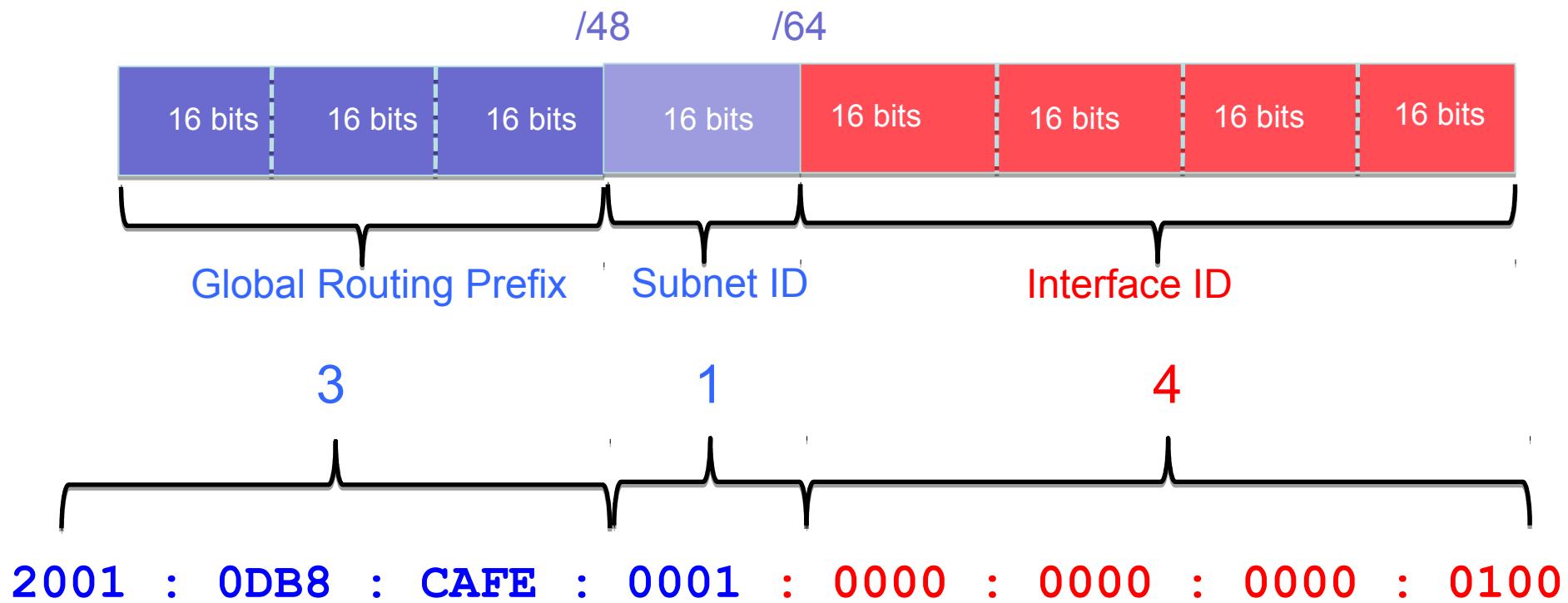
/48

/64

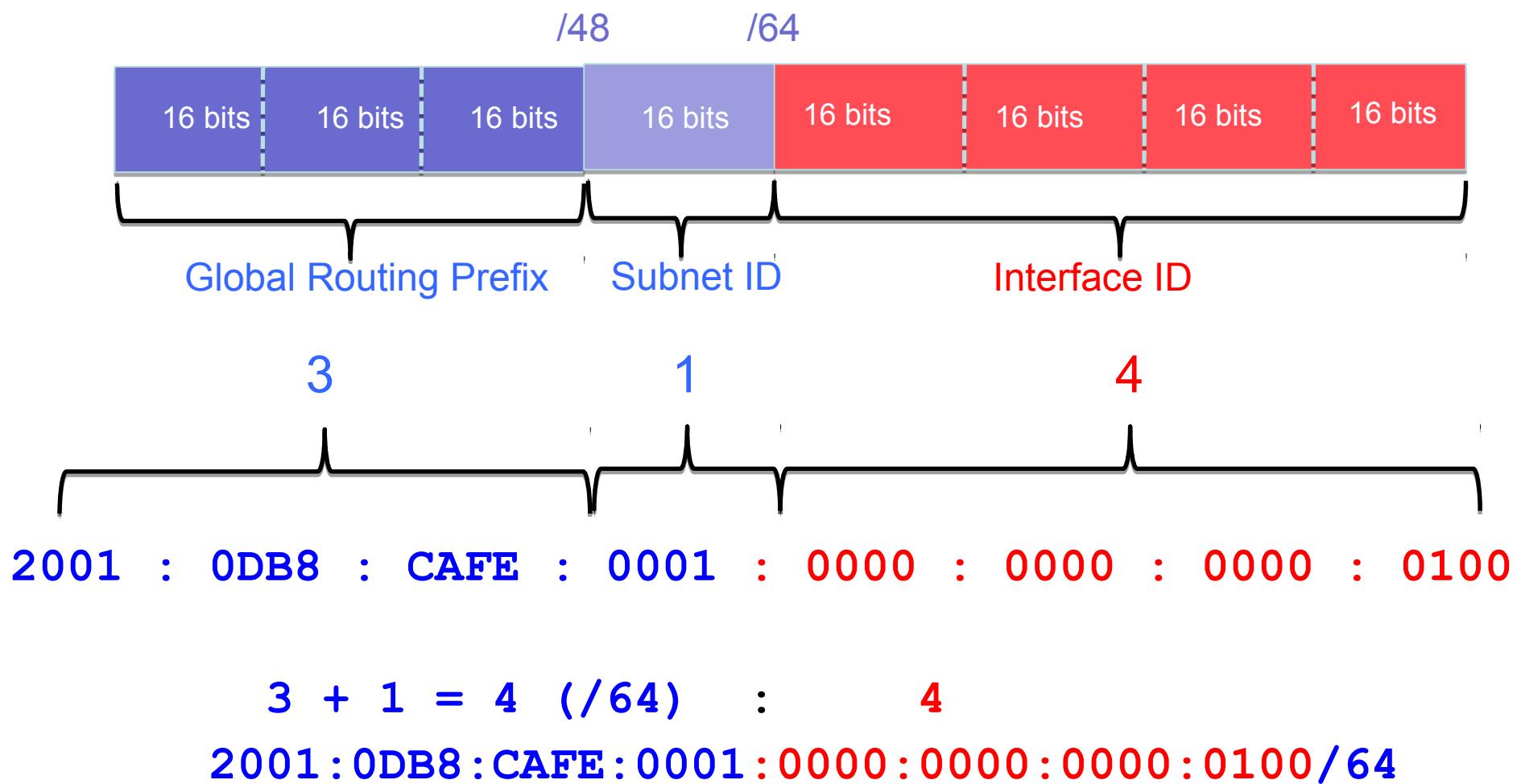


- 64-bit Interface ID = 18 quintillion (18,446,744,073,709,551,616) devices/subnet
- 16-bit Subnet ID (initially recommended) = 65,536 subnets

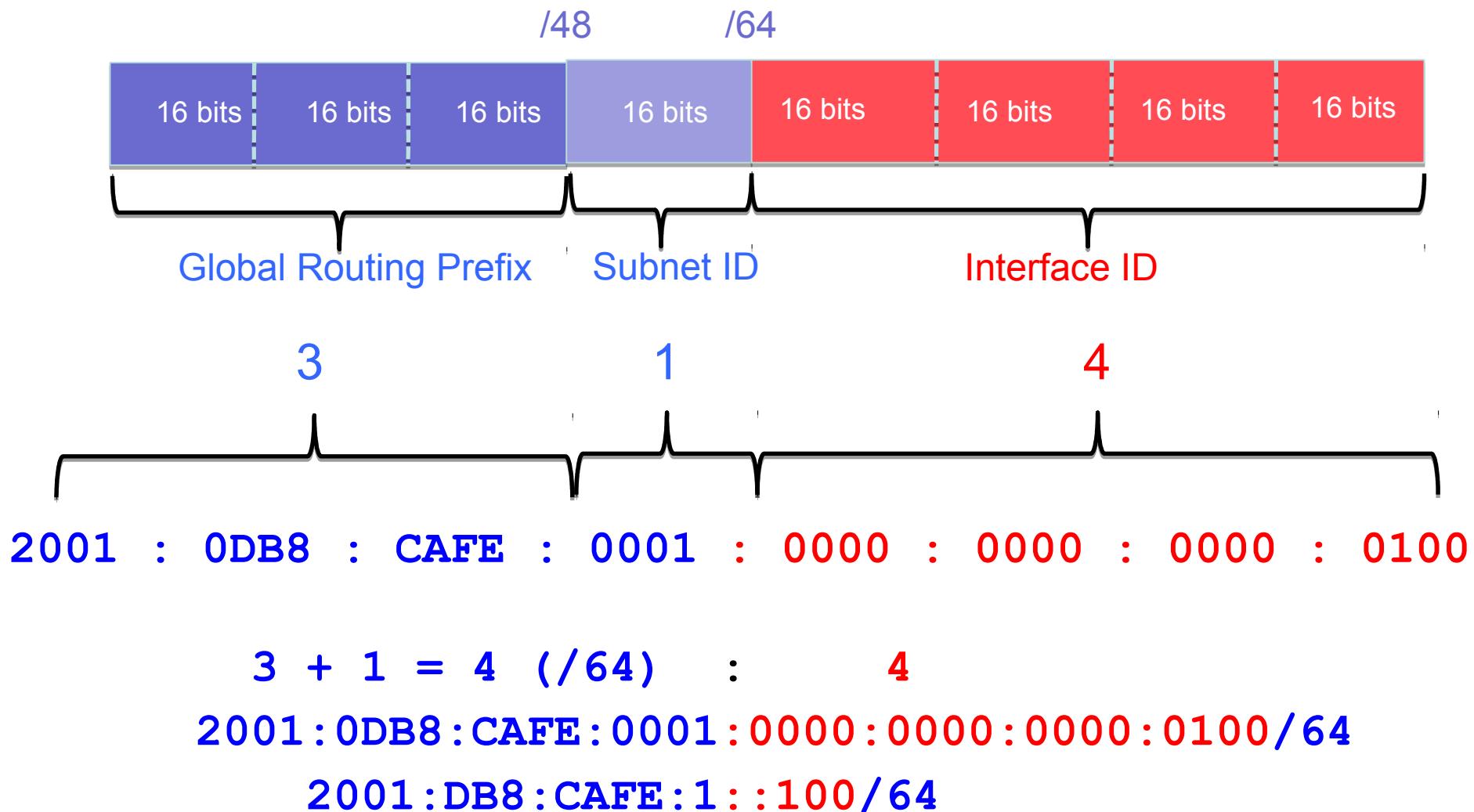
/64 Global Unicast Address and the 3-1-4 Rule



/64 Global Unicast Address and the 3-1-4 Rule



/64 Global Unicast Address and the 3-1-4 Rule



Subnetting IPv6



Can you count in hex?

Just increment by 1 in Hexadecimal



2001:0DB8:CAFE:0000::/64

2001:0DB8:CAFE:0001::/64

2001:0DB8:CAFE:0002::/64 ...

2001:0DB8:CAFE:0009::/64

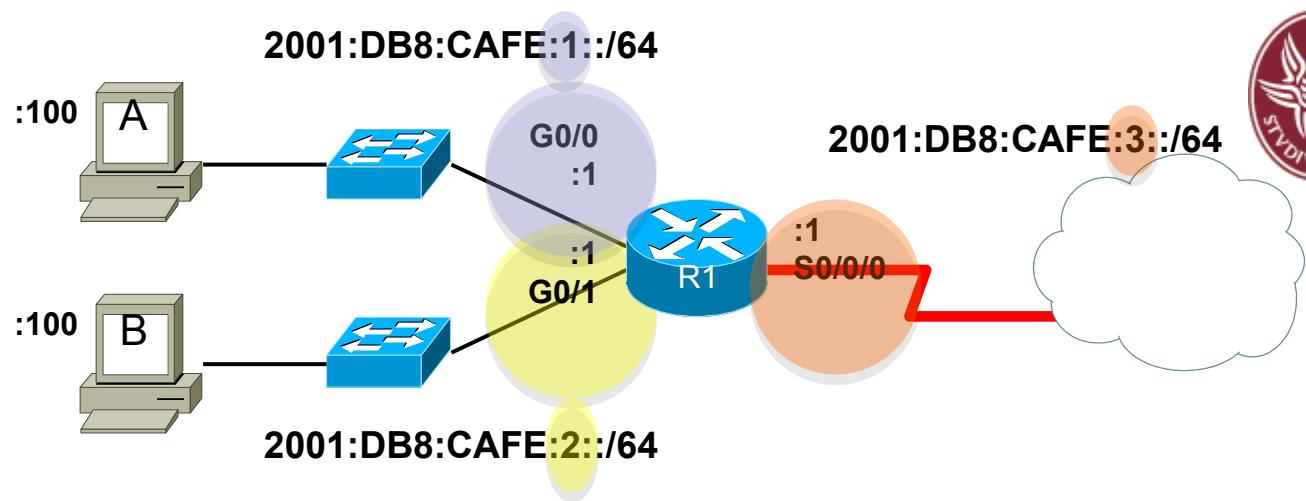
2001:0DB8:CAFE:000A::/64

Valid abbreviation is to remove the leading 0s:

2001:DB8:CAFE:1::/64

3-1-4 Rule

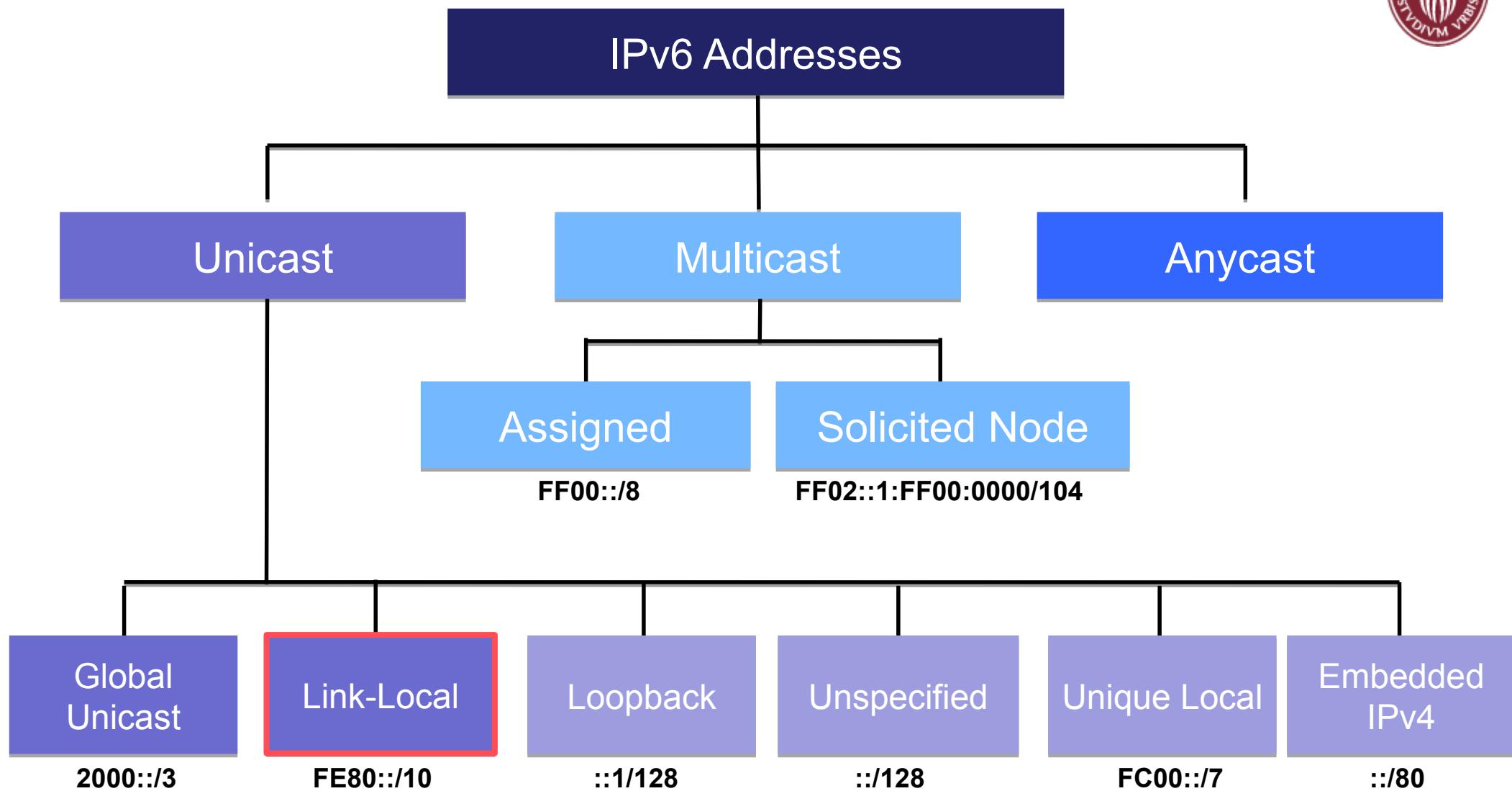
Static GUA Configuration



```
R1(config)#interface gigabitethernet 0/0
R1(config-if)#ipv6 address 2001:db8:cafe:1::1/64
R1(config-if)#no shutdown
R1(config-if)#exit
R1(config)#interface gigabitethernet 0/1
R1(config-if)#ipv6 address 2001:db8:cafe:2::1/64
R1(config-if)#no shutdown
R1(config-if)#exit
R1(config)#interface serial 0/0/0
R1(config-if)#ipv6 address 2001:db8:cafe:3::1/64
R1(config-if)#no shutdown
R1(config-if)#exit
```

Link-local Unicast

IPv6 Address Types

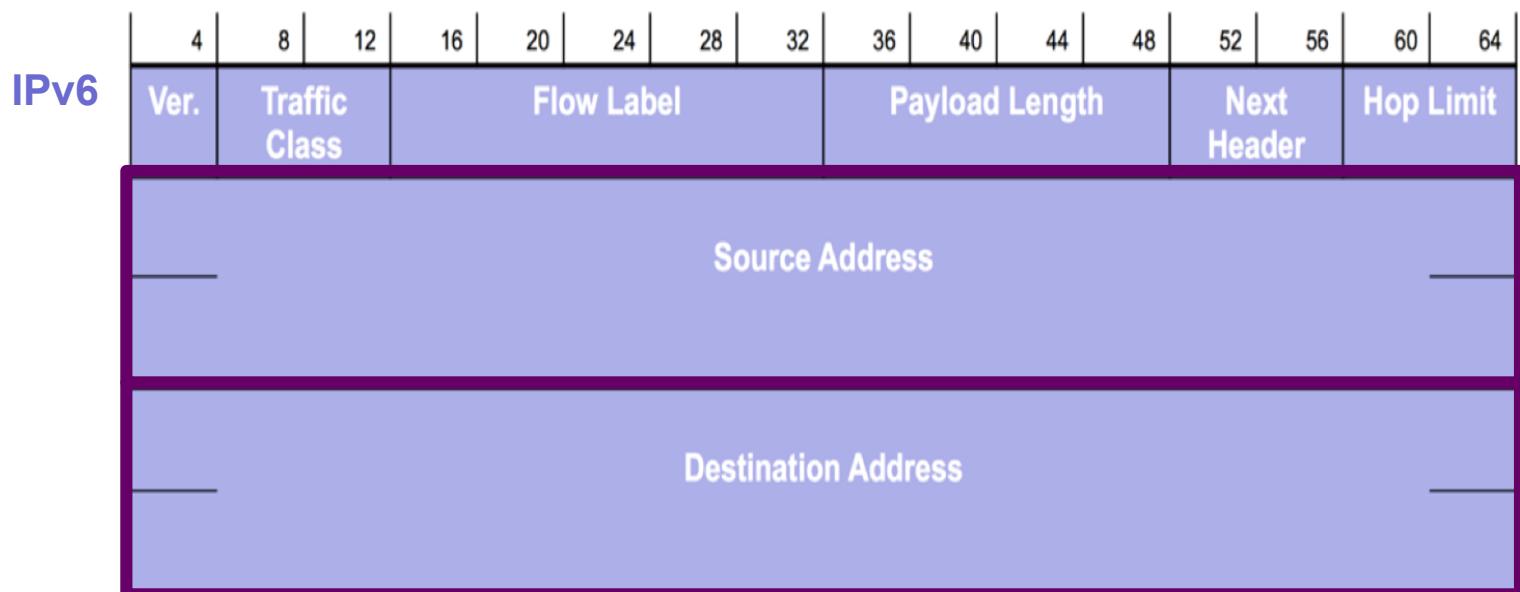
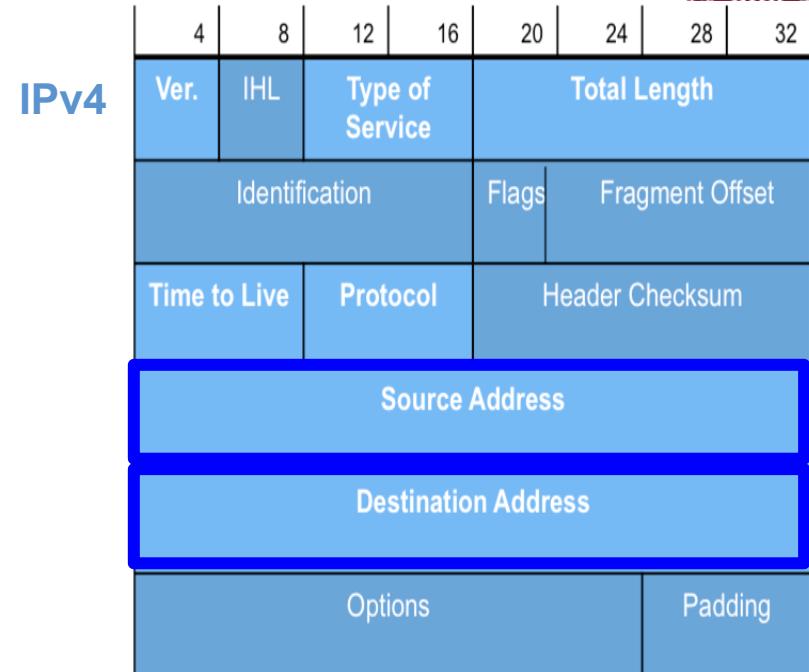


IPv6 does not have a “broadcast” address.

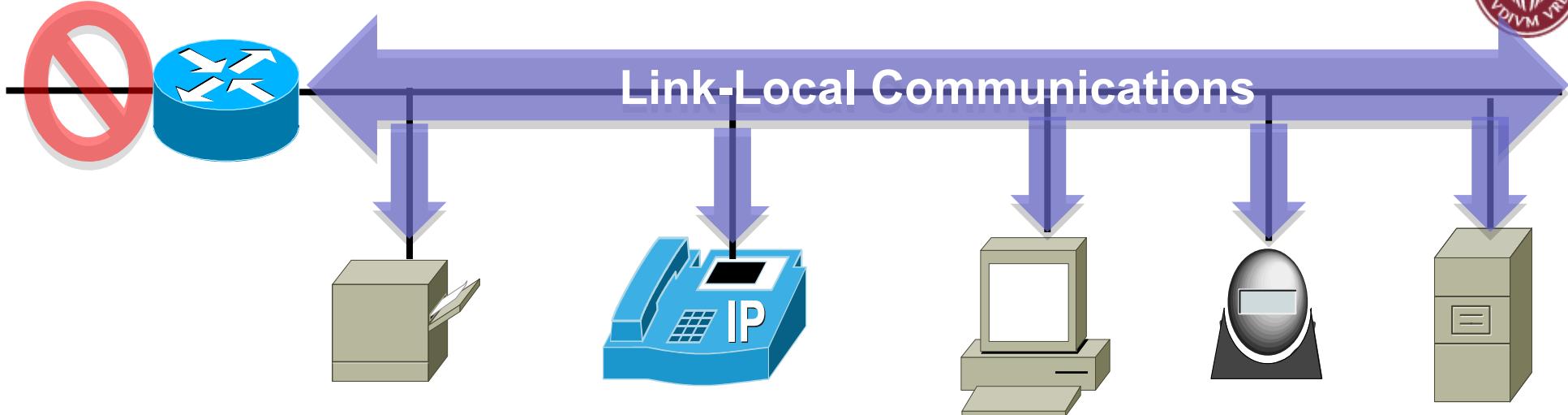
Link-Local Unicast Address



- **IPv6 Source** – Always a unicast
- **IPv6 Destination** – Unicast, multicast, or anycast.
- Unicast, including a *link-local* address



Link-Local Unicast Address



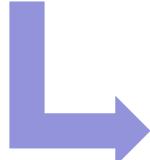
- Used to communicate with other devices on the link.
- Are NOT routable off the link (network).
- ***Only have to be unique on the link.***
- Not included in the IPv6 routing table.
- ***An IPv6 device must have at least a link-local address.***



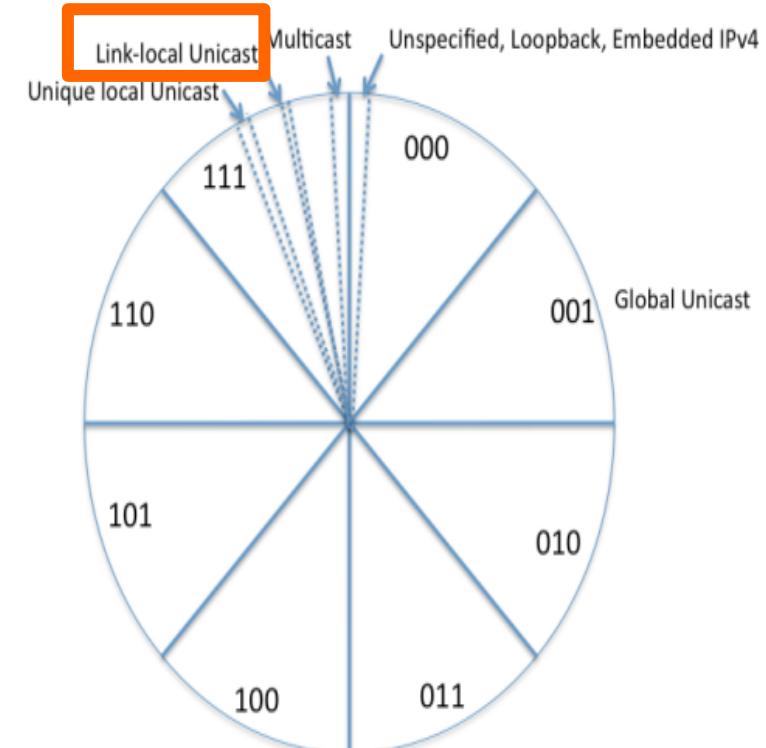
Link-Local Unicast Range

First 10 bits

1111 1110 10xx xxxx	Remaining 54 bits	64-bit Interface ID
---------------------	-------------------	---------------------



Range: FE80: 1111 1110 1000 0000 :
FEBF: 1111 1110 1011 1111 : **First hexet**

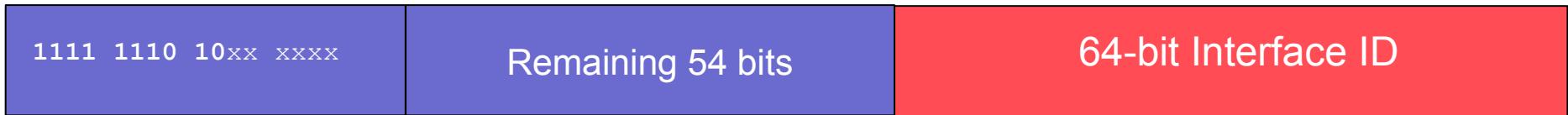


The remaining portion of IPv6 address space are reserved by IETF for future use.

Link-Local Unicast Address



First 10 bits



FE80::Interface ID

Link-local addresses are created

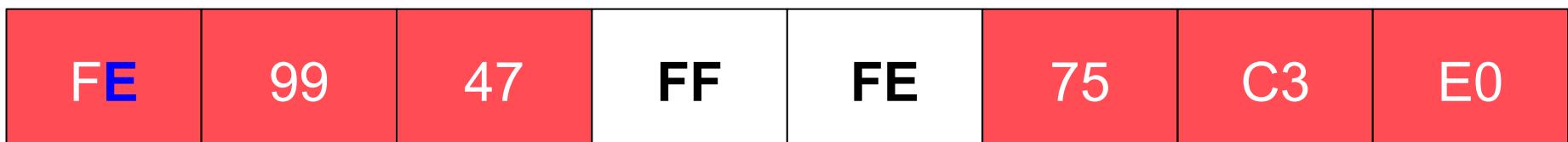
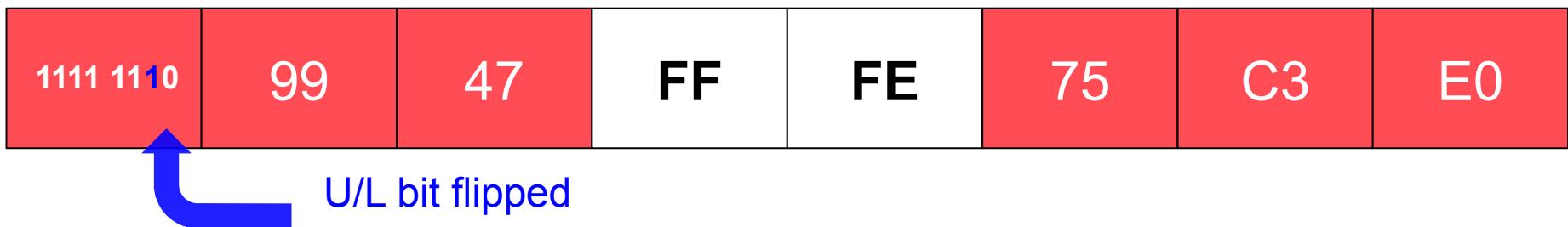
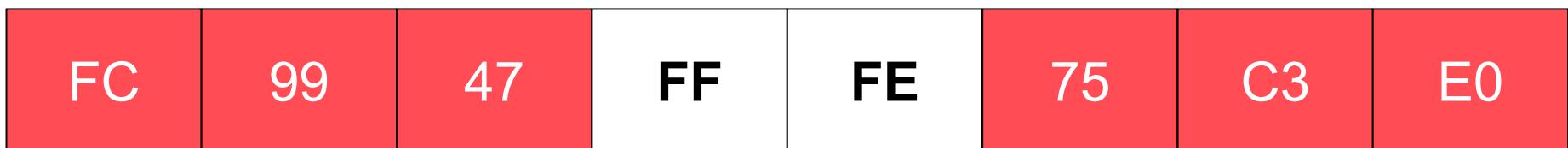
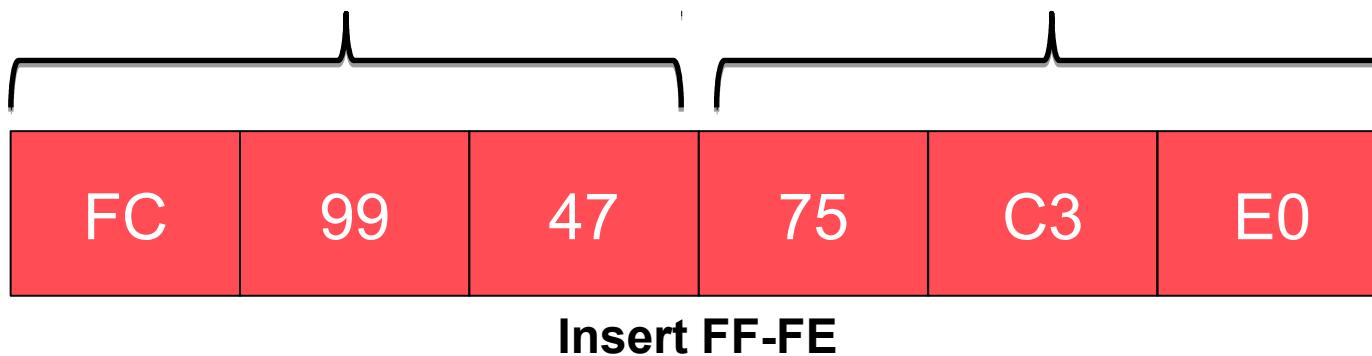
- **Automatically :**
 - **FE80** (usually) – First 10 bits
 - **Interface ID**
 - EUI-64 (Cisco routers)
 - Random 64 bits (many host operating systems)
- Static (manual) configuration – Common practice for routers.

Modified EUI-64 Format (Extended Unique Identifier-64)



OUI (24 bits)

Device Identifier (24 bits)



Verifying the PC's Link-Local Address



First 10 bits

1111 1110 10xx xxxx

Remaining 54 bits

64-bit Interface ID

EUI-64 or random 64-bit value

PC> **ipconfig**

Windows IP Configuration

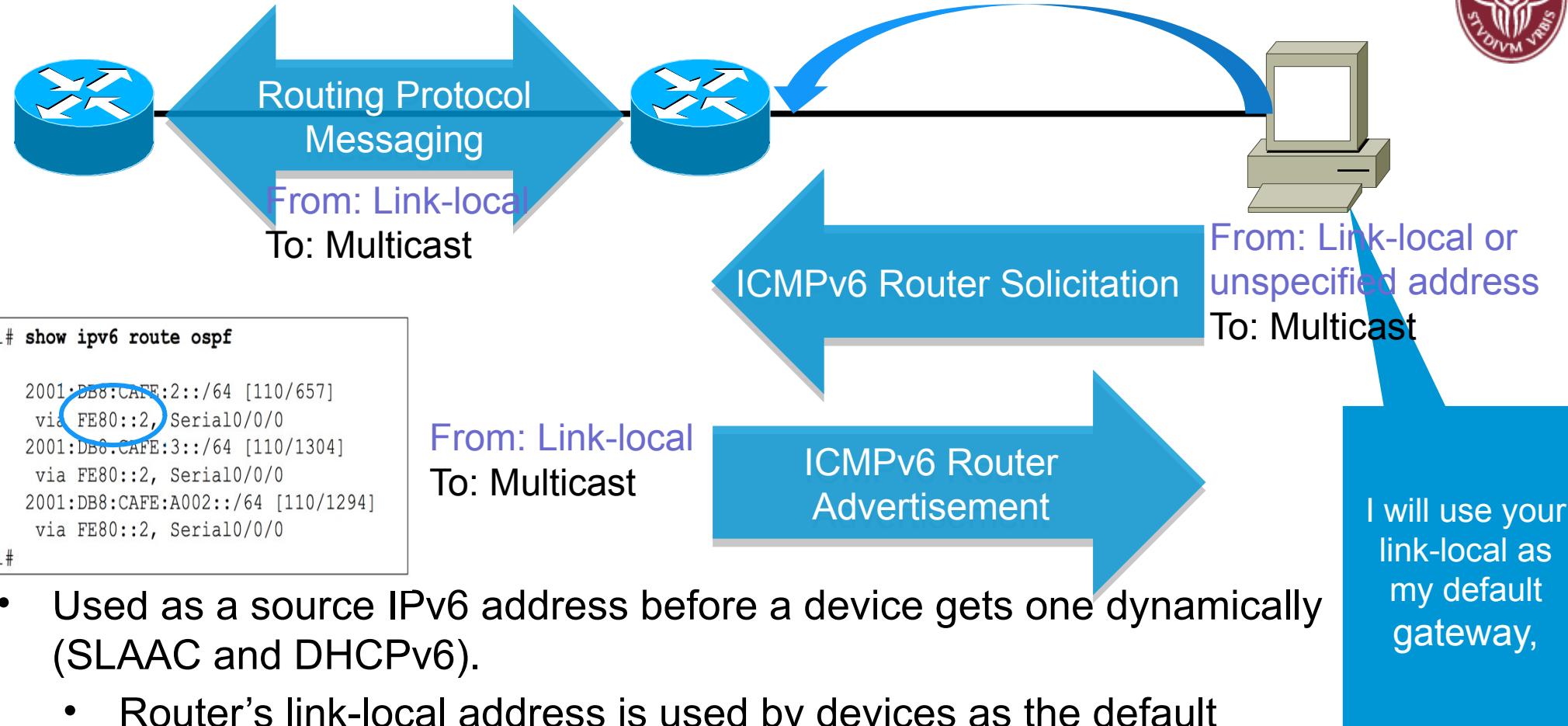
Ethernet adapter Local Area Connection:

Connection-specific DNS Suffix :

Link-local IPv6 Address : fe80::50a5:8a35:a5bb:66e1
IPv4 Address : 192.168.1.101
Subnet Mask : 255.255.255.0
Default Gateway : 192.168.1.1

- Many operating systems will use a random 64-bit Interface IDs for GUA and Link-Local IPv6 Addresses.

An Important Role in IPv6



- Used as a source IPv6 address before a device gets one dynamically (SLAAC and DHCPv6).
 - Router's link-local address is used by devices as the default gateway.
- Routers exchange routing messages.
- Router use the link-local address as the next-hop address in the routing table: **via link-local address**.

SLAAC

Stateless Address Autoconfiguration

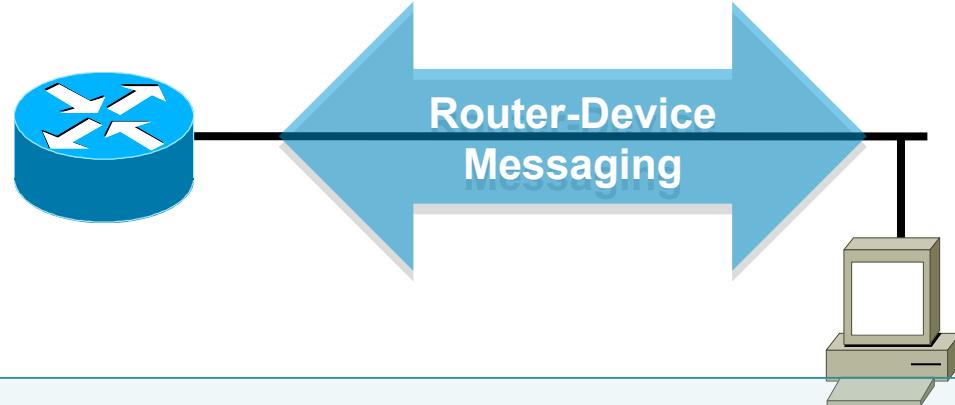
ICMPv6 Neighbor Discover Protocol



ICMPv6 Neighbor Discovery defines 5 different packet types:

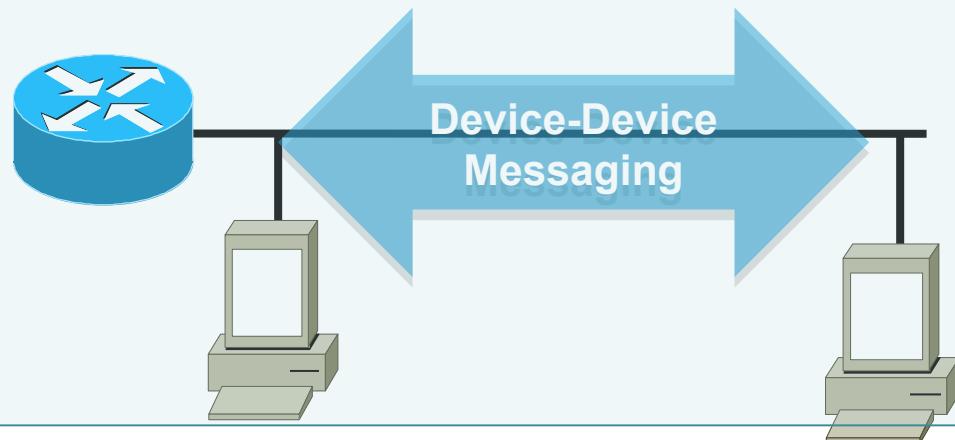
- **Router Solicitation Message**
- **Router Advertisement Message**

Used with dynamic address allocation



- **Neighbor Solicitation Message**
- **Neighbor Advertisement Message**

Used with address resolution (IPv4 ARP)



- **Redirect Message**

Similar to ICMPv4 redirect message

Router-to-Device messaging

See these processes with:
`R1# debug ipv6 nd`

Address Resolution: IPv4 and IPv6



IPv4: ARP over Ethernet

ARP Request: Broadcast



Ethernet

ARP Request/Reply

ARP Cache

Know IPv4,
what is the
MAC?

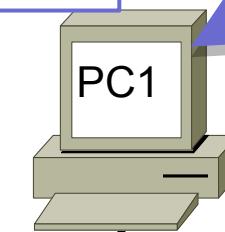
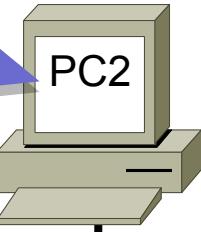
2

1

ARP Reply

ARP Request

My IPv4!
Here is the
MAC?

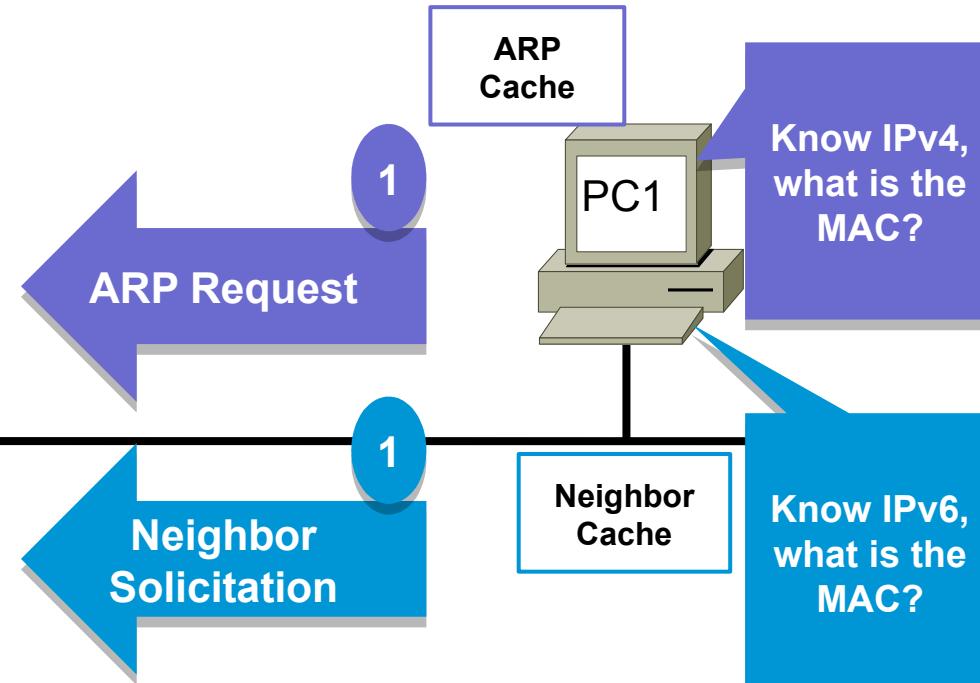


Address Resolution: IPv4 and IPv6



IPv4: ARP over Ethernet

ARP Request: Broadcast



IPv6: ICMPv6 over IPv6 over Ethernet

NS: Multicast

NS: Solicited Node Multicast

Ethernet

IPv6 Header

ICMPv6: Neighbor Solicitation/Advertisement

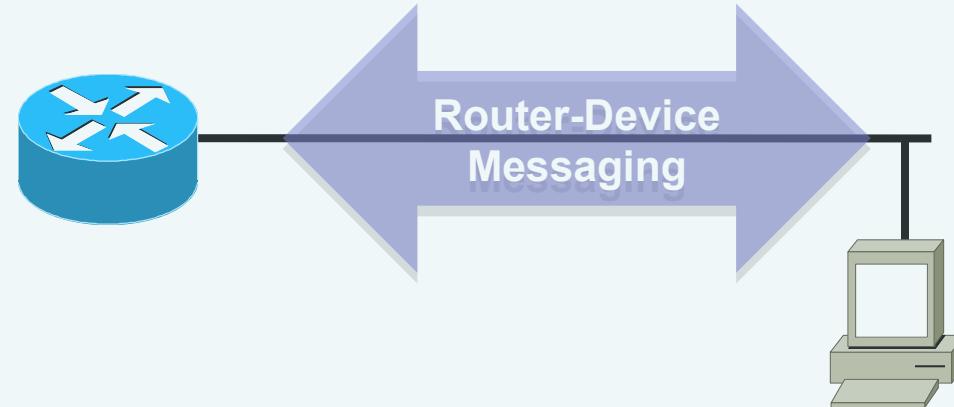
Router Solicitation & Router Advertisement Messages



ICMPv6 Neighbor Discovery defines 5 different packet types:

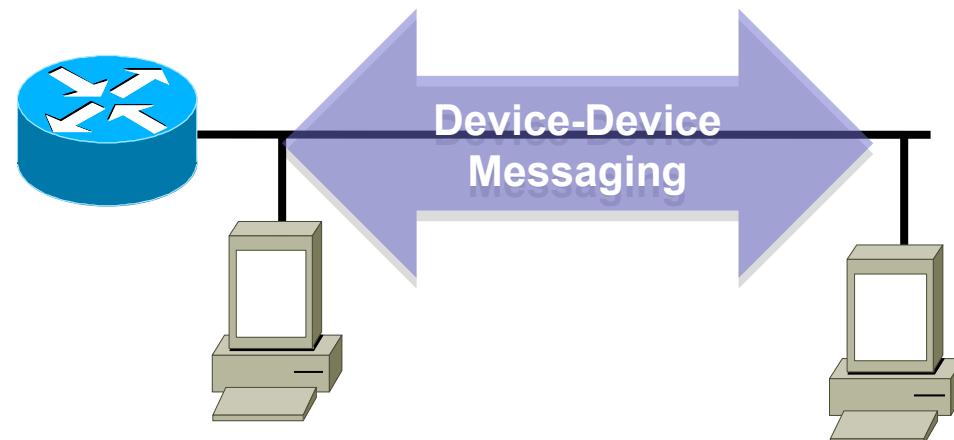
- **Router Solicitation Message**
- **Router Advertisement Message**

Used with dynamic address allocation



- **Neighbor Solicitation Message**
- **Neighbor Advertisement Message**

Used with address resolution (IPv4 ARP)

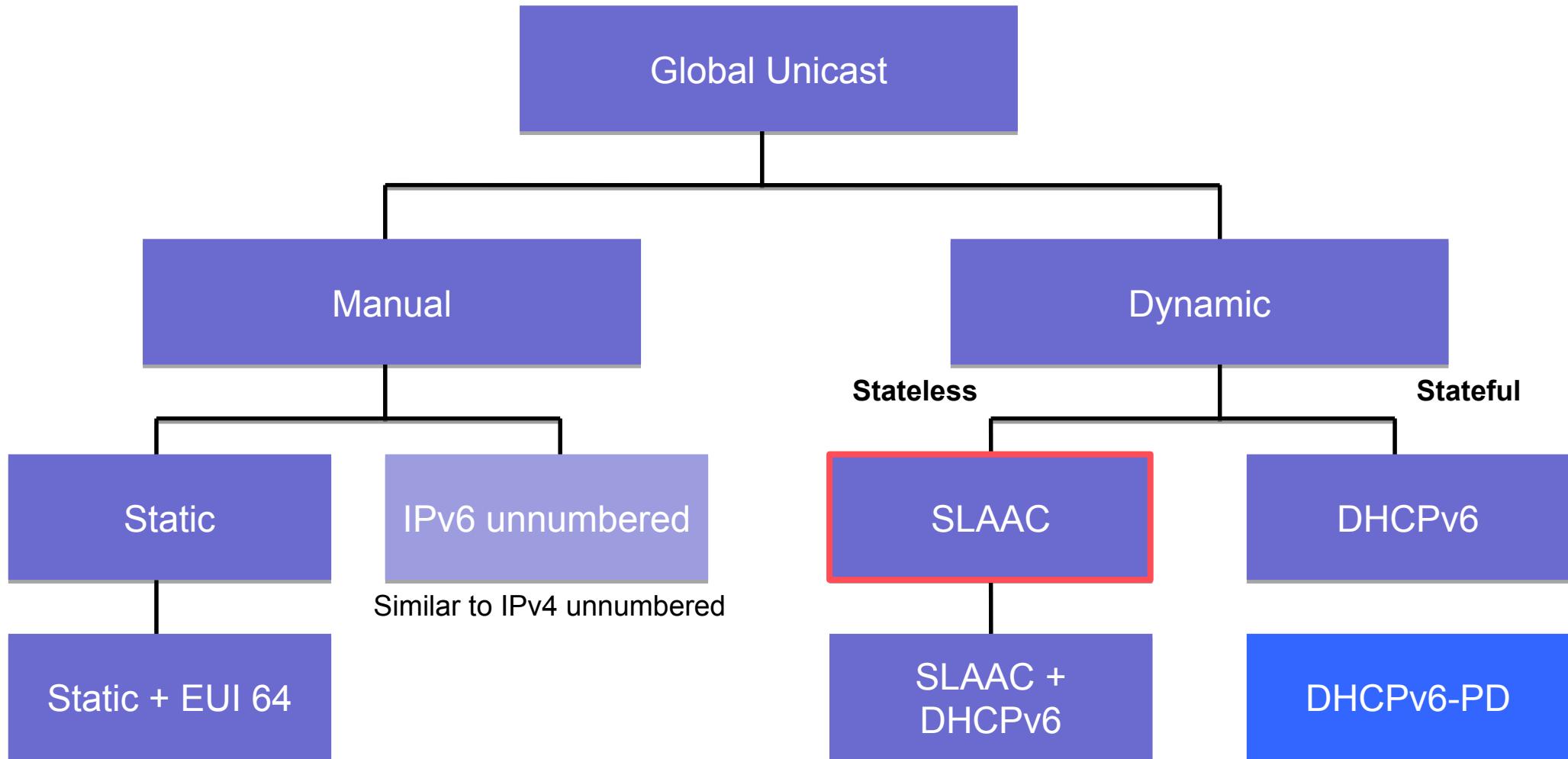


- **Redirect Message**

Similar to ICMPv4 redirect message

Router-to-Device messaging

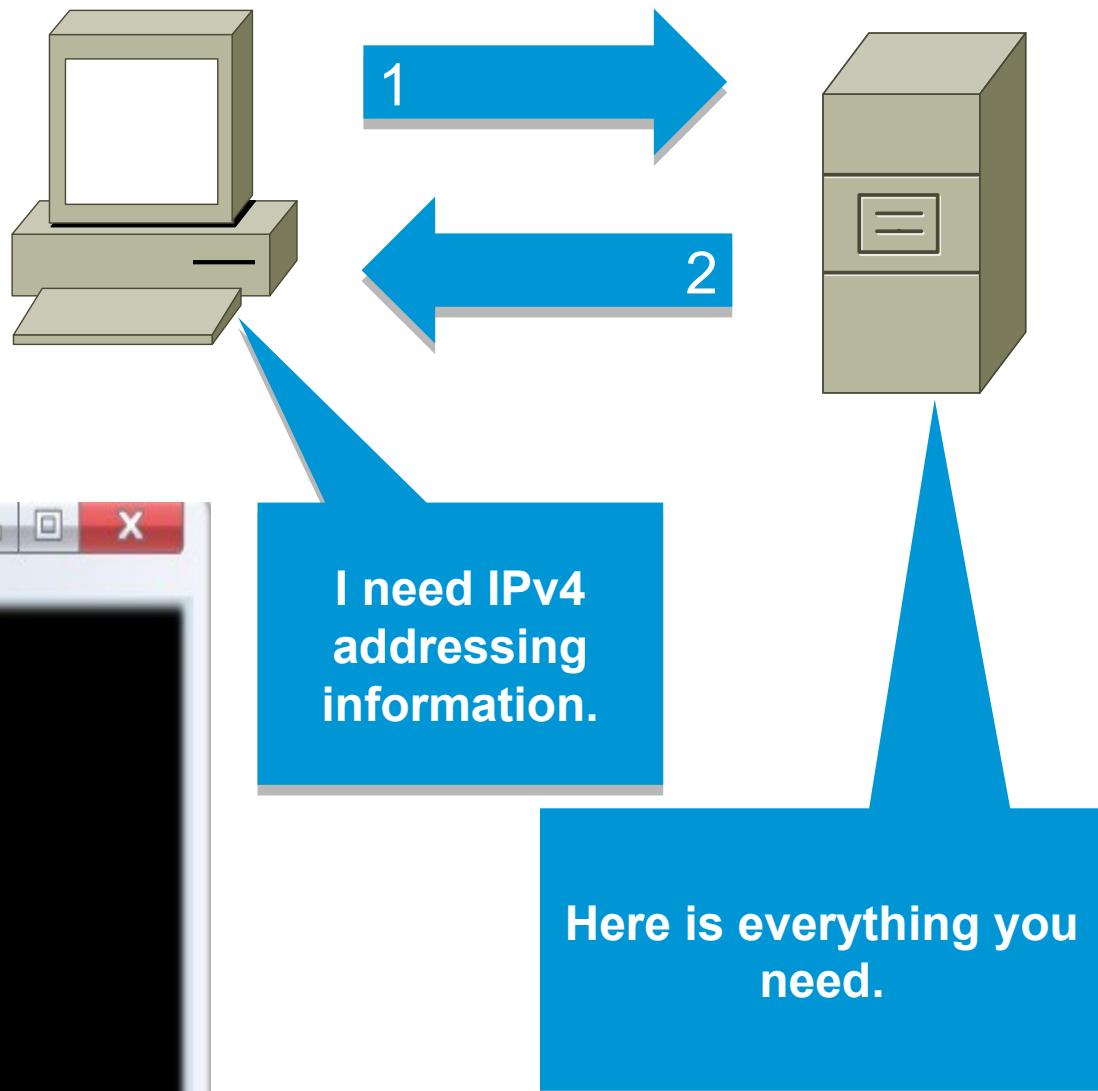
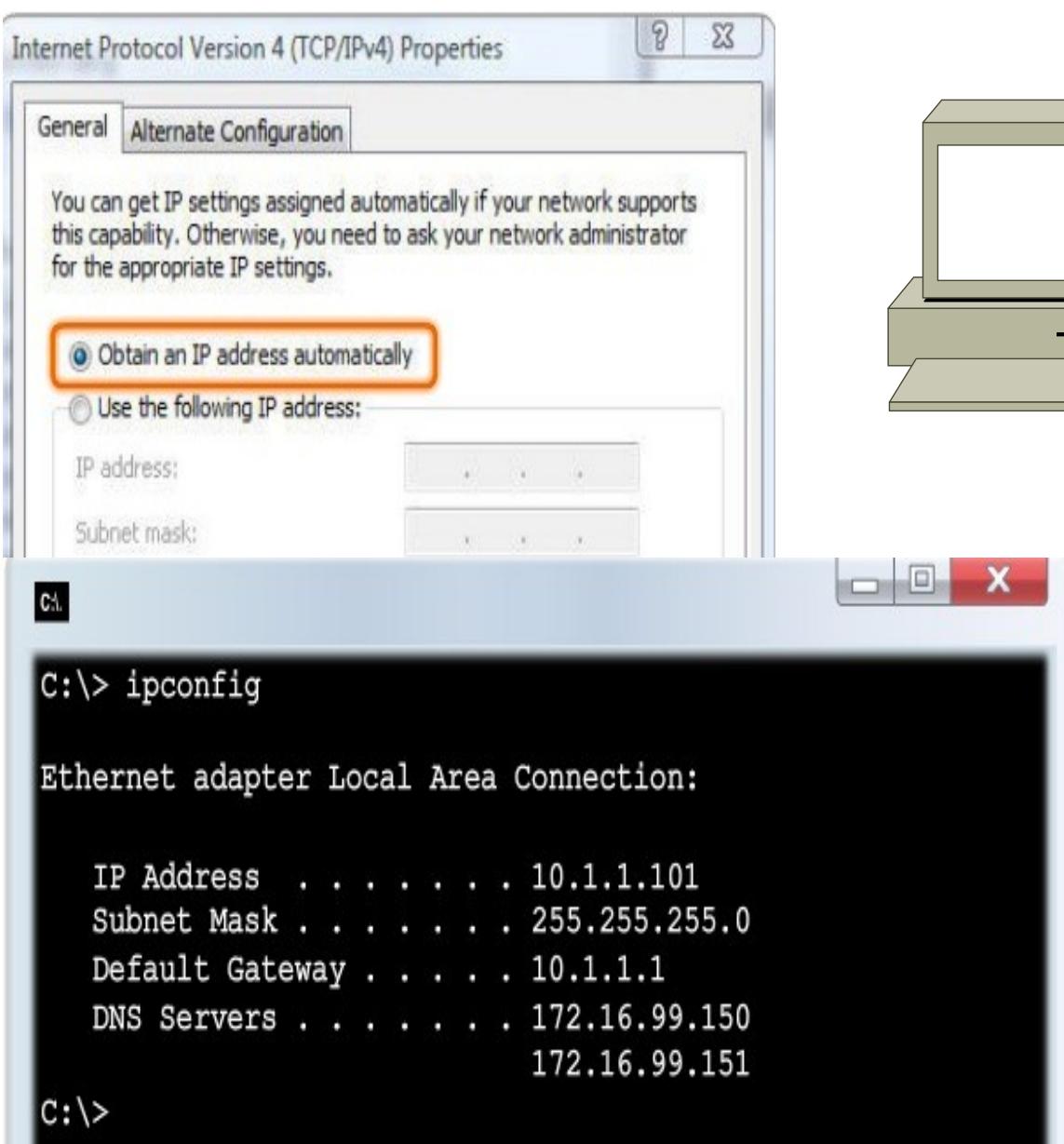
Dynamic IPv6 Address Allocation



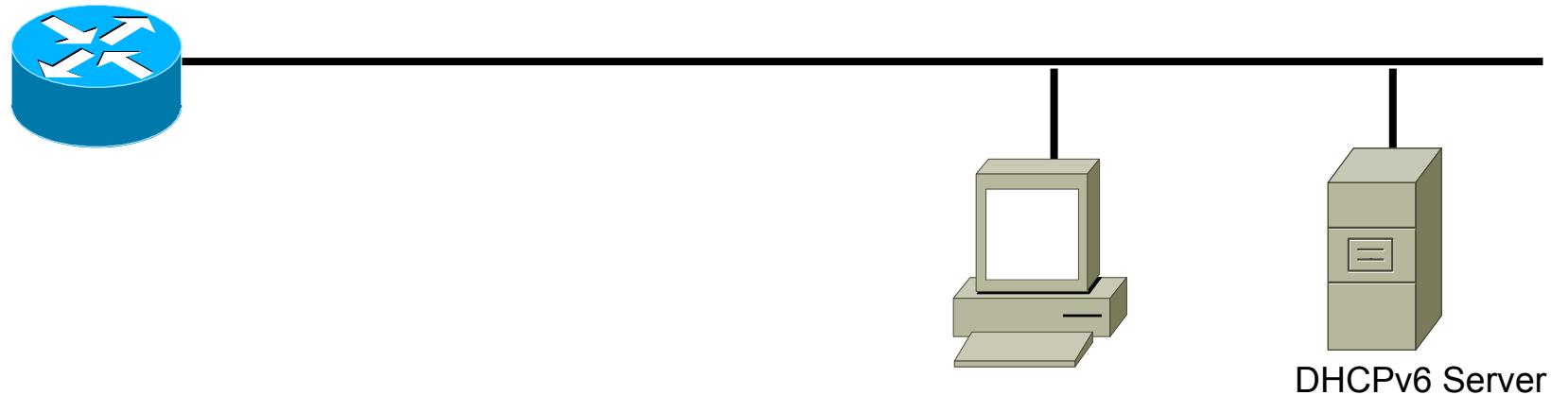
Dynamic Address Allocation in IPv4



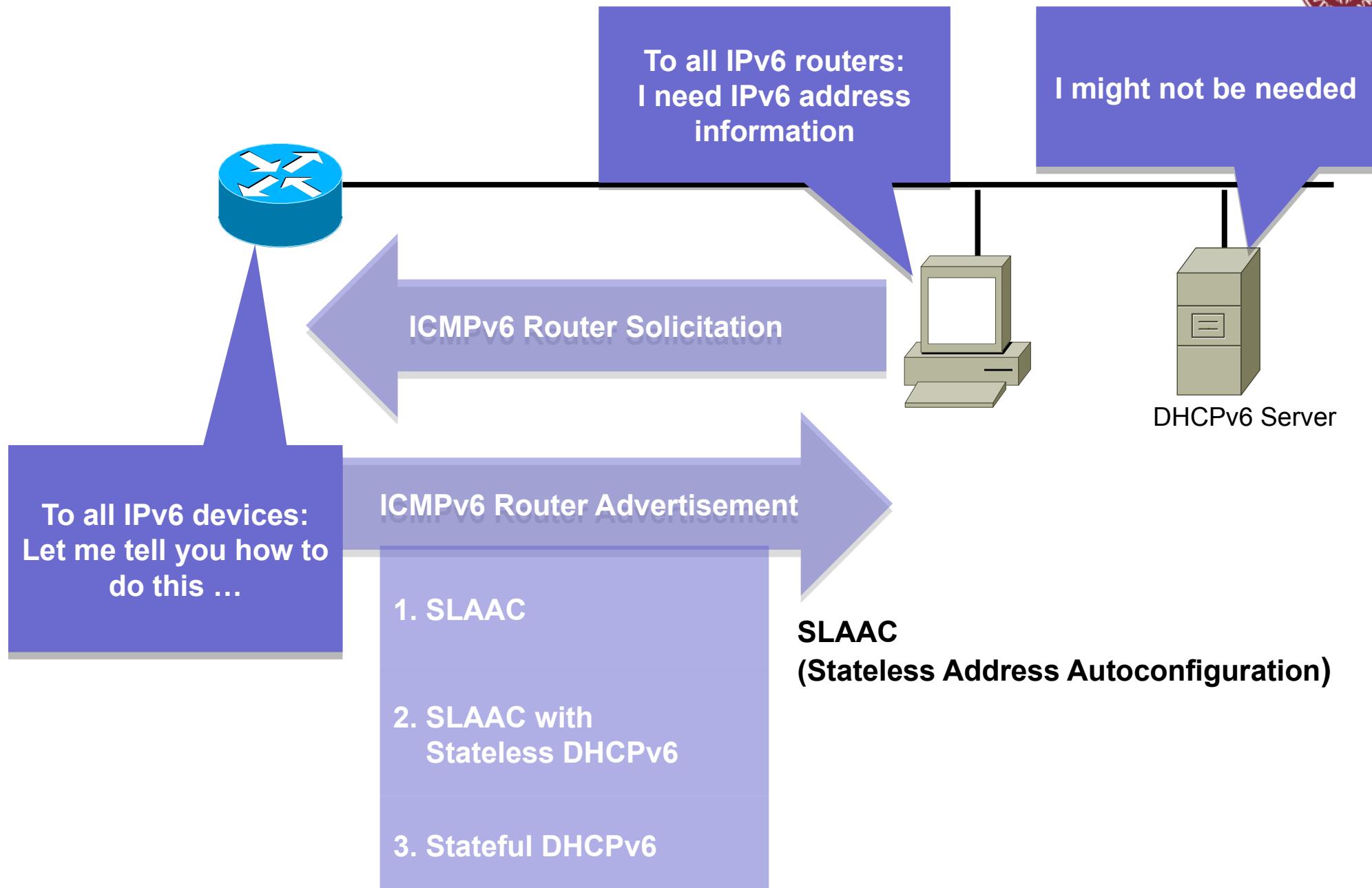
DHCPv4 Server



Dynamic Address Allocation in IPv6



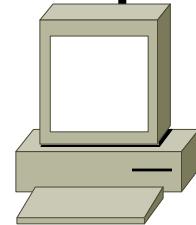
Dynamic Address Allocation in IPv6



Router Advertisement: 3 Options



```
Router(config)# ipv6 unicast-routing
```



DHCPv6 Server

Option 1: SLAAC – No DHCPv6 (Default on Cisco routers)

“I’m *everything* you need (Prefix, Prefix-length, Default Gateway)”

Option 2: SLAAC + Stateless DHCPv6 for DNS address

“Here is my information but you need to get other information such as DNS addresses from a **DHCPv6 server**.” (DNS can be in RA)

Option 3: All addressing except default gateway use DHCPv6

“I can’t help you. Ask a **DHCPv6** server for all your information.”

RA

Option 1 and 2: Stateless Address Autoconfiguration

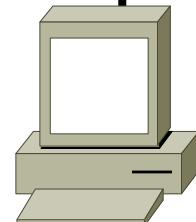
- DHCPv6 Server does not maintain state of addresses

Option 3: Stateful Address Configuration

- Address received from DHCPv6 Server



```
Router(config)# ipv6 unicast-routing
```



Option 1: SLAAC – No DHCPv6 (Default on Cisco routers)

“I’m *everything* you need (Prefix, Prefix-length, Default Gateway)”

Option 2: SLAAC + Stateless DHCPv6 for DNS address

“Here is my information but you need to get other information such as DNS addresses from a **DHCPv6 server**.” (DNS can be in RA)

Option 3: All addressing except default gateway use DHCPv6

“I can’t help you. Ask a **DHCPv6** server for all your information.”

DHCPv6 Server

RA

Obtaining an IPv6 Address Automatically



Internet Protocol Version 6 (TCP/IPv6) Properties

General

You can get IPv6 settings assigned automatically if your network supports this capability. Otherwise, you need to ask your network administrator for the appropriate IPv6 settings.

Obtain an IPv6 address automatically

Use the following IPv6 address:

IPv6 address:

Subnet prefix length:

Default gateway:

Obtain DNS server address automatically

Use the following DNS server addresses:

Preferred DNS server:

Alternate DNS server:

Validate settings upon exit

Advanced...

OK Cancel

Network

Ethernet Location: Automatic

TCP/IP DNS WINS 802.1X Proxies Hardware

Configure IPv4: Using DHCP

IPv4 Address: Renew DHCP Lease

Subnet Mask: Configure IP DHCP Client ID: (If required)

Router:

Configure IPv6: Automatically

IPv6 Address: Router: DNS Server: Search Domains:

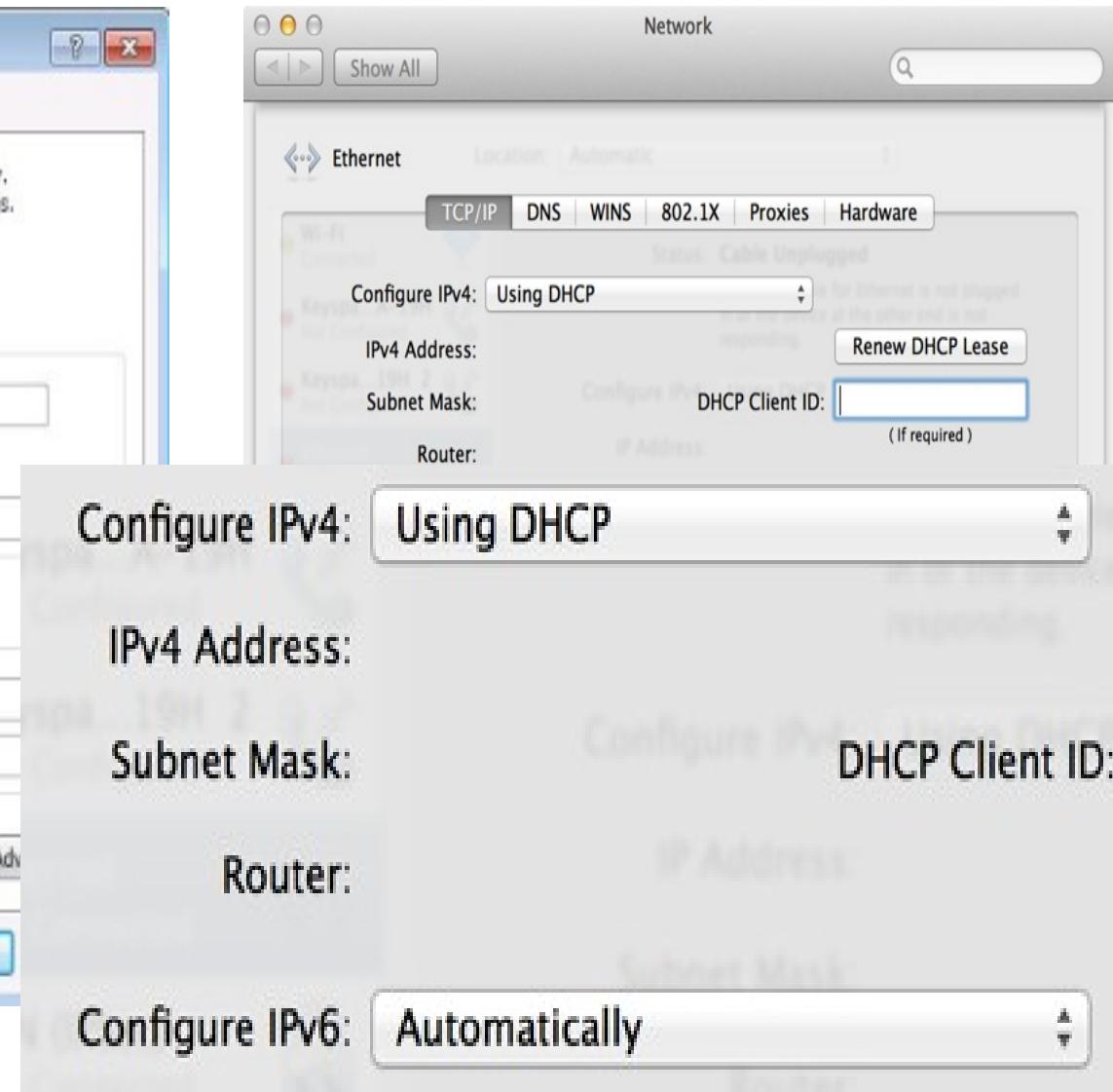
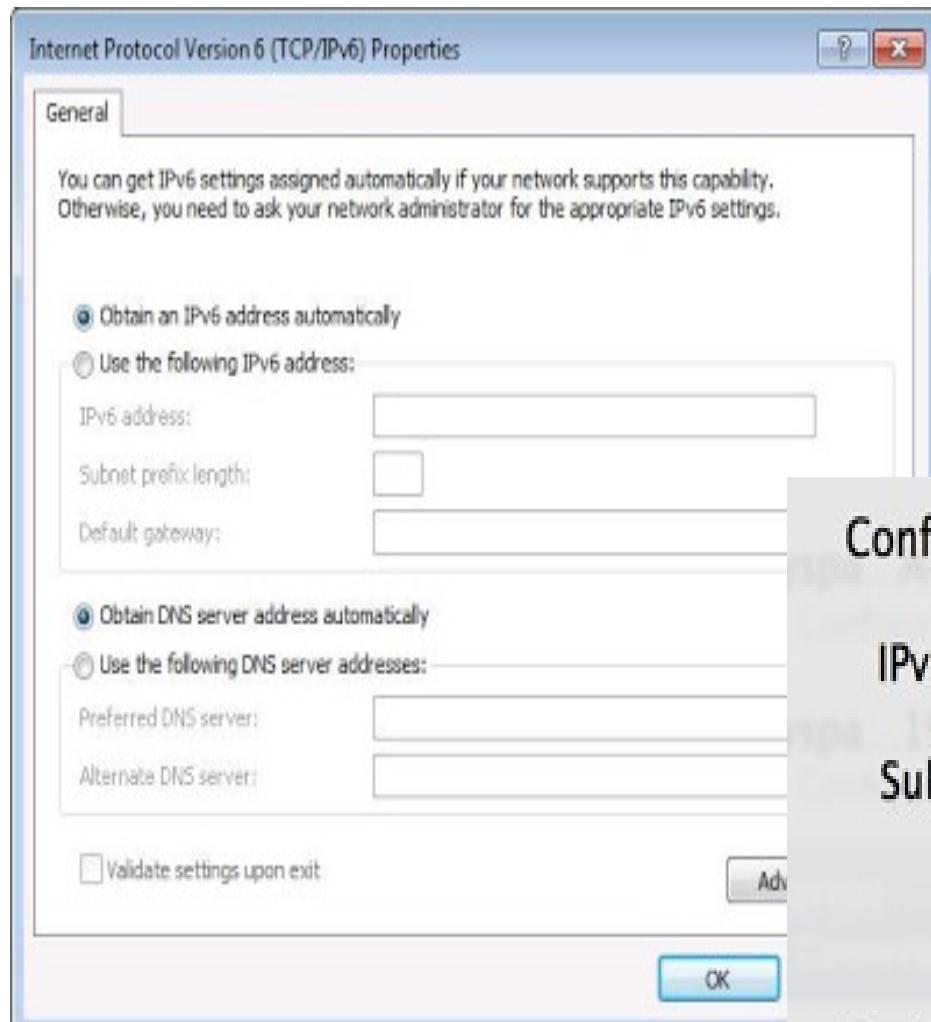
Prefix Length: 802.1X: WPA secure- Connect

Advanced...

Cancel OK

Click the lock to prevent further changes Assist me... Help

Obtaining an IPv6 Address Automatically

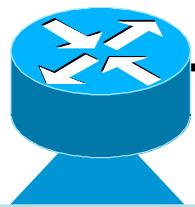


SLAAC: Stateless Address Autoconfiguration



2001:DB8:CAFE:1::/64

MAC: 00-19-D2-8C-E0-4C



SLAAC Option 1 – RA Message

To: FF02::1 (All-IPv6 devices)

From: FE80::1 (Link-local address)

Prefix: 2001:DB8:CAFE:1::

Prefix-length: /64

1

RA

2

Prefix: 2001:DB8:CAFE:1::

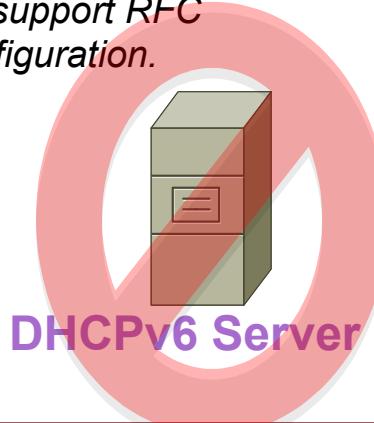
Prefix-length: /64

Default Gateway: FE80::1

Global Unicast Address:

2001:DB8:CAFE:1: + Interface ID

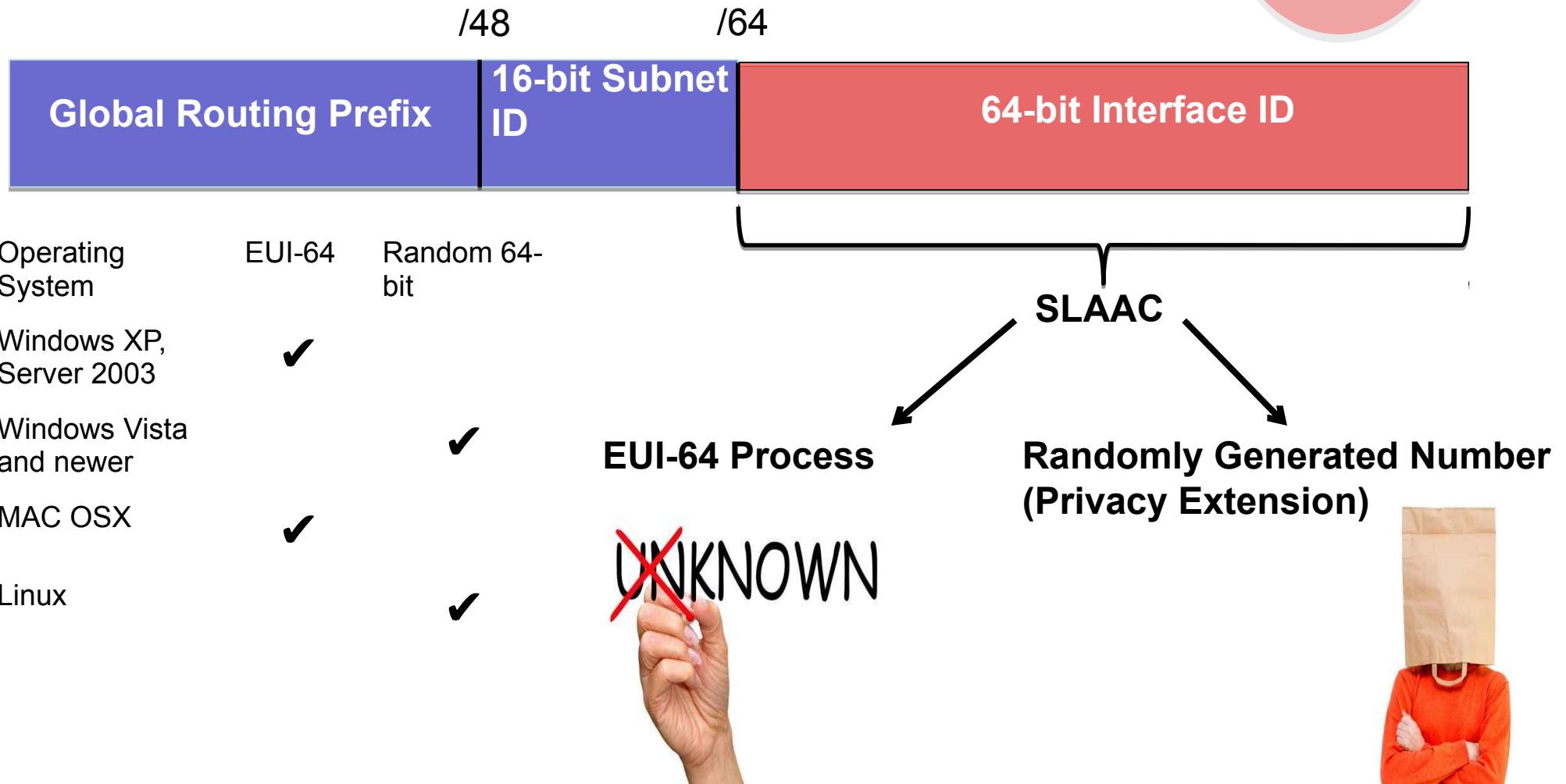
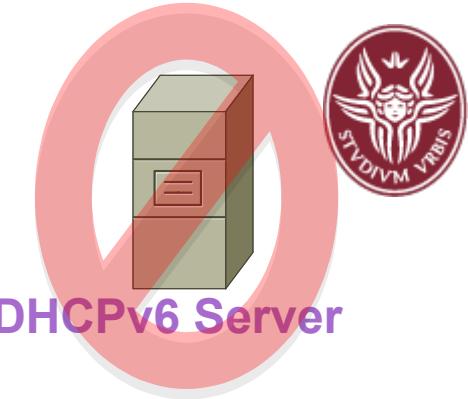
Note: Domain name and DNS server list may be included if router (and end system) support RFC 6106 IPv6 RA Options for DNS Configuration.



3

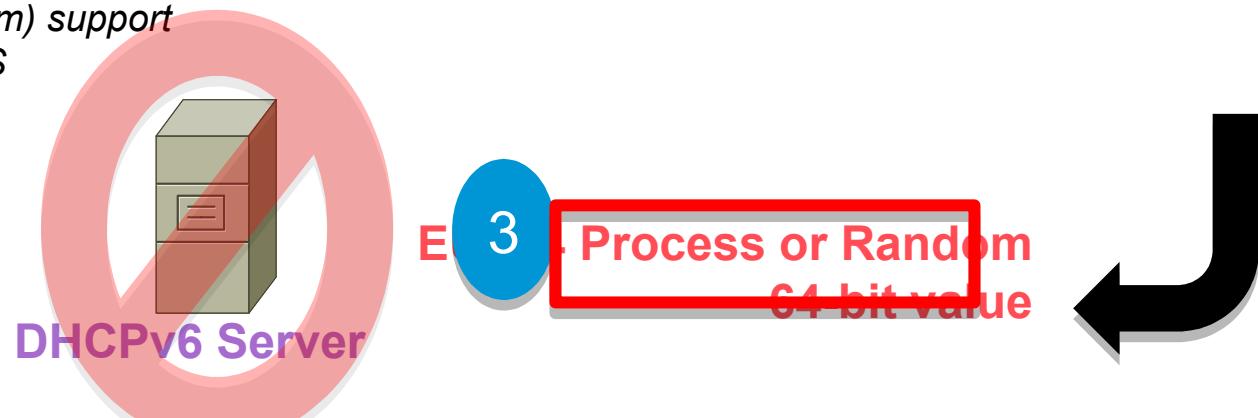
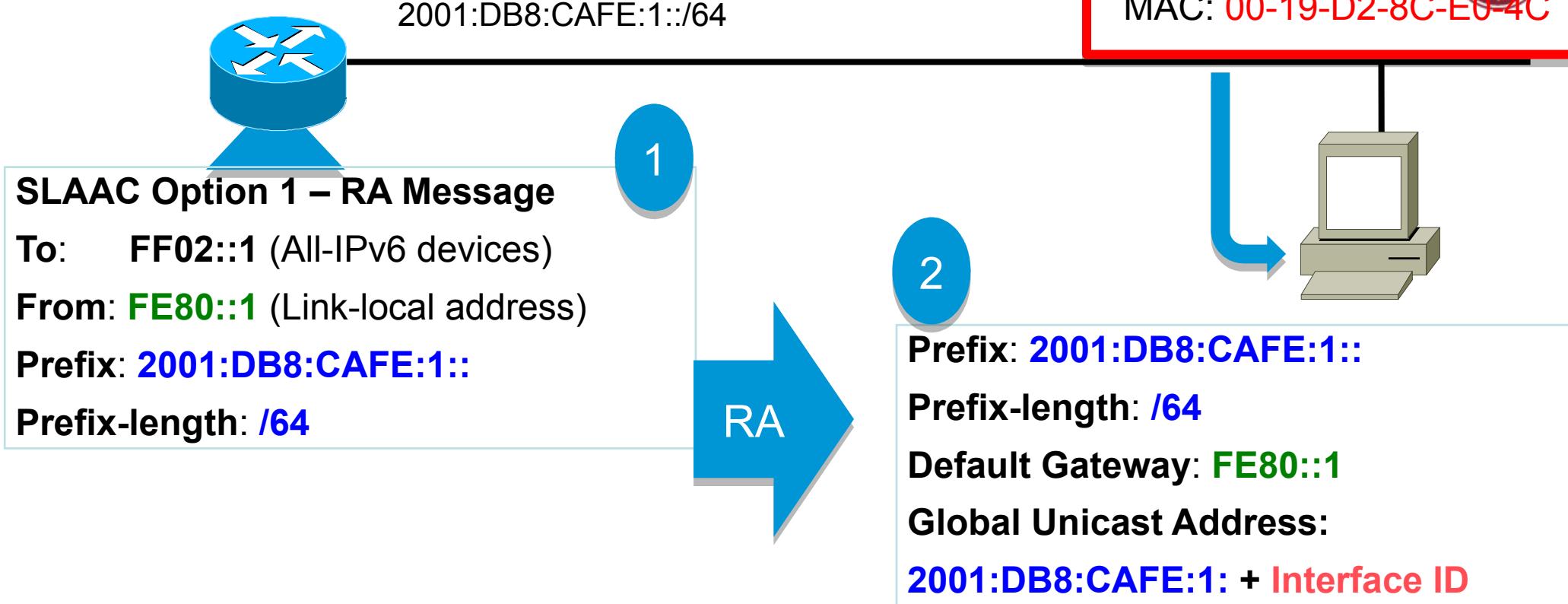
EUI-64 Process or Random
64-bit value

SLAAC: Interface ID



Default OS behavior can be changed.

SLAAC: EUI-64 Option

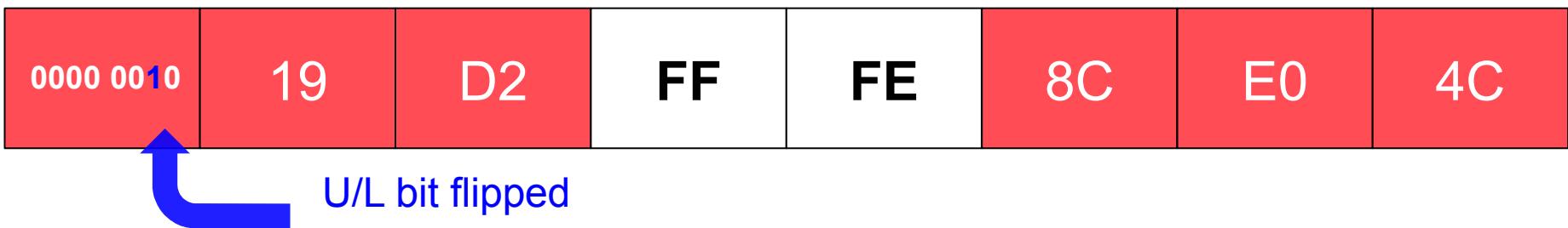
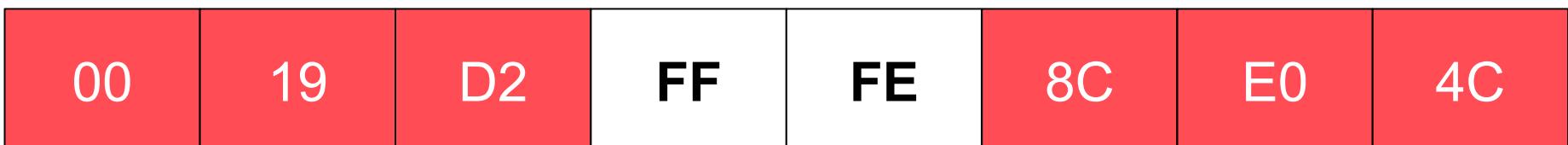
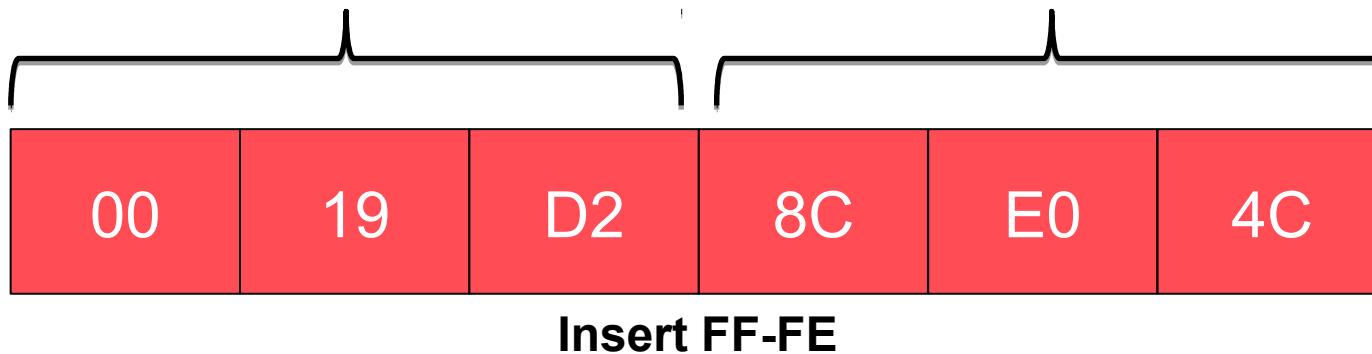


Modified EUI-64 Format (Extended Unique Identifier-64)

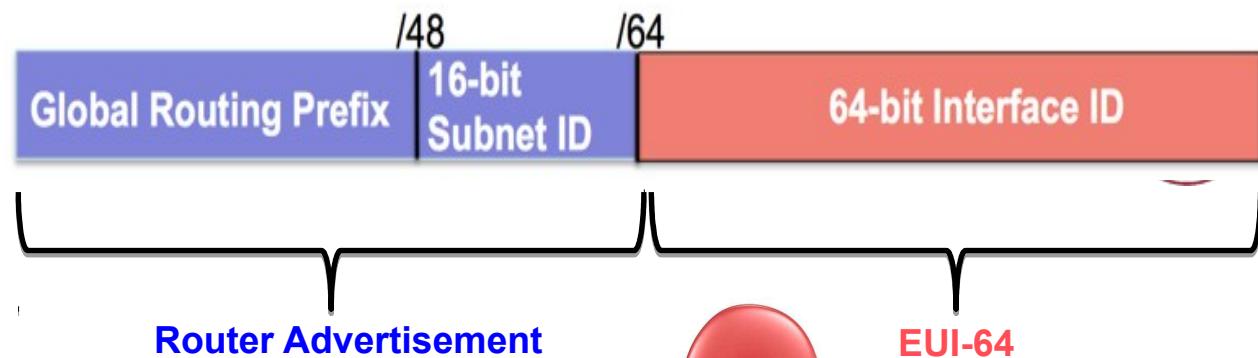


OUI (24 bits)

Device Identifier (24 bits)



Verifying SLAAC on the PC Using EUI-64



```
PC> ipconfig
```

Windows IP Configuration

Ethernet adapter Local Area Connection:

 IPv6 Address : **2001:db8:cafe:1:0219:d2ff:fe8c:e04c**

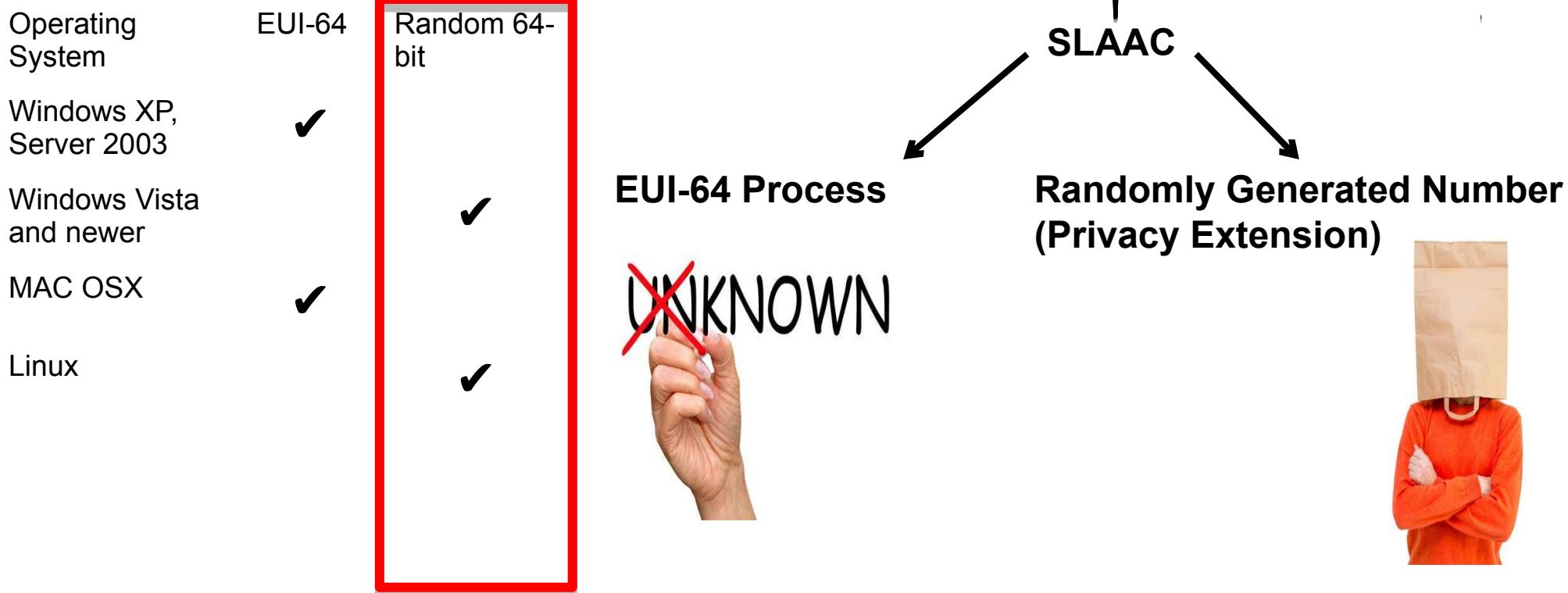
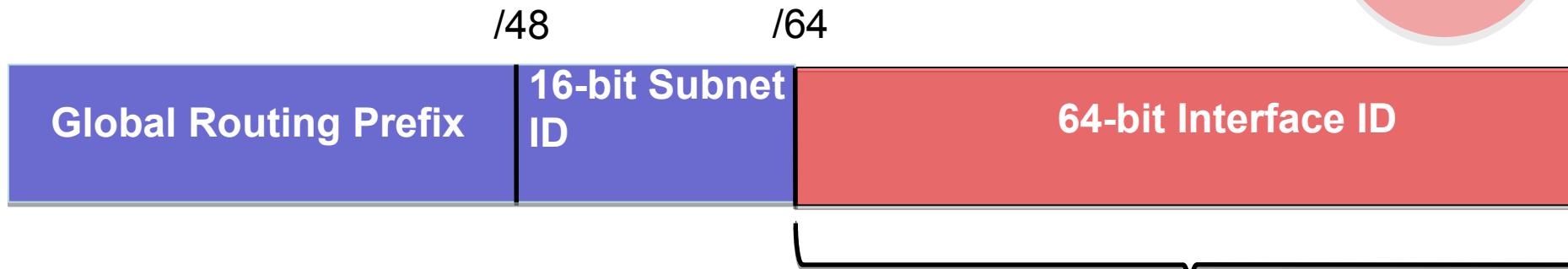
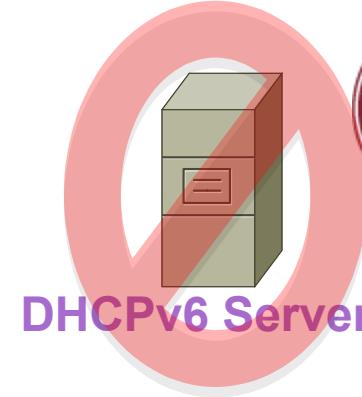
 Link-local IPv6 Address . . . : **fe80::0219:d2ff:fe8c:e04c**

 Default Gateway : **fe80::1**

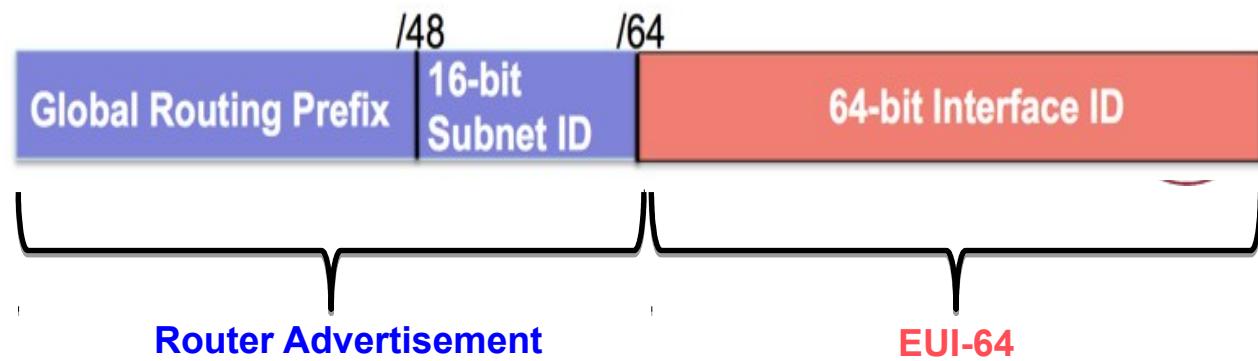
A 64-bit Interface ID and the EUI-64 process accommodates:

- The IEEE specification for a 64-bit MAC address
- 64-bit boundary processing

SLAAC: Random 64-bit Interface ID



Verifying SLAAC on the PC Using Privacy Extension



```
PC-Windows7> ipconfig  
Windows IP Configuration  
Ethernet adapter Local Area Connection:  
    IPv6 Address . . . . . : 2001:db8:cafe:1:50a5:8a35:a5bb:66e1  
    Link-local IPv6 Address . . . : fe80::50a5:8a35:a5bb:66e1  
    Default Gateway . . . . . : fe80::1
```

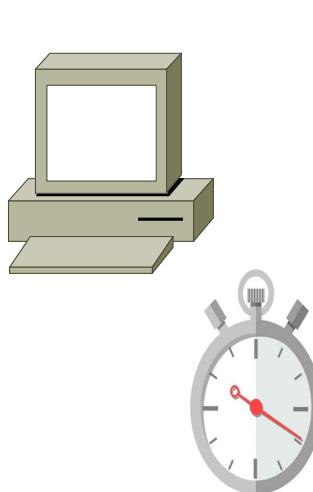
No FF-FE

Ensuring Unique Unicast Addresses



Global Unicast - `2001:db8:cafe:1:0219:d2ff:fe8c:e04c`

Link-local - `fe80::50a5:8a35:a5bb:66e1`



Neighbor Solicitation

Neighbor Advertisement?

Not received = unique address
Received = duplicate address

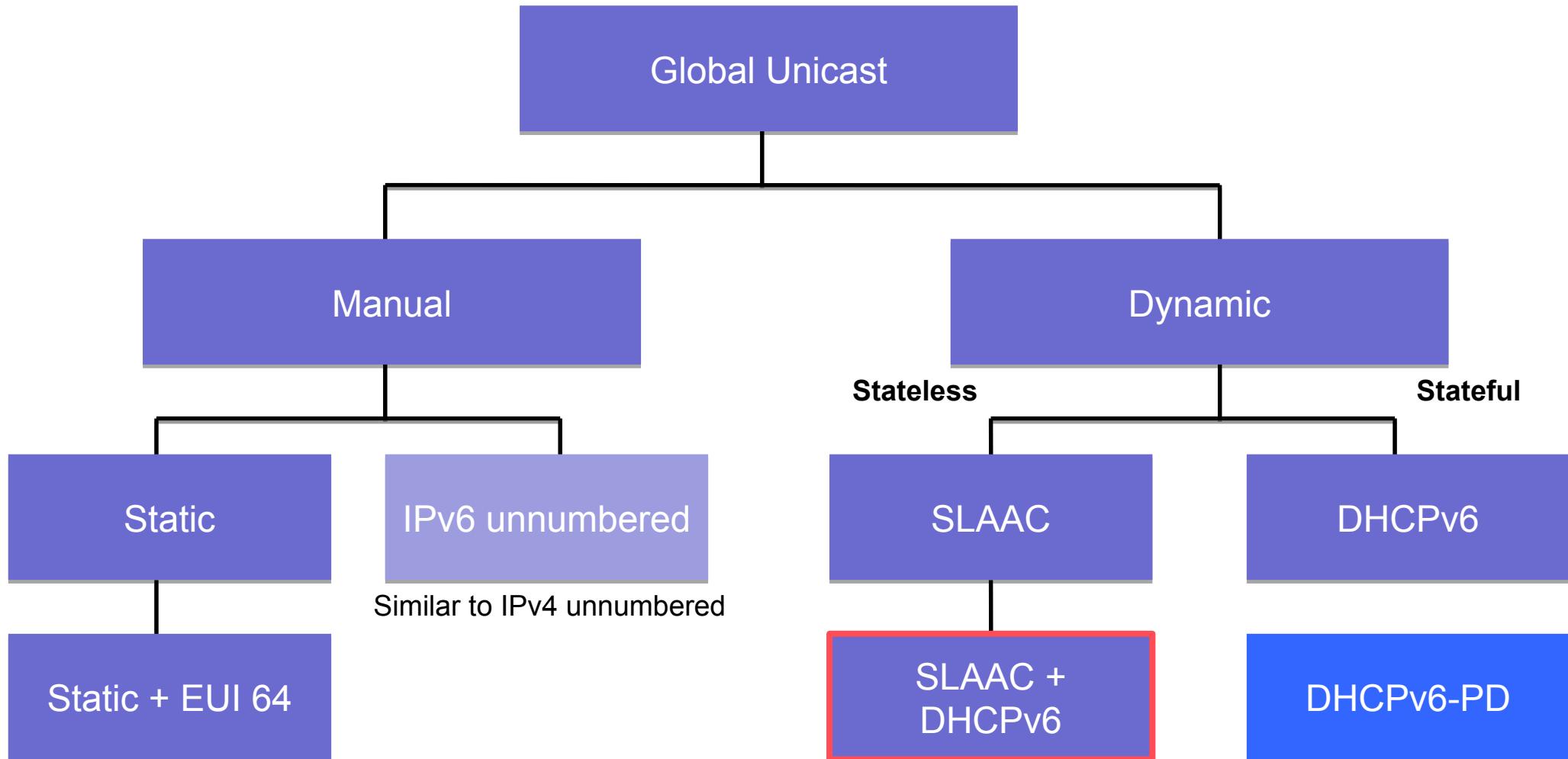
- SLAAC is **stateless**, no entity (DHCPv6 server) maintaining a state address-to-device mappings.
- **How can we guarantee the address is unique?**
- **Duplicate Address Detection (DAD)**
 - Once required for all unicast addresses (static or dynamic), RFC was updated that DAD is only recommended.
 - /64 Interface IDs!



SAPIENZA
UNIVERSITÀ DI ROMA

DHCPv6 (Stateless vs Stateful)

DHCPv6

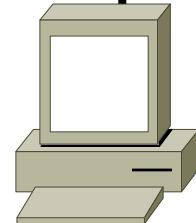


RA Message

- Option 1 and 2: Stateless Address Autoconfiguration
- DHCPv6 Server does not maintain state of addresses
- Option 3: Stateful Address Configuration
- Address received from DHCPv6 Server



```
Router(config)# ipv6 unicast-routing
```



DHCPv6 Server

Option 1: SLAAC – No DHCPv6 (Default on Cisco routers)

“I’m *everything* you need (Prefix, Prefix-length, Default Gateway)”

Option 2: SLAAC + Stateless DHCPv6 for DNS address

“Here is my information but you need to get other information such as DNS addresses from a **DHCPv6 server**.” (DNS can be in RA)

Option 3: All addressing except default gateway use DHCPv6

“I can’t help you. Ask a **DHCPv6** server for all your information.”

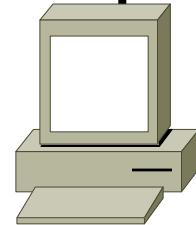
RA

RA Message

- Option 1 and 2: Stateless Address Autoconfiguration
- DHCPv6 Server does not maintain state of addresses
- Option 3: Stateful Address Configuration
- Address received from DHCPv6 Server



```
Router(config)# ipv6 unicast-routing
```



DHCPv6 Server

Option 1: SLAAC – No DHCPv6 (Default on Cisco routers)

"I'm *everything* you need (Prefix, Prefix-length, Default Gateway)"

Option 2: SLAAC + Stateless DHCPv6 for DNS address

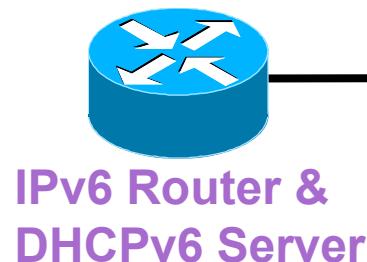
"Here is my information but you need to get other information such as DNS addresses from a **DHCPv6 server**." (DNS can be in RA)

Option 3: All addressing except default gateway use DHCPv6

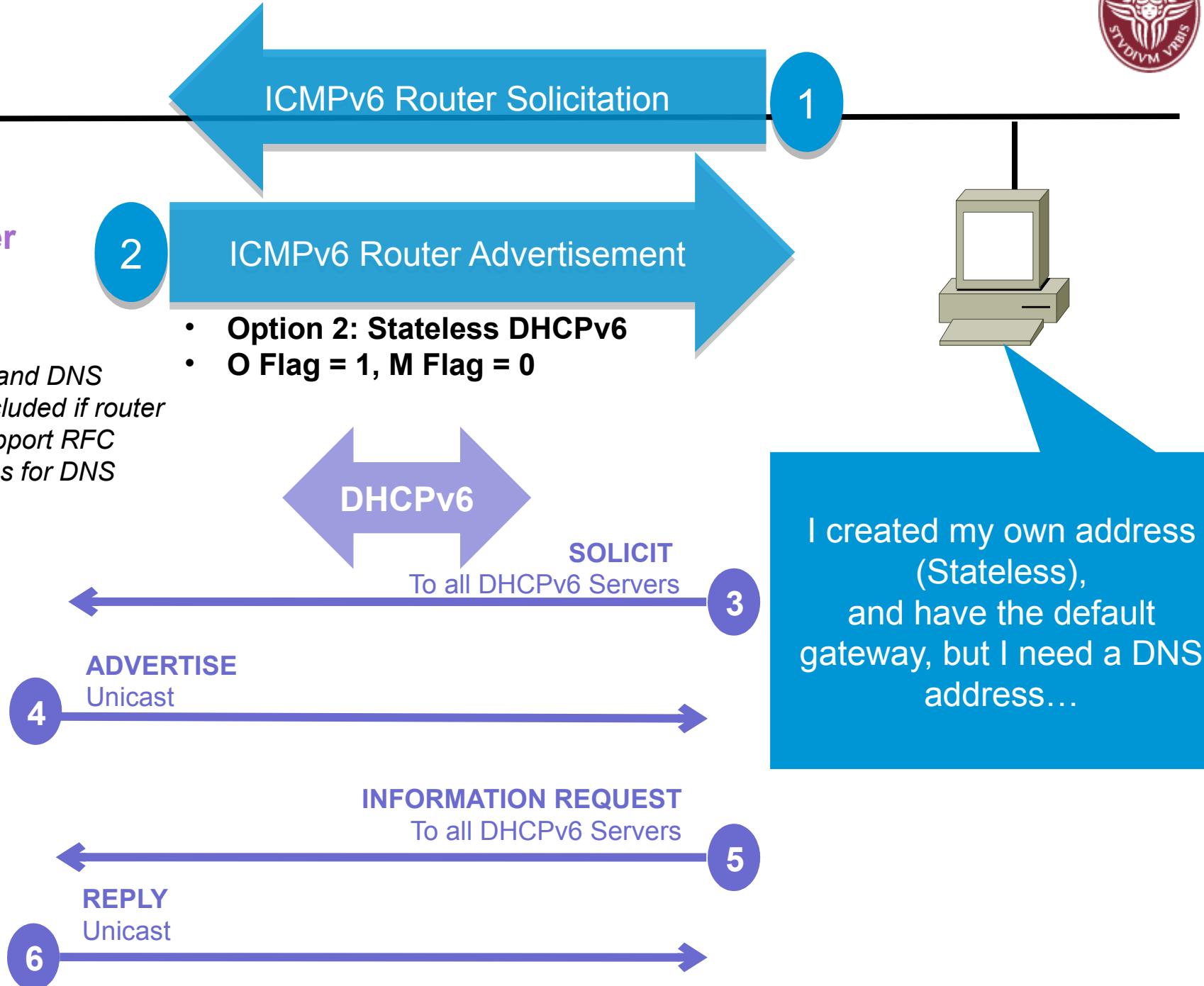
"I can't help you. Ask a **DHCPv6** server for all your information."

RA

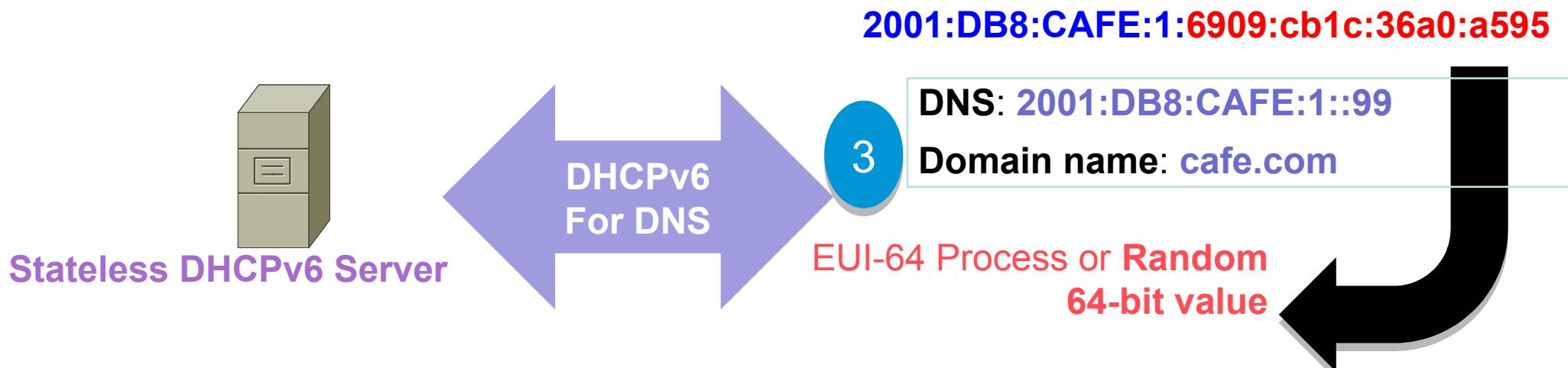
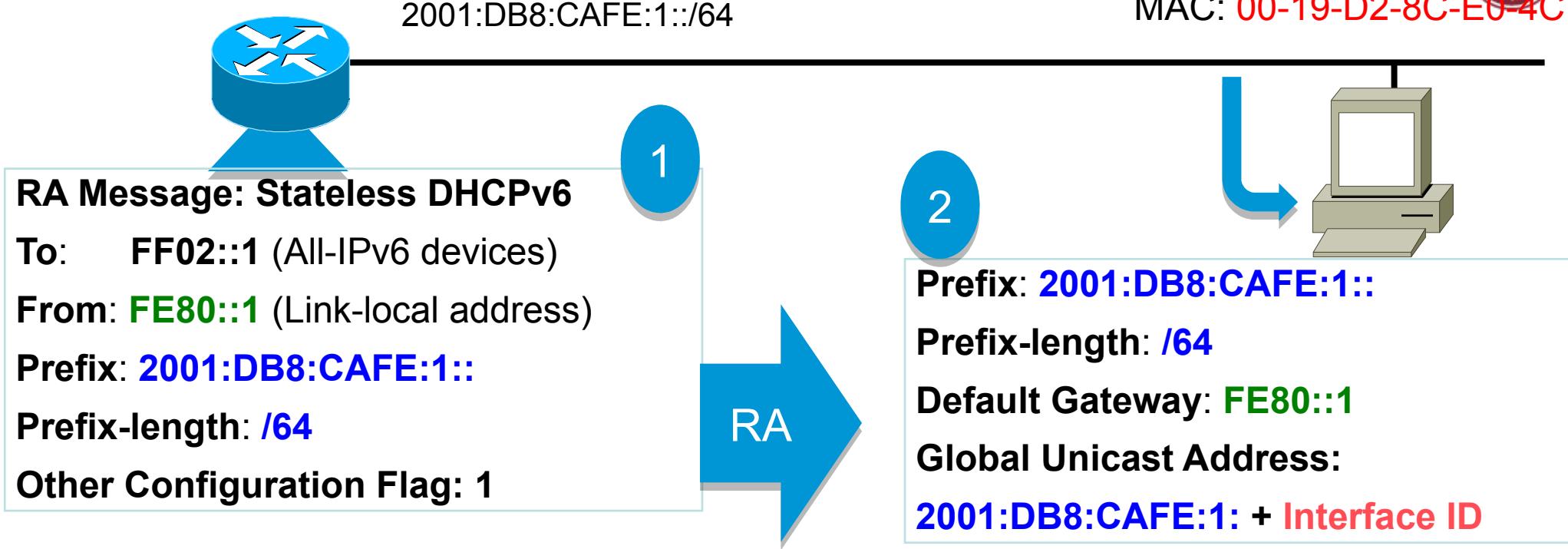
Router as a Stateless DHCPv6 Server



Note: Domain name and DNS server list may be included if router (and end system) support RFC 6106 IPv6 RA Options for DNS Configuration.



SLAAC for Addressing & DNS for Other Information

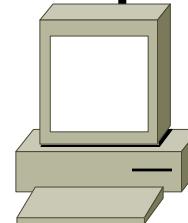


RA Message

- Option 1 and 2: Stateless Address Autoconfiguration
- DHCPv6 Server does not maintain state of addresses
- Option 3: Stateful Address Configuration
- Address received from DHCPv6 Server



```
Router(config)# ipv6 unicast-routing
```



DHCPv6

DHCPv6 Server

Option 1: SLAAC – No DHCPv6 (Default on Cisco routers)

“I’m *everything* you need (Prefix, Prefix-length, Default Gateway)”

Option 2: SLAAC + Stateless DHCPv6 for DNS address

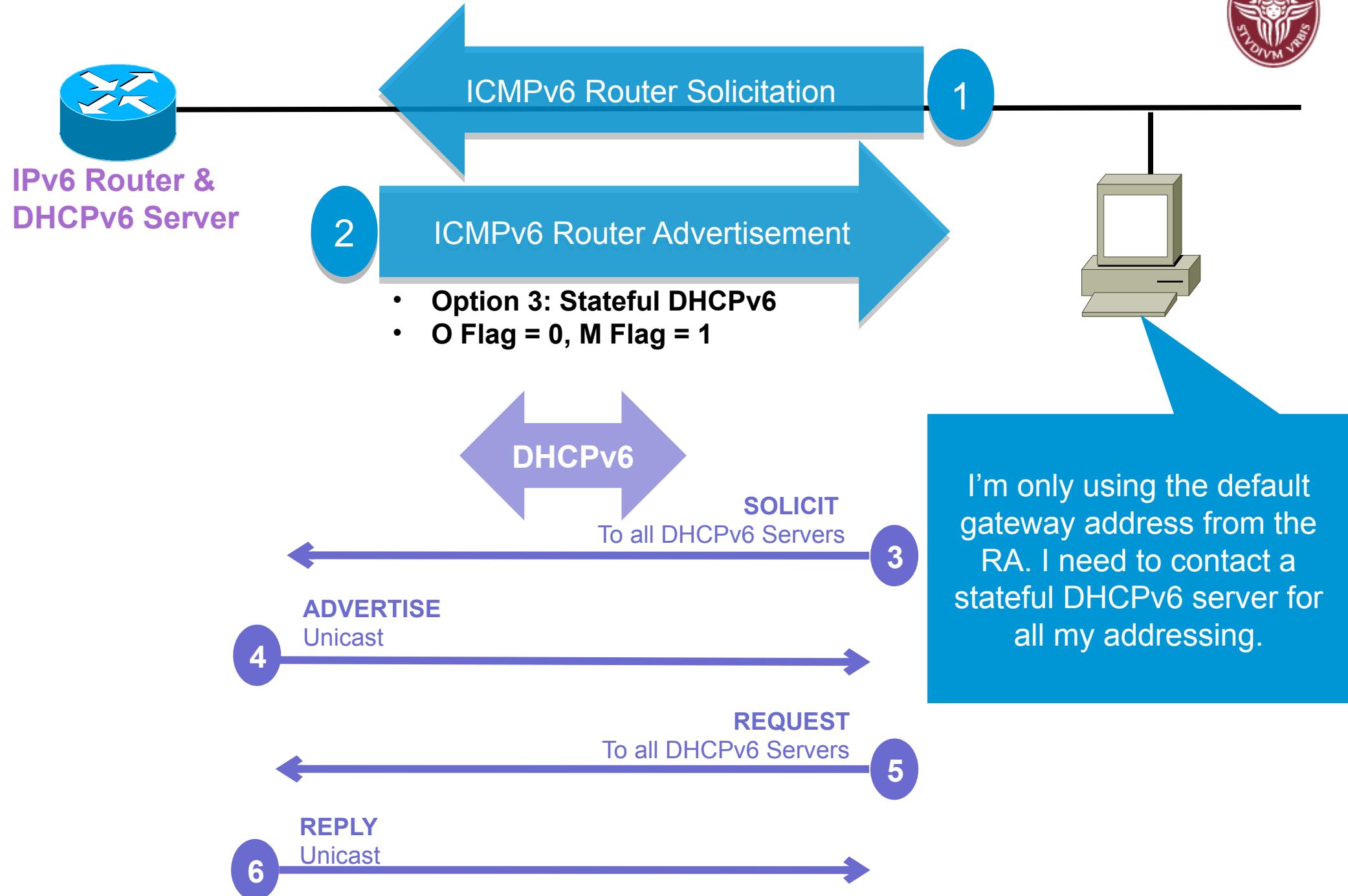
“Here is my information but you need to get other information such as DNS addresses from a **DHCPv6 server**.” (DNS can be in RA)

Option 3: All addressing except default gateway use DHCPv6

“I can’t help you. Ask a **DHCPv6** server for all your information.”

RA

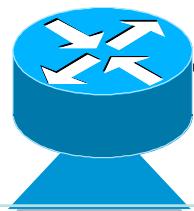
Router as a Stateful DHCPv6 Server



Stateful DHCPv6



I need to get all my addressing from DHCPv6, HOWEVER I will use the router as my default gateway.



2001:DB8:CAFE:2::/64

RA Message: Stateful DHCPv6

To: FF02::1 (All-IPv6 devices)

From: FE80::1 (Link-local address)

Prefix: 2001:DB8:CAFE:2::

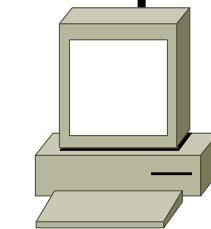
Prefix-length: /64

Managed Configuration Flag: 1

1

RA

2



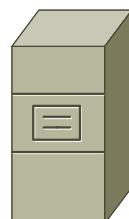
Default Gateway: FE80::1

Global Unicast Address: DHCPv6

2001:DB8:CAFE:1:6909:cb1c:36a0:a595

DNS: 2001:DB8:CAFE:1::99

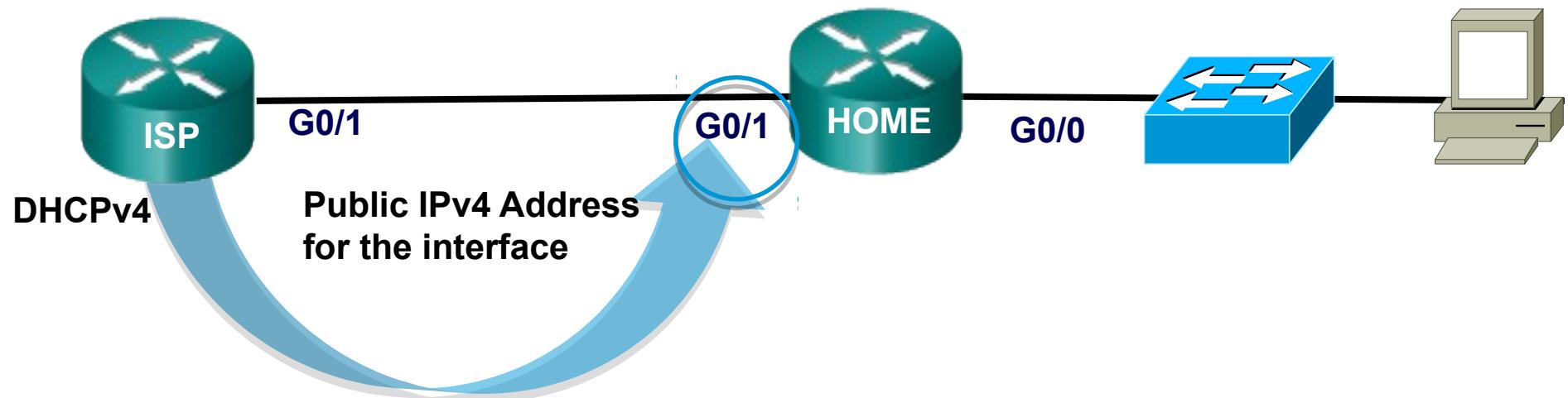
Domain name: cafe.com



Stateful DHCPv6 Server

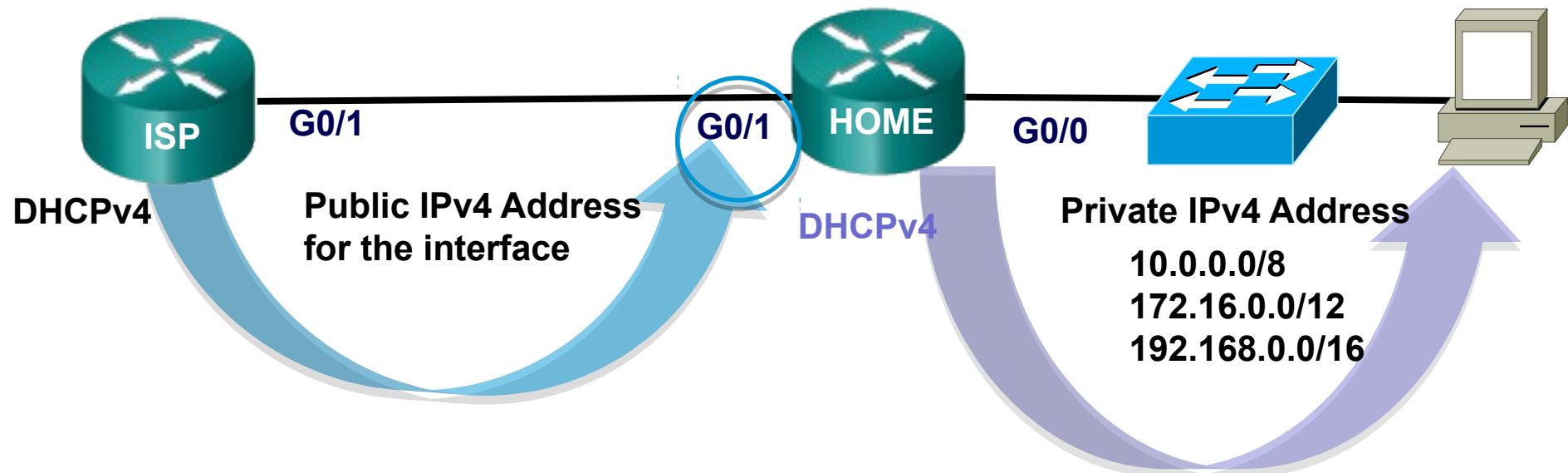
DHCPv6 Prefix Delegation Process

DHCPv4 and Private Addresses for the Home



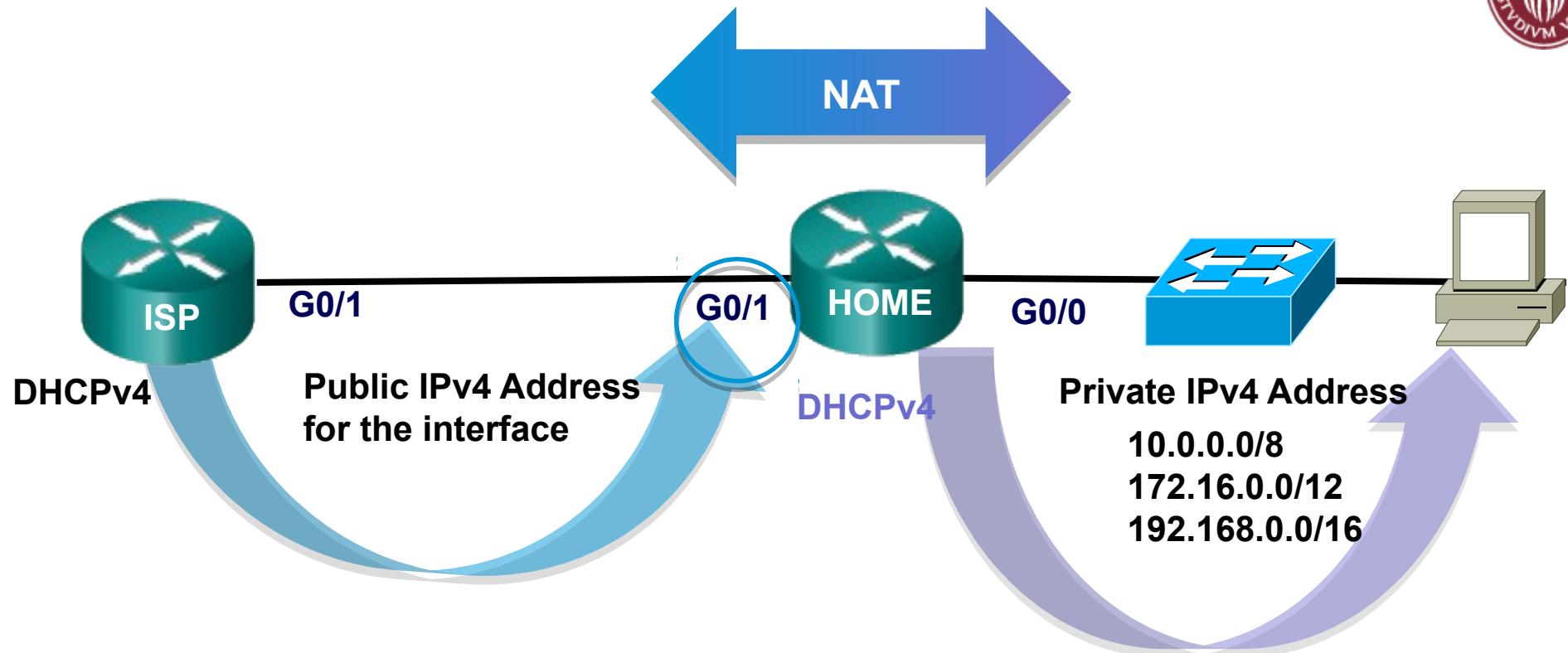
- ISP only has to deliver a public IPv4 address for Home router interface.

DHCPv4 and Private Addresses for the Home



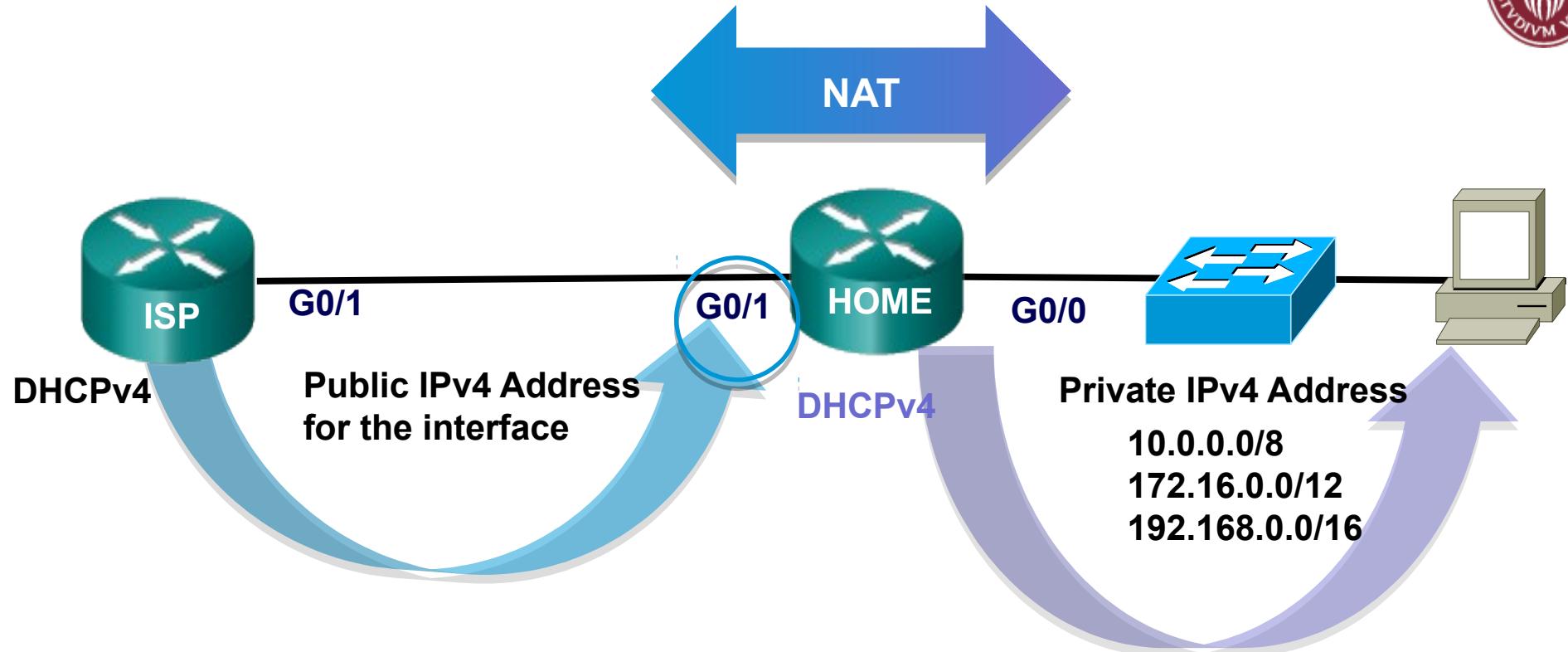
- ISP only has to deliver a public IPv4 address for Home router interface.
- DHCPv4 and RFC 1918 private address space is used for home network.

DHCPv4 and Private Addresses for the Home



- ISP only has to deliver a public IPv4 address for Home router interface.
- DHCPv4 and RFC 1918 private address space is used for home network.
- NAT is used for translation – but has its drawbacks!

DHCPv4 and Private Addresses for the Home



- ISP only has to deliver a public IPv4 address for Home router interface.
- DHCPv4 and RFC 1918 private address space is used for home network.
- NAT is used for translation – but has its drawbacks!
- No NAT between private-public IPv6 (always in debate)

HOME Router's ISP Facing Interface

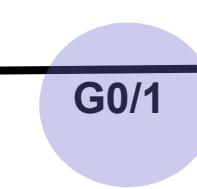


Delegating
Router (DR)

Complete IPv6 Reachability



G0/1

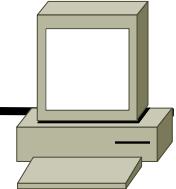


G0/1



HOME-RR
Requesting
Router (RR)

G0/0



- First, HOME's ISP facing interface needs an IPv6 address.

HOME Router's ISP Facing Interface



Delegating
Router (DR)

Complete IPv6 Reachability

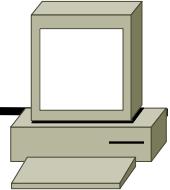


G0/1

G0/1



G0/0



IPv6 Address for the interface:

- SLAAC
- DHCPv6 (Stateful or Stateless)

- First, HOME's ISP facing interface needs an IPv6 address.
- Similar to any IPv6 client it may dynamically get an address using:

HOME Router's ISP Facing Interface



Delegating
Router (DR)

Complete IPv6 Reachability

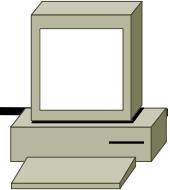


G0/1

G0/1



G0/0



IPv6 Address for the interface:

- SLAAC
- DHCPv6 (Stateful or Stateless)

Requesting
Router (RR)

- First, HOME's ISP facing interface needs an IPv6 address.
- Similar to any IPv6 client it may dynamically get an address using:
 - SLAAC - Using prefix in RA

HOME Router's ISP Facing Interface



Delegating
Router (DR)

Complete IPv6 Reachability

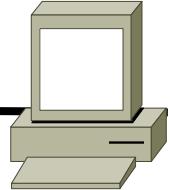


G0/1

G0/1



G0/0



IPv6 Address for the interface:

- SLAAC
- DHCPv6 (Stateful or Stateless)

- First, HOME's ISP facing interface needs an IPv6 address.
- Similar to any IPv6 client it may dynamically get an address using:
 - SLAAC - Using prefix in RA
 - Stateless DHCPv6 – SLAAC but DHCPv6 for DNS address

HOME Router's ISP Facing Interface



Delegating
Router (DR)

Complete IPv6 Reachability

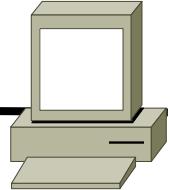


G0/1

G0/1



G0/0



IPv6 Address for the interface:

- SLAAC
- DHCPv6 (Stateful or Stateless)

- First, HOME's ISP facing interface needs an IPv6 address.
- Similar to any IPv6 client it may dynamically get an address using:
 - SLAAC - Using prefix in RA
 - Stateless DHCPv6 – SLAAC but DHCPv6 for DNS address
 - Stateful DHCPv6 - Like DHCPv4

HOME Router's ISP Facing Interface



Delegating
Router (DR)

Complete IPv6 Reachability

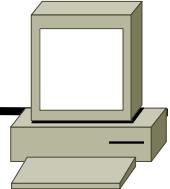


G0/1

G0/1



G0/0



?

IPv6 Address for the interface:

- SLAAC
- DHCPv6 (Stateful or Stateless)

- First, HOME's ISP facing interface needs an IPv6 address.
- Similar to any IPv6 client it may dynamically get an address using:
 - **SLAAC** - Using prefix in RA
 - **Stateless DHCPv6** – SLAAC but DHCPv6 for DNS address
 - **Stateful DHCPv6** - Like DHCPv4
- *What about the address for the HOME LAN?*

DHCPv6 Steps

Delegating Router (DR)



G0/1

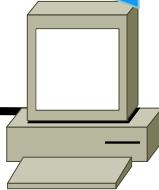
Requesting Router (RR)



G0/1

G0/0

2001:DB8:CAFE:9::
+
Interface ID
(EUI-64 or Random)



DHCPv6-PD REQUEST

1

2

DHCPv6-PD REPLY

Here is a separate IPv6 prefix
for you to give out to your
LAN.

2001:DB8:CAFE:9::
(DNS, domain name)

3

RA with /64 prefix
2001:DB8:CAFE:9::
(DNS, domain name)



That's all for today

- **Questions?**
- See you in the lab
- References:
 - http://www.tcpipguide.com/free/t_InternetProtocolVersion6IPv6IPNextGenerationIPng.htm
 - <http://6diss.6deploy.eu/tutorials/>
 - <http://www.cabrillo.edu/~rgraziani/ipv6-presentations.html>
 - Book chapter 11 (even if quite obsoleted)

Practical Network Defense

Master's degree in Cybersecurity 2021-22

IPv6 addressing lab

Angelo Spognardi
[*spognardi@di.uniroma1.it*](mailto:spognardi@di.uniroma1.it)

*Dipartimento di Informatica
Sapienza Università di Roma*



Lab activity



Main tasks

- Properly configure the topology provided in the lab packages
- Manual configuration
 - Via ip and via interfaces file
- Automatic configuration
 - Via SLAAC
 - Via SLAAC + stateless DHCP



Reference links

- Linux ipv6 configuration: ipv6 sysctl
 - <https://www.kernel.org/doc/Documentation/networking/ip-sysctl.txt>
- Routing Advertisement deaemon: radvd
 - <https://manpages.debian.org/testing/radvd/radvd.conf.5.en.html>
 - https://www.linuxtopia.org/online_books/network_administration_guides/Linux+IPv6-HOWTO/hints-daemons-radvd.html
- dibbler DHCPv6 server/client
 - <https://klub.com.pl/dhcpv6/doc/dibbler-user.pdf>
- dnsmasq network Swiss-knife:
 - <https://thekelleys.org.uk/dnsmasq/docs/dnsmasq-man.html#index>



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/open?id=15WLXIlTWXQnZuXEdYk2WSM5KlFa9Fqx>
- 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>
- You can move in the directory and run **lstart**
 - **NOTE:** the first **lstart** attempt can (...will...) fail



Lab activity: ex1



Exercise 1: pnd-labs/lab2/ex1

- Manually configure pc1, pc2, pc3 and pc4 in order to be in two different subnetworks and r1 to be the default gateway for all of the hosts
 - See the README file for the addresses to assign
 - Configure pc1 using the **interfaces** file
 - Configure pc2 using the **ip** command
 - Configure pc3 using the **ifconfig** command
- The DNS server can be the server used by the host machine
 - This should be used also in the r1
- The default gateway must be the r1 host
 - Remember: its link-local address
- Verify connectivity within the network with ping
 - See the difference when pinging a link-local address and a GUA



Lab activity: ex2



Exercise 2: pnd-labs/lab2/ex2

- Configure the four PC in order to receive their networking configuration using SLAAC
- See the README file for the different settings
 - Start with the interface file, then set the `sysctl` parameters accordingly to the specifications
- Capture the router advertisements/solicitation sent in the network
 - The `radvd` has to be started manually, so that you can launch `tcpdump` before
- Verify connectivity within the network with `ping`



Understanding addr_gen_mode parameter

- Defines how link-local and autoconf addresses are generated.
 - 0: generate address based on EUI64 (default)
 - 1: do no generate a link-local address, use EUI64 for addresses generated from autoconf
 - 2: generate stable privacy addresses, using the secret from `stable_secret` (RFC7217, see **stable_secret** parameter)
 - This allows for a balance between privacy and stability
 - 3: generate stable privacy addresses, using a random secret if unset



Understanding use_tempaddr parameter

- Preference for Privacy Extensions (RFC3041).
 - ≤ 0 : disable Privacy Extensions
 - $= 1$: enable Privacy Extensions, but prefer publicaddresses over temporary addresses.
 - > 1 : enable Privacy Extensions and prefer temporary addresses over public addresses.
- See also:
 - temp_valid_lft - INTEGER
 - valid lifetime (in sec) for temporary addresses (default: 604800, 7 days)
 - temp_preferred_lft - INTEGER
 - preferred lifetime (in sec) for temporary addresses (default: 86400, 1 day)



Lab activity: ex3



Exercise 3: pnd-labs/lab2/ex3

- Configure the router and the PC
 - Router has to correctly advertise prefix, route and stateless directive (namely, via sysctl, dnsmasq)
 - PCs have to receive their networking configuration using SLAAC and stateless DHCP
- See the README file for the different settings
- Capture the router advertisements/solicitation sent in the network
 - You should start the dnsmasq in foreground (-d option), so that you can launch tcpdump before
- Verify connectivity within the network with ping



Dnsmasq

- Very comfortable with dual stack hosts
- It handles both IPv4 and IPv6
- With DHCP it is very useful:
 - You can use DHCPv4 to get IPv4 configuration AND to send your hostname
 - You can use DHCPv6 to get IPv6 configuration
 - If the dnsmasq is also the DNS it knows ALL the hostnames in a quite automated way



That's all for today

- **Questions?**
- See you next lecture!!

Practical Network Defense

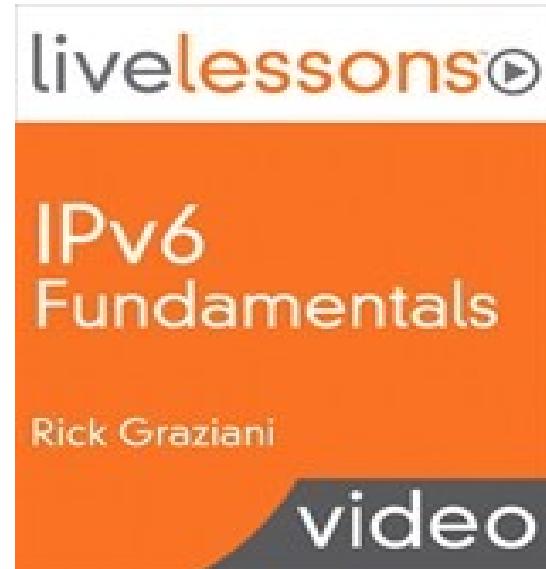
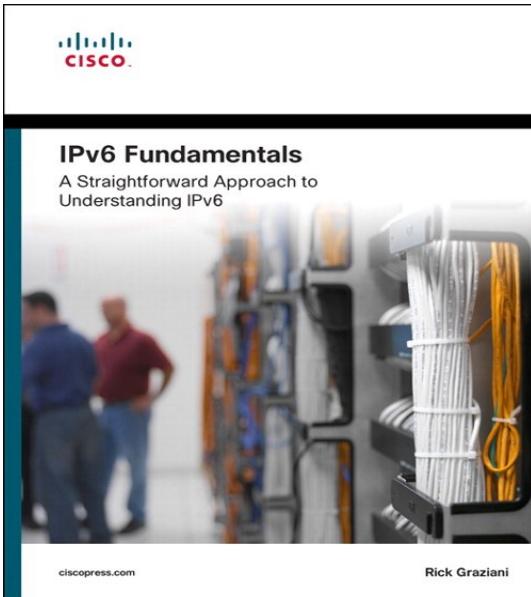
Master's degree in Cybersecurity 2021-22

IPv6: protocol overview

Angelo Spognardi
spognardi@di.uniroma1.it

*Dipartimento di Informatica
Sapienza Università di Roma*

Material taken from Rick Graziani IPv6 courses



IPv6 Fundamentals: A Straightforward Approach to Understanding IPv6

- By Rick Graziani
- ISBN-10: 1-58714-313-5

IPv6 Fundamentals LiveLessons: A Straightforward Approach to Understanding IPv6

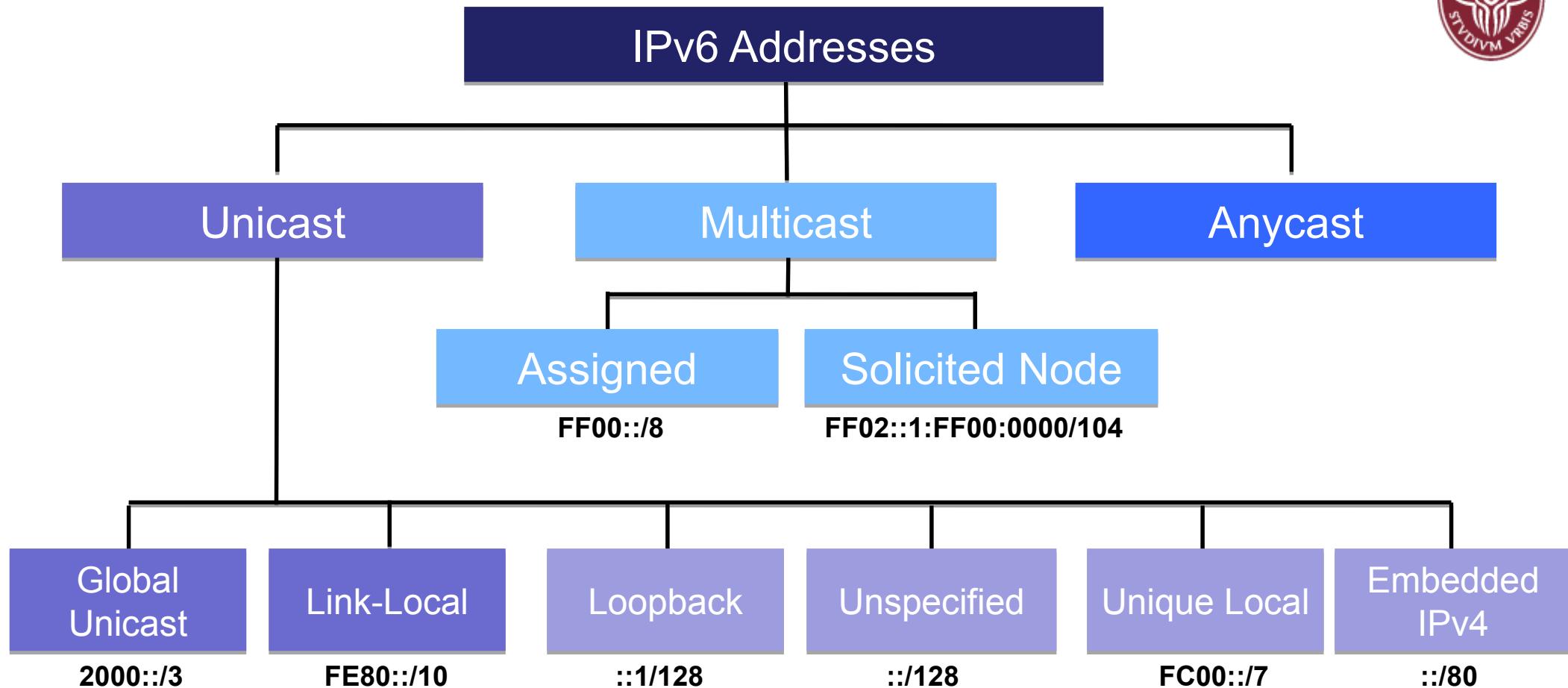
- By Rick Graziani
- ISBN-10: 1-58720-457-6



Recap last lectures

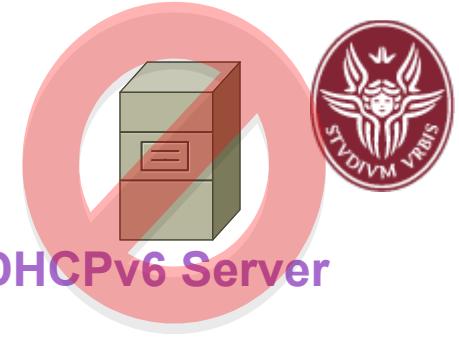
- IPv6 main features
- IPv6 address types
 - Global Unicast Address
 - Local-link unicast Address
- IPv6 dynamic assignment options
 - SLAAC
 - SLAAC+DNS via DHCPv6
 - Stateful DHCPv6
- IPv6 prefix delegation, via DHCPv6-PD

IPv6 Address Types

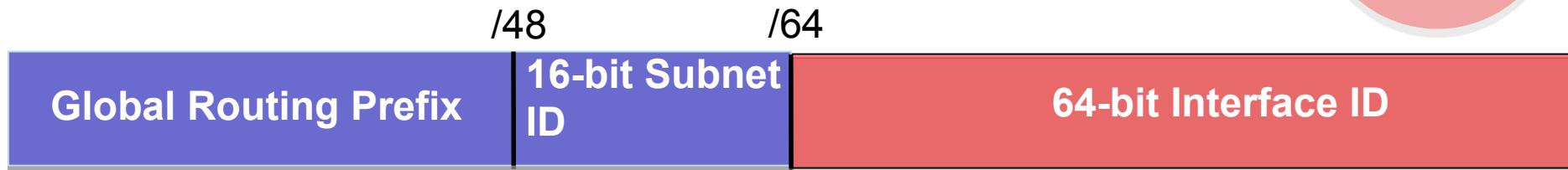


IPv6 does not have a “broadcast” address.

SLAAC: Random 64-bit Interface ID



DHCPv6 Server



Operating System

Windows XP,
Server 2003

Windows Vista
and newer

MAC OSX

Linux

EUI-64

Random 64-bit



EUI-64 Process

~~UNKNOWN~~



64-bit Interface ID

SLAAC

Randomly Generated Number
(Privacy Extension)



SLAAC: Temporary Addresses



- Idea: provide additional addresses that have relatively short lifetimes and are used as the source address when originating connections
- Same prefix as a public address, randomized value for the Interface ID
- Short lifetime, usually hours or days
- It is common to have multiple temporary addresses to make sure existing connections can continue while a new temporary address is created for new connections

**Randomly Generated Number
(Privacy Extension)**



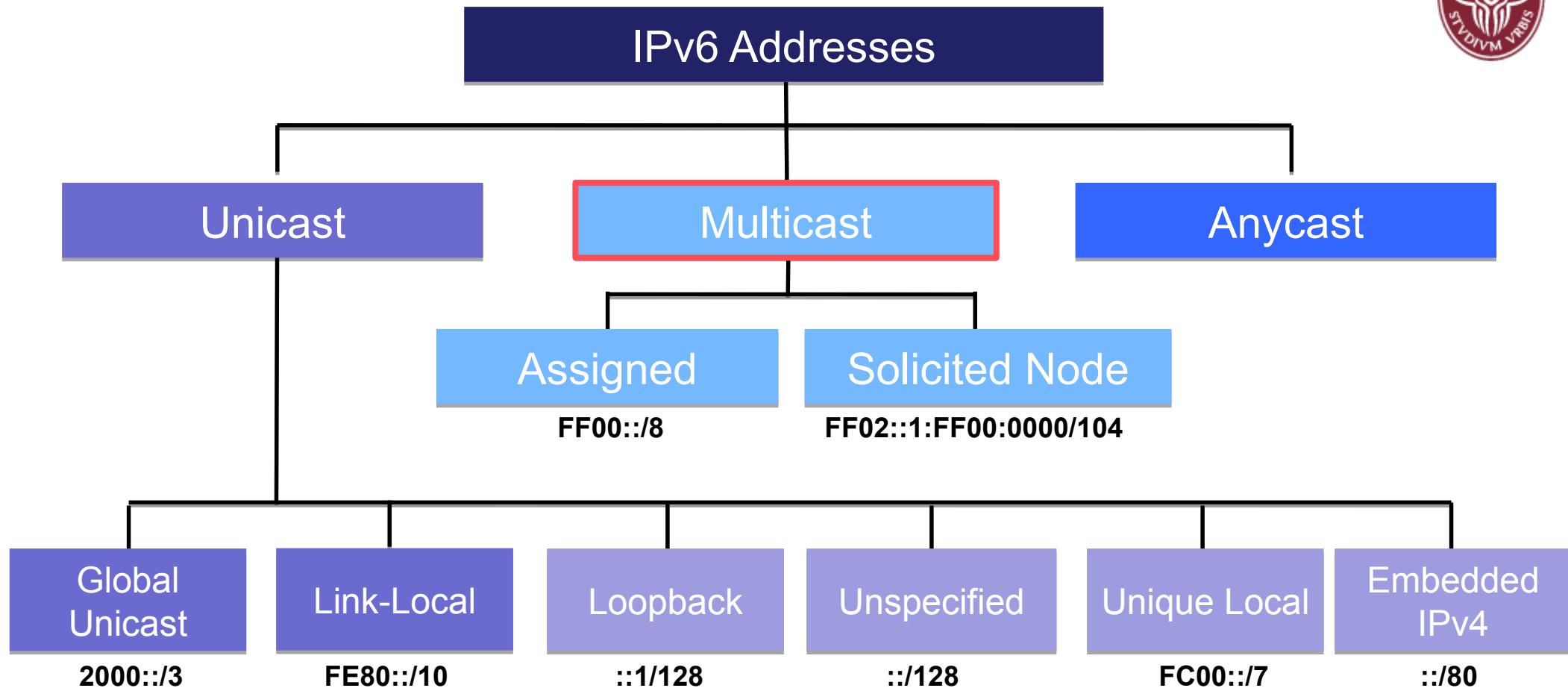
```
LinuxPC# ip -6 addr show dev eth0

2: eth0: <BROADCAST,MULTICAST,UP, LOWER_UP> mtu 1500 state UNKNOWN qlen1000
    inet6 2001:db8:cafe:4:314a:dd3e:762f:e140/64 scope global temporary dynamic
          valid_lft 604747sec preferred_lft 85747sec

    inet6 2001:db8:cafe:4:250:56ff:feaf:2524/64 scope global dynamic
          valid_lft 2591947sec preferred_lft 604747sec

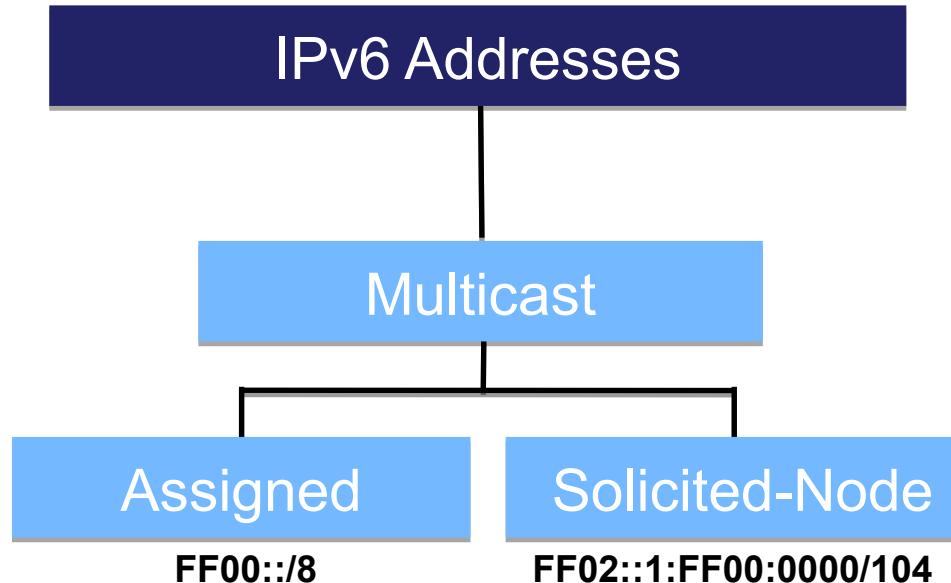
    inet6 fe80::250:56ff:feaf:2524/64 scope link
          valid_lft forever preferred_lft forever
```

IPv6 Address Types



IPv6 does not have a “broadcast” address.

IPv6 Multicast Addresses



- Used by a device to send a single packet to multiple destinations simultaneously (one-to-many).
- Equivalent to 224.0.0.0/4 in IPv4.
- Two types of multicast addresses:
 - Assigned
 - Solicited-Node

IPv6 Multicast Addresses



- **IPv6 Source** – Always a unicast
 - **IPv6 Destination** – Unicast, *multicast*, or anycast.

IPv4

The diagram illustrates the structure of an IPv4 header. It consists of several fields arranged horizontally, each with its corresponding bit width indicated above it. The fields are:

- Version (4 bits)
- IHL (4 bits)
- Type of Service (8 bits)
- Total Length (16 bits)
- Identification (16 bits)
- Flags (4 bits)
- Fragment Offset (13 bits)
- Time to Live (8 bits)
- Protocol (8 bits)
- Header Checksum (16 bits)
- Source Address (32 bits)
- Destination Address (32 bits)
- Options (variable length)
- Padding (variable length)

The fields are color-coded: Version, IHL, Type of Service, Total Length, Identification, Flags, Fragment Offset, Time to Live, Protocol, Header Checksum, Source Address, Destination Address, Options, and Padding are in light blue; the fields from Identification to Fragment Offset are in white.

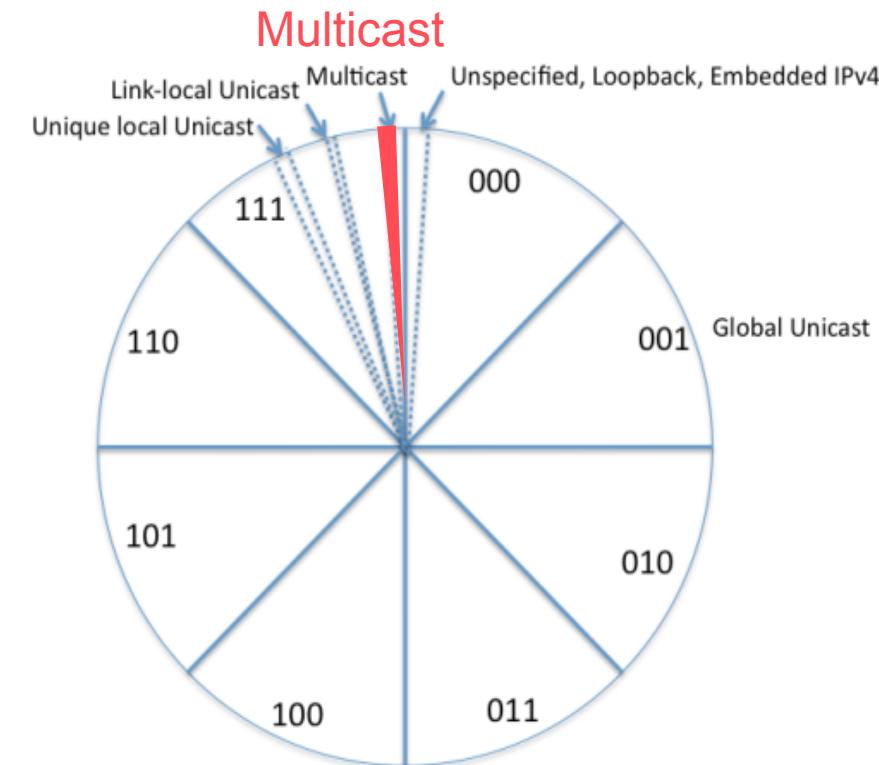
IPv6

	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																												

Multicast Range



IPv6 multicast addresses have the prefix **FF00::/8**

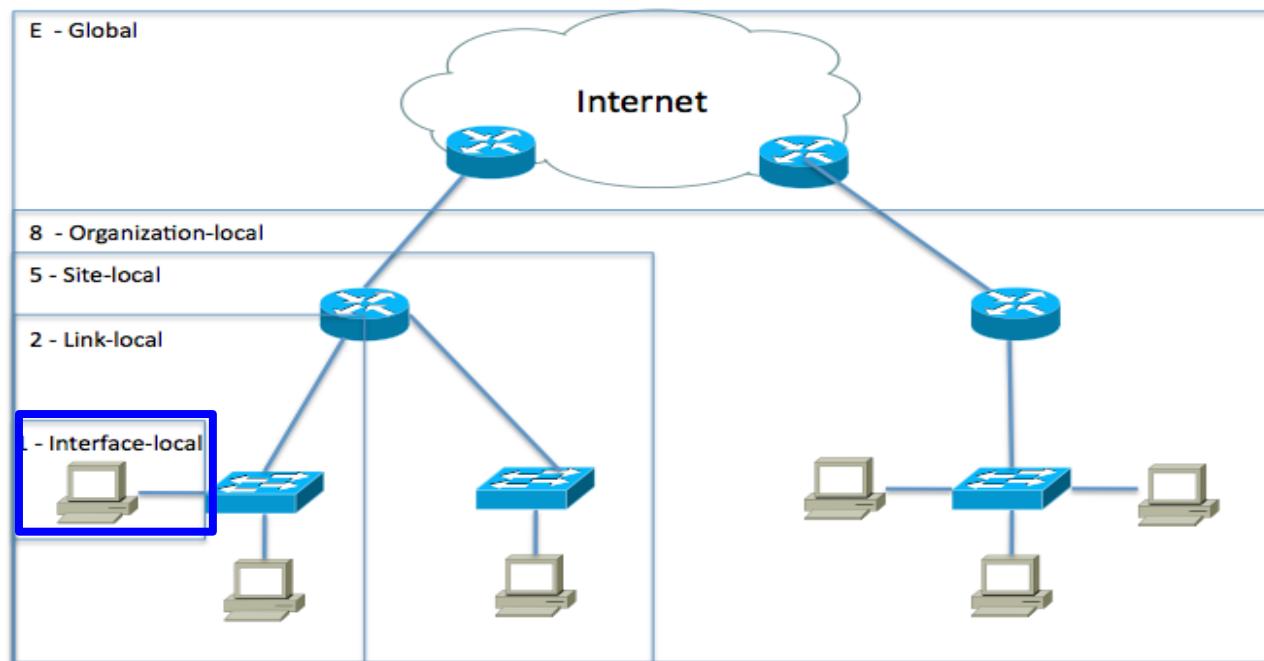


The remaining portion of IPv6 address space are reserved by IETF for future use.

IPv6 Multicast Addresses - Scope



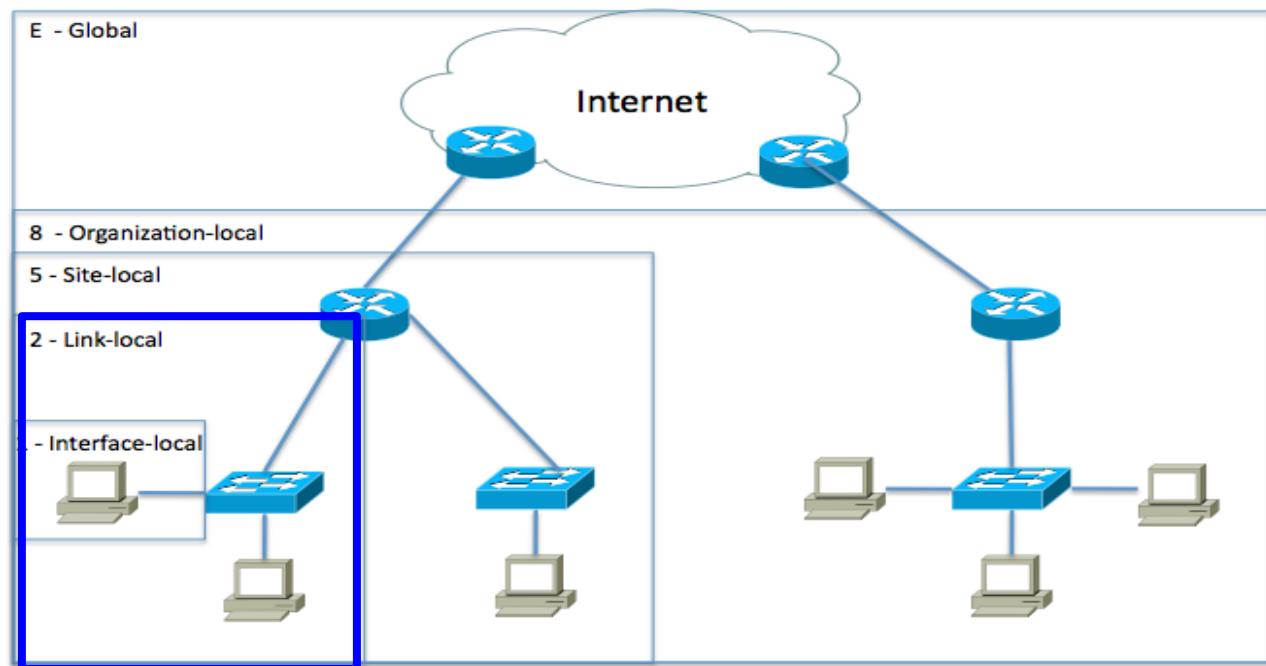
- Scope is a 4-bit field used to define the range of the multicast packet.
- **Scope** (partial list):
 - 0 Reserved
 - 1 Interface-Local scope



IPv6 Multicast Addresses - Scope



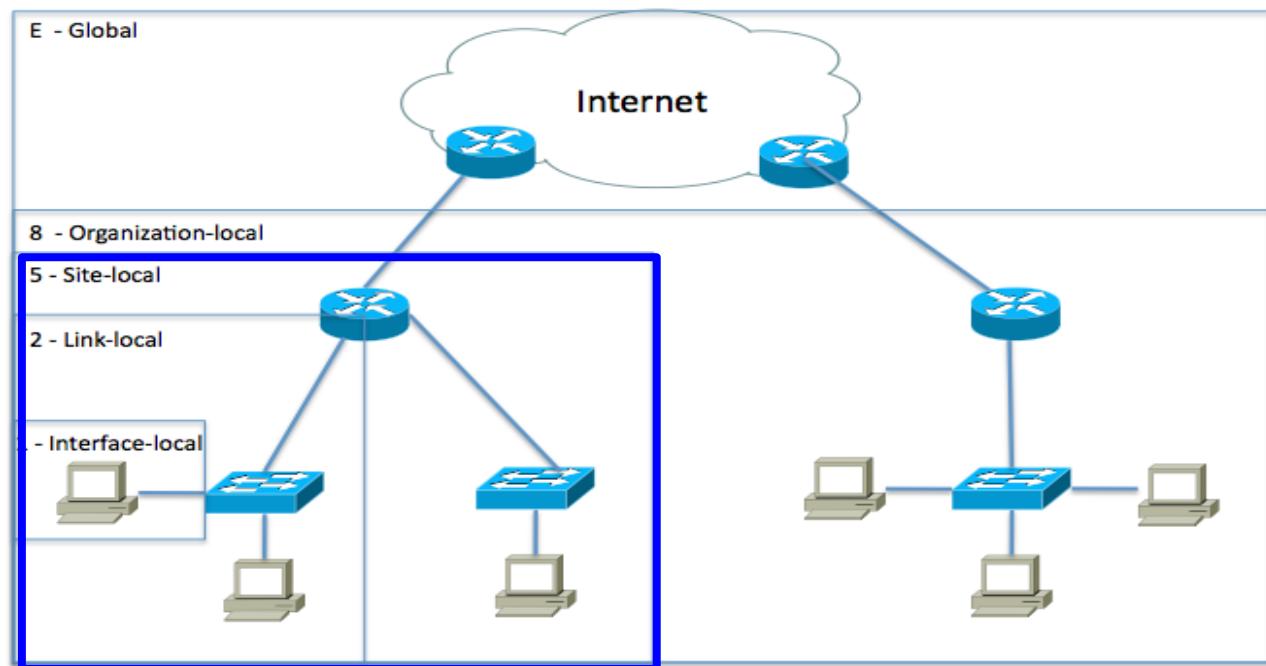
- Scope is a 4-bit field used to define the range of the multicast packet.
- **Scope** (partial list):
 - 0 Reserved
 - 1 Interface-Local scope
 - 2 Link-Local scope



IPv6 Multicast Addresses - Scope



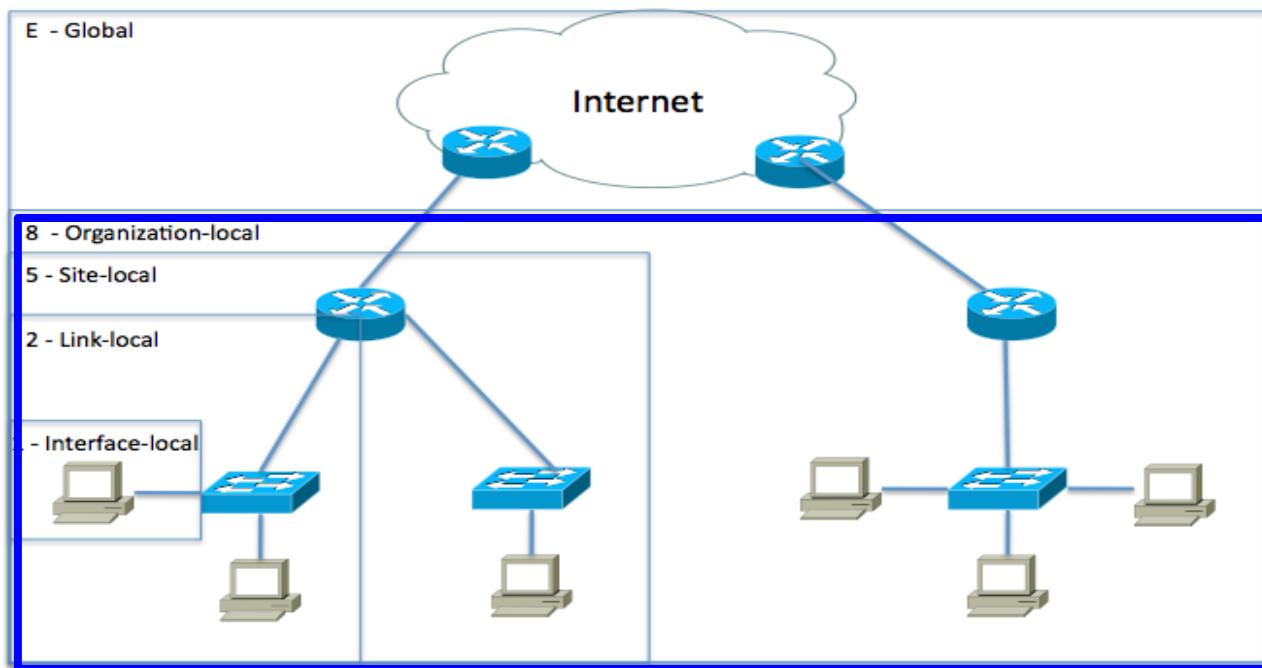
- Scope is a 4-bit field used to define the range of the multicast packet.
- **Scope** (partial list):
 - 0 Reserved
 - 1 Interface-Local scope
 - 2 Link-Local scope
 - 5 Site-Local scope



IPv6 Multicast Addresses - Scope



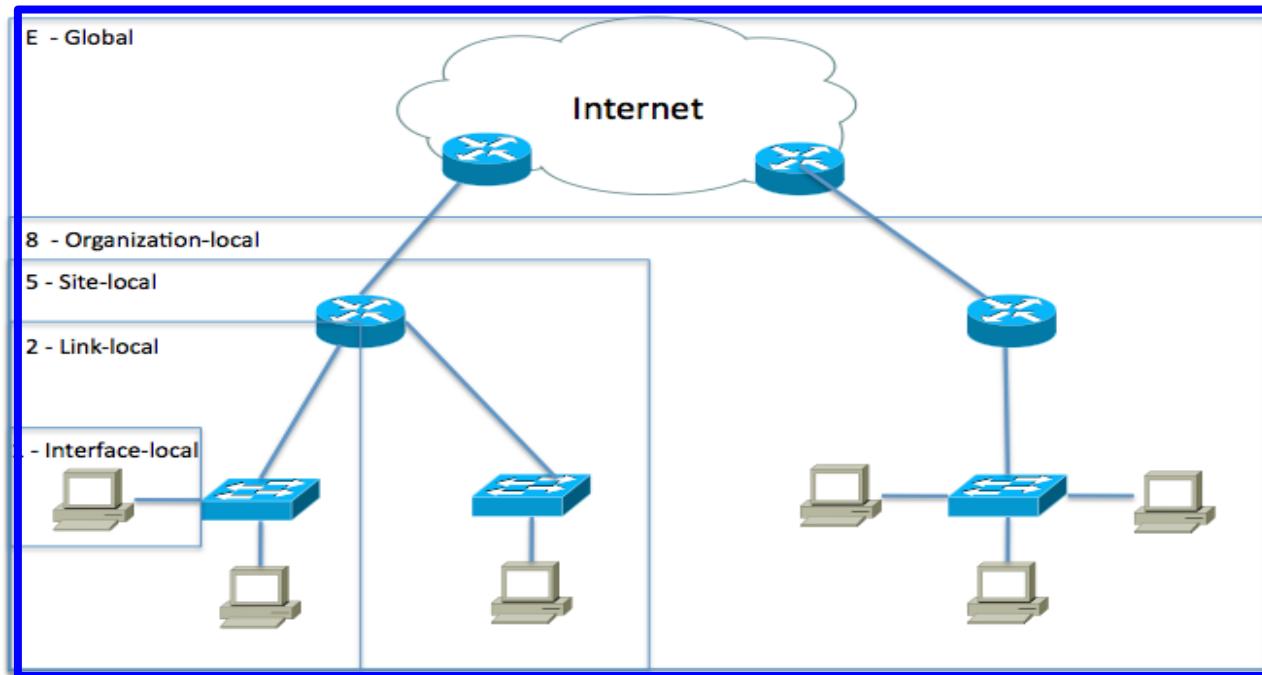
- Scope is a 4-bit field used to define the range of the multicast packet.
- **Scope** (partial list):
 - 0 Reserved
 - 1 Interface-Local scope
 - 2 Link-Local scope
 - 5 Site-Local scope
 - 8 Organization-Local scope



IPv6 Multicast Addresses - Scope



- Scope is a 4-bit field used to define the range of the multicast packet.
- **Scope** (partial list):
 - 0 Reserved
 - 1 Interface-Local scope
 - 2 Link-Local scope
 - 5 Site-Local scope
 - 8 Organization-Local scope
 - E Global scope

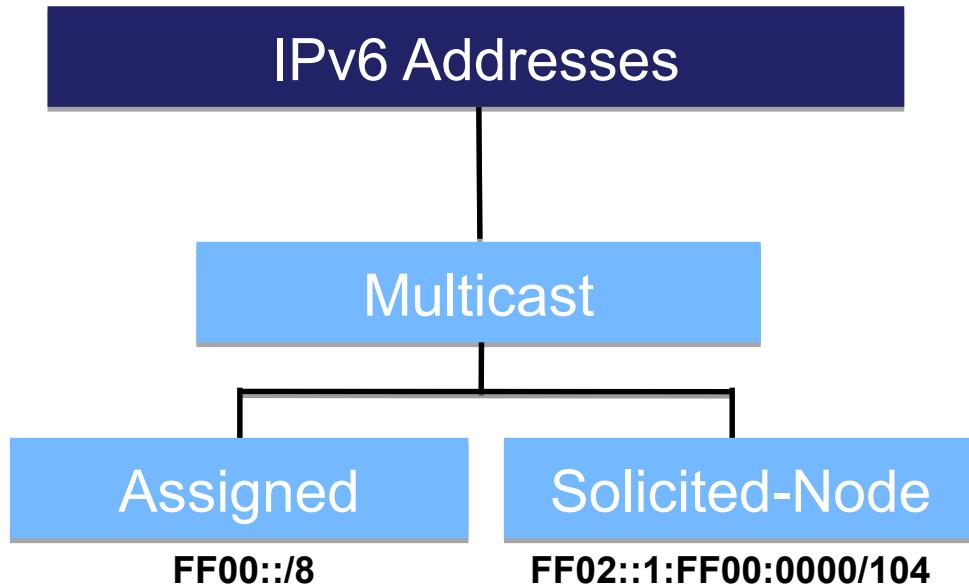


IPv6 Multicast Addresses - Flag



- **Flag**
 - 0 - Permanent, well-known multicast address assigned by IANA.
 - Includes both assigned and solicited-node multicast addresses.
 - 1 - Non-permanently-assigned, "dynamically" assigned multicast address.
 - An example might be FF18::CAFE:1234, used for a multicast application with organizational scope.

Assigned IPv6 Multicast Addresses



- RFC 2375, IPv6 Multicast Address Assignments, defines the initial assignment of IPv6 multicast addresses that have permanently assigned Global IDs.
- Reference for assigned multicast addresses:
 - (IANA) IPv6 Multicast Address Space Registry -
<http://www.iana.org/assignments/ipv6-multicast-addresses/ipv6-multicast-addresses.xhtml>

IPv6 Multicast Address Space Registry

Last Updated

2014-08-17

Expert(s)

Stig Venaas

Note

IPv6 multicast addresses are defined in "IP Version 6 Addressing Architecture" [[RFC4291](#)]. This defines fixed scope and variable scope multicast addresses.

IPv6 multicast addresses are distinguished from unicast addresses by the value of the high-order octet of the addresses: a value of 0xFF (binary 11111111) identifies an address as a multicast address; any other value identifies an address as a unicast address.

The rules for assigning new IPv6 multicast addresses are defined in [[RFC3307](#)]. IPv6 multicast addresses not listed below are reserved.

Available Formats



Registries included below

- [IPv6 Multicast Address Scopes](#)
- [Node-Local Scope Multicast Addresses](#)
- [Link-Local Scope Multicast Addresses](#)
- [Site-Local Scope Multicast Addresses](#)
- [Variable Scope Multicast Addresses](#)
- [Source-Specific Multicast block](#)

Assigned Multicast Addresses with Link-local Scope



Flag = 0, Assigned multicast
Scope = 2, Link-local scope



Prefix	Flag	Scope	Predefined Group ID	Compressed Format	Description (IPv6 assumed)
FF	0	2	0:0:0:0:0:0:1	FF02::1	All-devices
FF	0	2	0:0:0:0:0:0:2	FF02::2	All-routers
FF	0	2	0:0:0:0:0:0:5	FF02::5	OSPF routers
FF	0	2	0:0:0:0:0:0:6	FF02::6	OSPF DRs
FF	0	2	0:0:0:0:0:0:9	FF02::9	RIP routers
FF	0	2	0:0:0:0:0:0:A	FF02::A	EIGRP routers
FF	0	2	0:0:0:0:0:1:2	FF02::1:2	DHCP servers/relay agents

Assigned Multicast Addresses with Site-local Scope



Flag = 0, Assigned multicast
Scope = 5, Site-local scope



Prefix	Flag	Scope	Predefined Group ID	Compressed Format	Description (IPv6 assumed)
FF	0	5	0:0:0:0:0:0:2	FF05::2	All-routers
FF	0	5	0:0:0:0:0:1:3	FF05::1:3	All DHCP servers

- Used to communicate within a “site”, possibly routed within the site.
- Must have IPv6 multicast routing enabled:
Router(config) # **ipv6 multicast-routing**
- DHCPv6, relay agents and DHCPv6 multicast addresses are included in Lesson 8.*

“All IPv6 Devices” Assigned Multicast Address



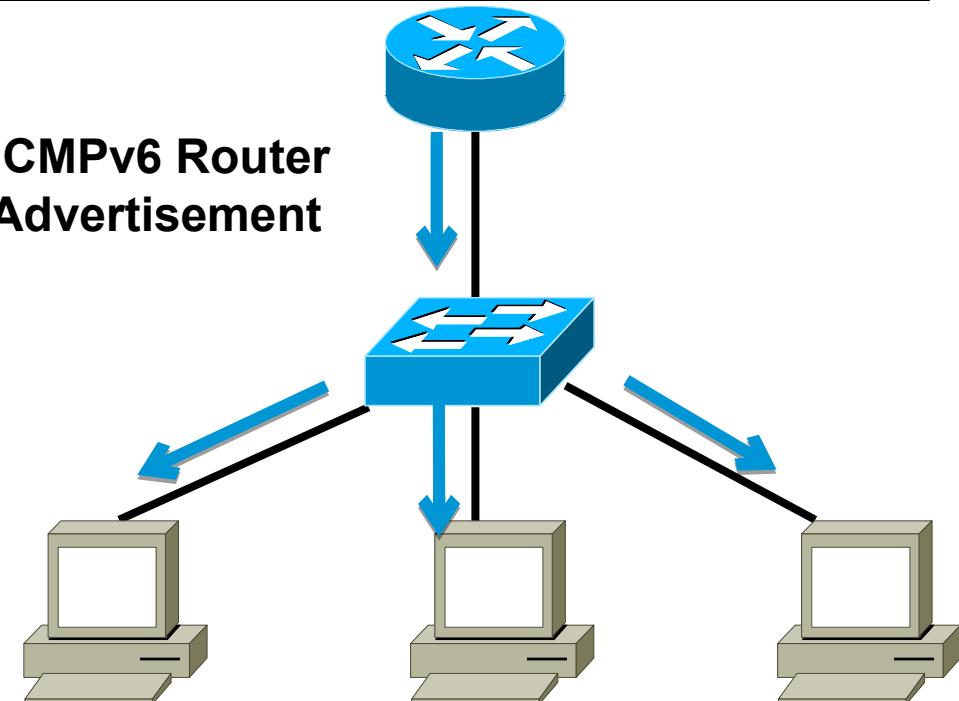
ICMPv6 Router Advertisement

FF02::1	FE80::1	Rest of IPv6 Packet
---------	---------	---------------------

Destination
IPv6 Address Source
IPv6 Address

```
Router (config) # ipv6 unicast-routing
```

- **FF02::1 – All IPv6 Devices**
- All IPv6 devices, including the router, belong to this group.
- Every IPv6 device will listen and process packets to this address.
- Isn't this the same as a broadcast?
- No, because it maps to a Layer 2 MAC address which is more efficient... coming soon!



“All IPv6 Routers” Assigned Multicast Address



ICMPv6 Router Solicitation

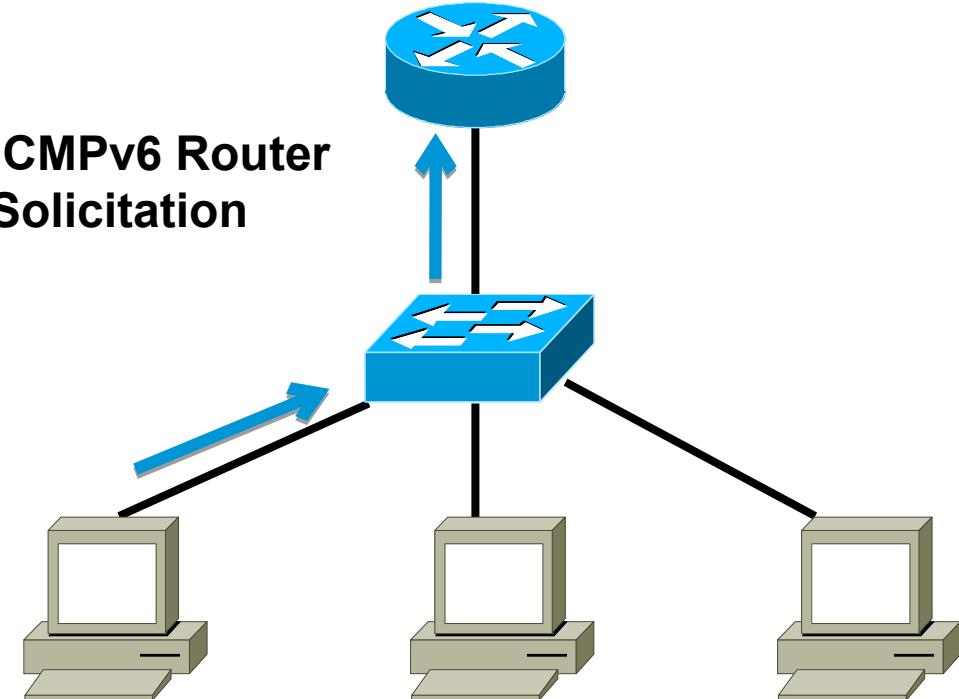
FF02::2	FE80::12:3456: 7890:ABCD	Rest of IPv6 Packet
---------	-----------------------------	---------------------

Destination
IPv6 Address

Source
IPv6 Address

```
Router (config) # ipv6 unicast-routing
```

- **FF02::2 – All IPv6 Routers**
- All IPv6 routers belong to this group. (Process these packets.)
- Used by devices to communicate with an IPv6 Router.



IPv6 vs. IPv4

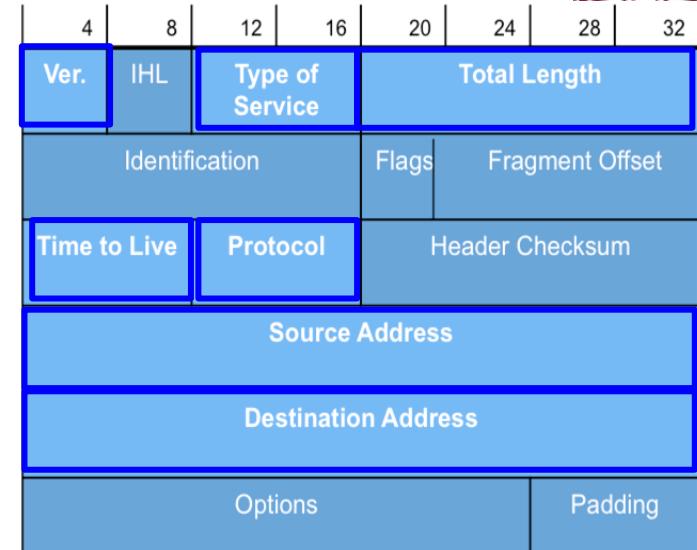
IPv6 Header



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

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

IPv4



IPv6



Similar fields

IPv6 Version



- **IPv4 Version** contains 4.
 - **IPv6 Version** contains 6.
 - Version 5: Internet Stream Protocol (ST2)

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										

IPv6

Ver.	Traffic Class	Flow Label	Payload Length		Next Header	Hop Limit
		Source Address				
		Destination Address				

IPv4 Internet Header Length



- **IPv4 Internet Header Length (IHL)**
 - Length of IPv4 header in 32-bit words including any Options or Padding.
- **IPv6**
 - IHL for IPv6 is not needed.
 - IPv6 header is fixed at 40 bytes.

IPv4	4	8	12	16	20	24	28	32									
1	Ver.	IHL	Type of Service			Total Length											
2	Identification			Flags	Fragment Offset												
3	Time to Live		Protocol		Header Checksum												
4	Source Address																
5	Destination Address																
?	Options						Padding										

IPv6	4	8	12	16	20	24	28	32	36	40	44	48	52	56	60	64													
8 bytes	Ver.	Traffic Class		Flow Label				Payload Length				Next Header		Hop Limit															
8 bytes	Source Address																												
8 bytes	Destination Address																												
8 bytes																													

40 bytes =



IPv6 Traffic Class



- **IPv4 Type of Service**
 - **IPv6 Traffic Class**
 - Not mandated by any IPv6 RFCs.
 - Same functionality as IPv4.
 - Uses same Differentiated Services technique (RFC 2474) as IPv4.

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													

IPv6

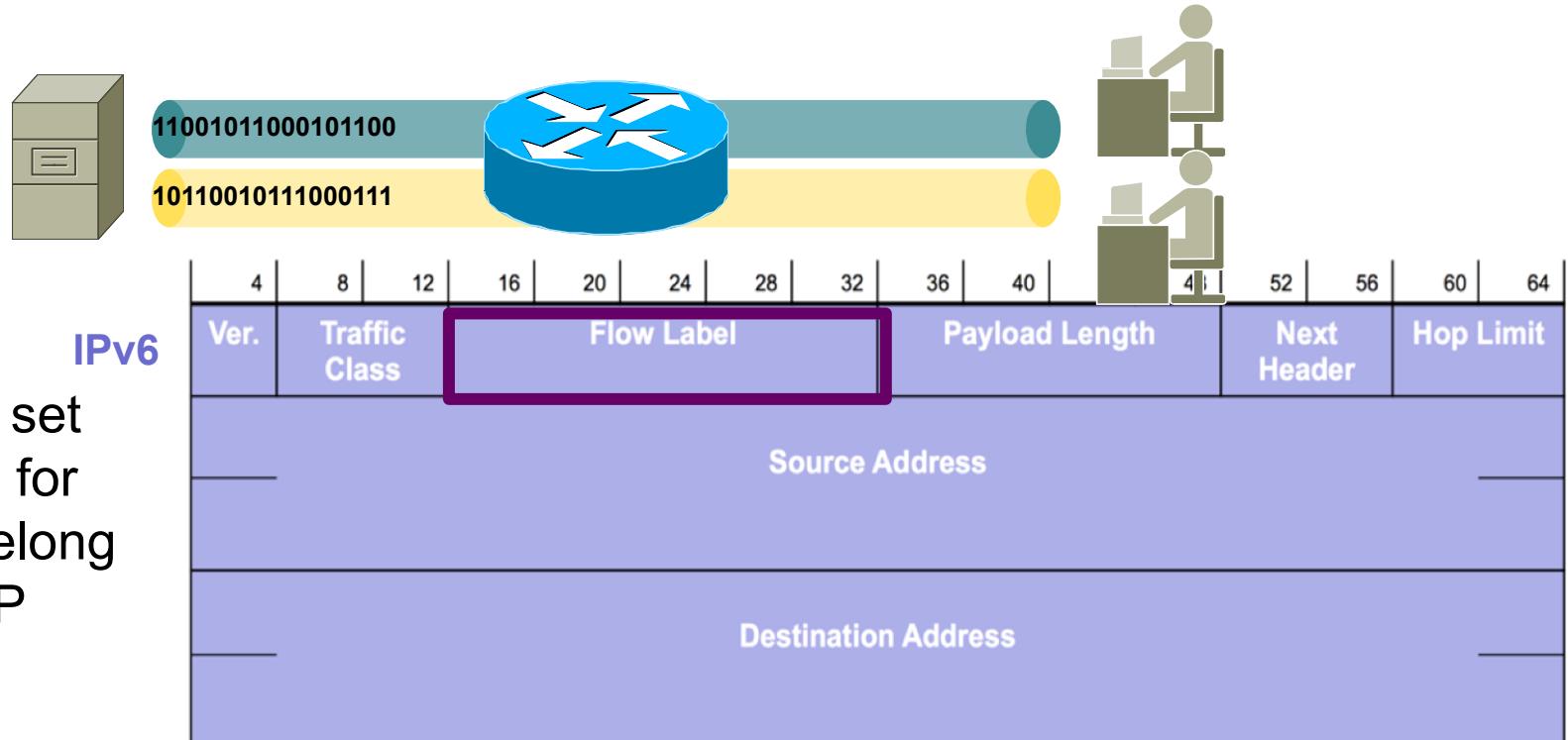
The diagram illustrates the structure of an IPv6 header. The total header length is 64 bytes, indicated by the scale at the top. The header fields are as follows:

- Ver.**: Version (4 bits)
- Traffic Class**: Traffic Class (8 bits)
- Flow Label**: Flow Label (20 bits)
- Payload Length**: Payload Length (16 bits)
- Next Header**: Next Header (8 bits)
- Hop Limit**: Hop Limit (8 bits)
- Source Address**: Source Address (128 bits)
- IP Precedence**: IP Precedence (4 bits)
- Unused**: Unused (4 bits)
- DiffServ Code Point (DSCP)**: DiffServ Code Point (DSCP) (6 bits)
- IP ECN**: IP ECN (2 bits)
- Destination Address**: Destination Address (128 bits)

IPv6 Flow Label



- New field in IPv6 – not part of IPv4.
- Flow label is used to identify the packets in a common stream or flow.
- Traffic from source to destination share a common flow label.
- RFC 6437 IPv6 Flow Label Specification



- Many systems set the Flow Label for packets that belong to different TCP sessions

IPv6 Payload Length

- IPv4 Total Length** – Number of bytes of the IPv4 header (options) + data.

IPv4 Header

Data (Payload)

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									

IPv4

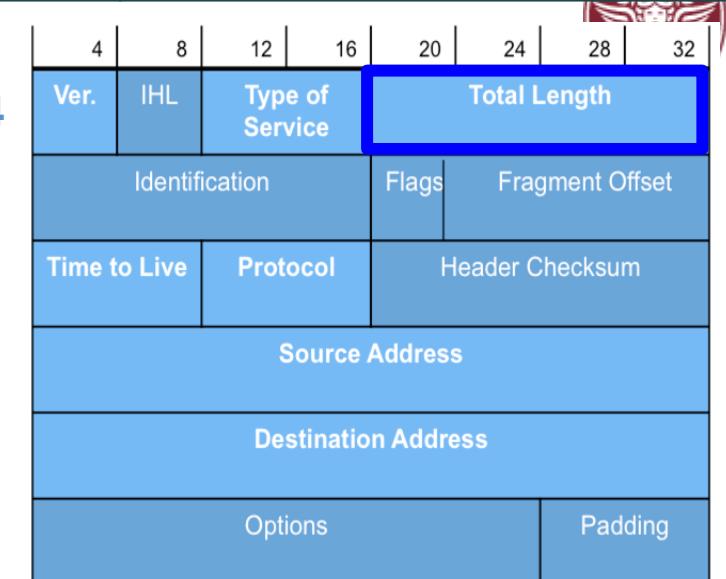
4	8	12	16	20	24	28	32	36	40	44	48	52	56	60	64													
Ver.	Traffic Class	Flow Label				Payload Length				Next Header		Hop Limit																
Source Address																												
Destination Address																												

IPv6

IPv6 Payload Length

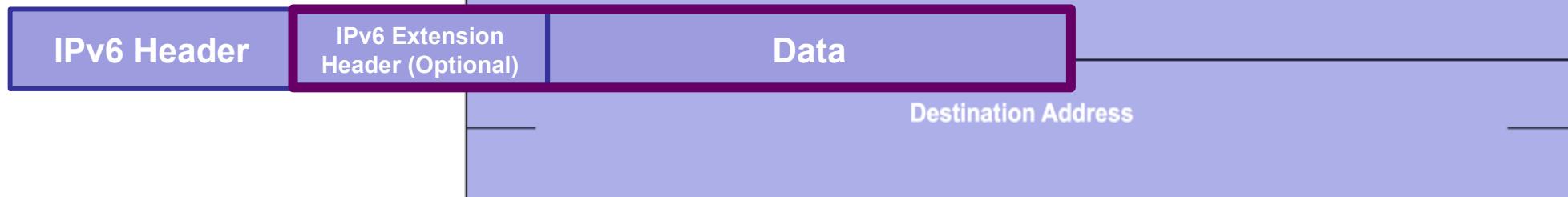
- **IPv4 Total Length** – Number of bytes of the IPv4 header (options) + data.
- **IPv6 Payload Length** – Number of bytes of the payload.
 - Does not include the main IPv6 header.
 - Includes extension headers + data

IPv4



IPv6

Payload



IPv4 Fragmentation



- IPv4 fields used for fragmentation and reassembly.
 - Intermediate devices such as IPv6 routers do not perform fragmentation.
 - Any fragmentation needed will be handled by the source using an extension header.

IPv4

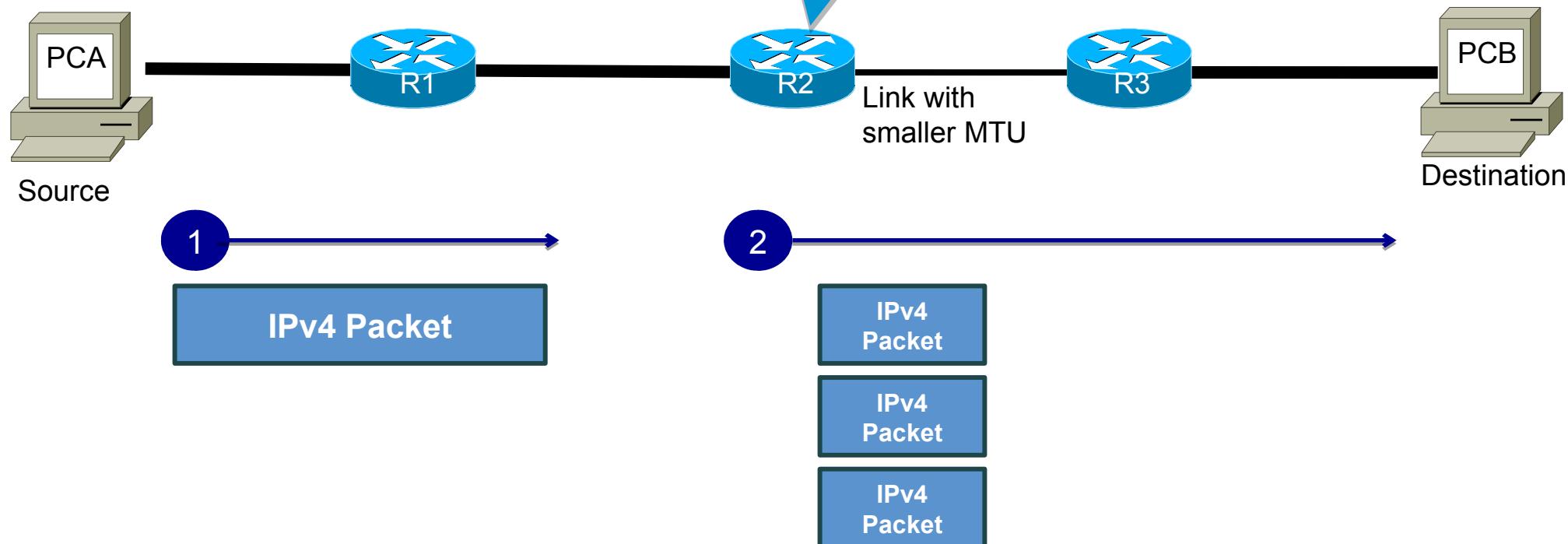
4	8	12	16	20	24	28	32									
Ver.	IHL	Type of Service	Total Length													
Identification			Flags	Fragment Offset												
Time to Live		Protocol	Header Checksum													
Source Address																
Destination Address																
Options						Padding										

IPv6

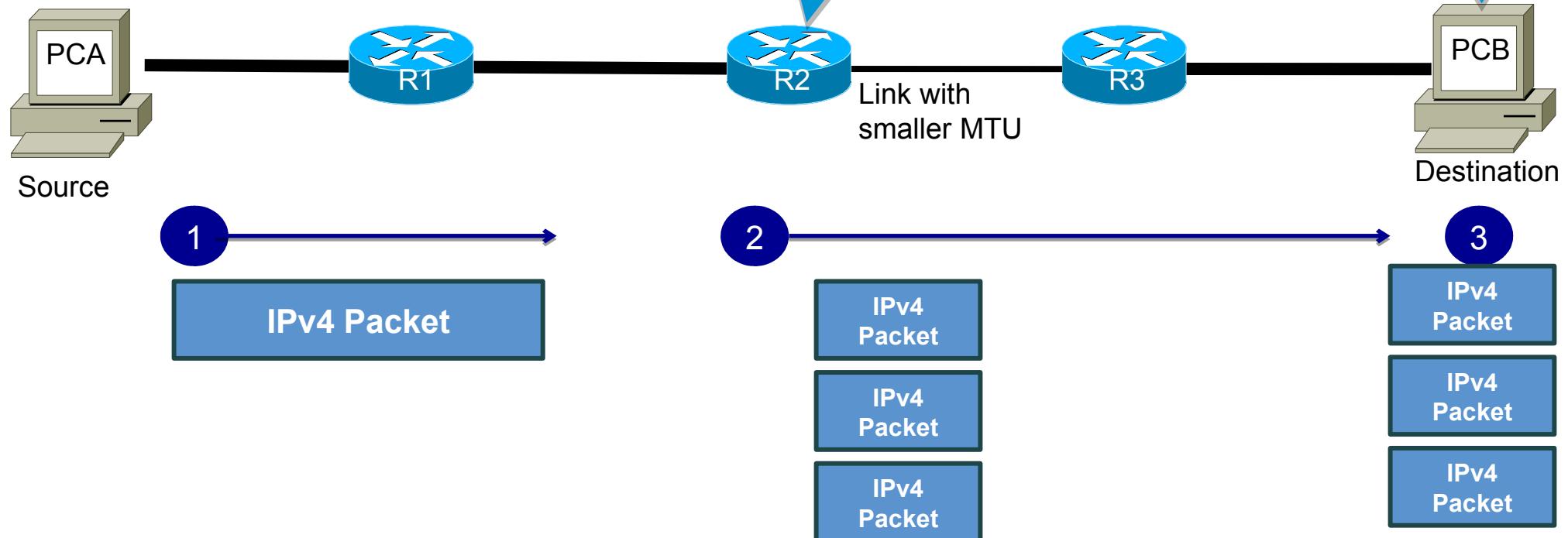
IPv4 Fragmentation



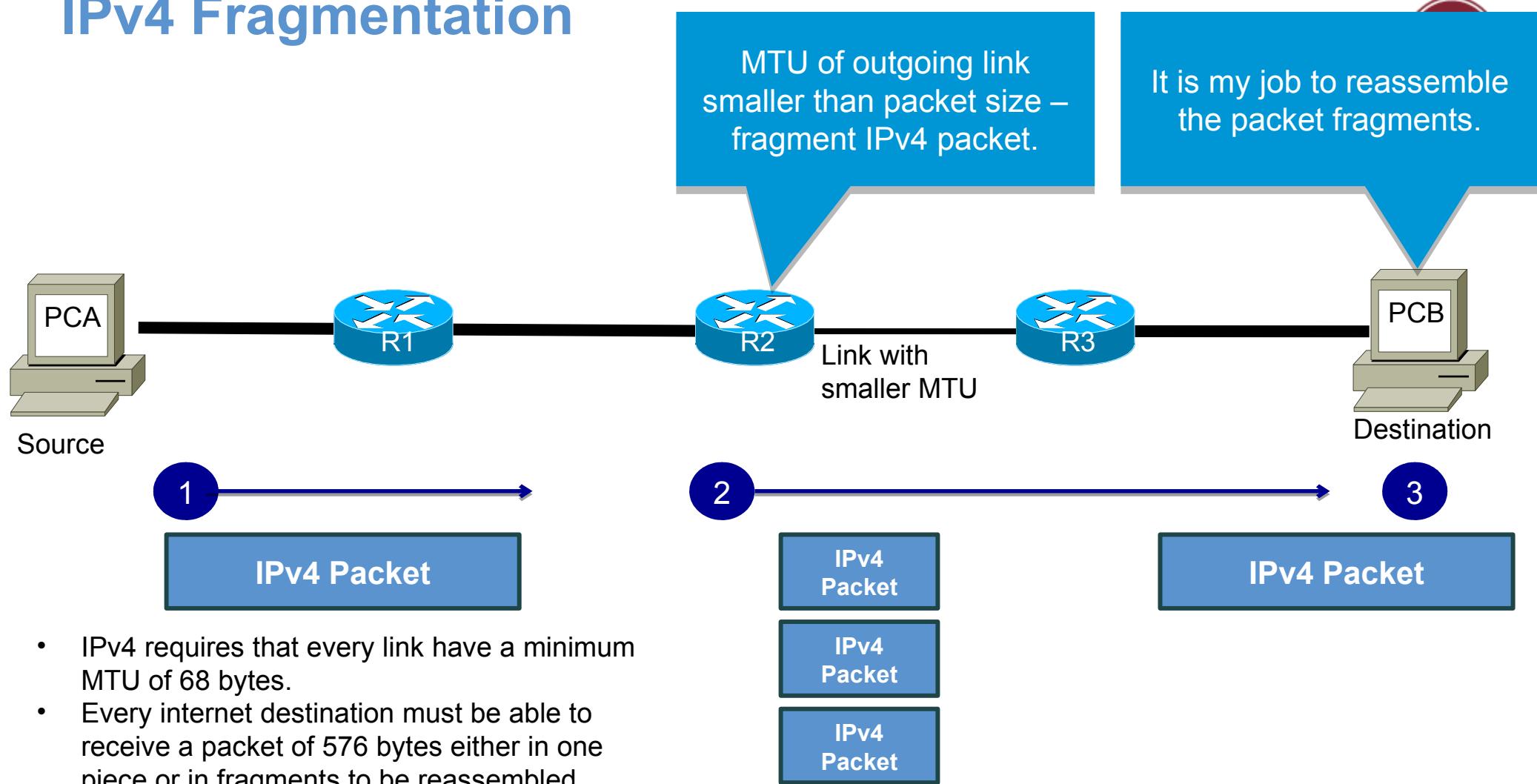
MTU of outgoing link
smaller than packet size –
fragment IPv4 packet.



IPv4 Fragmentation



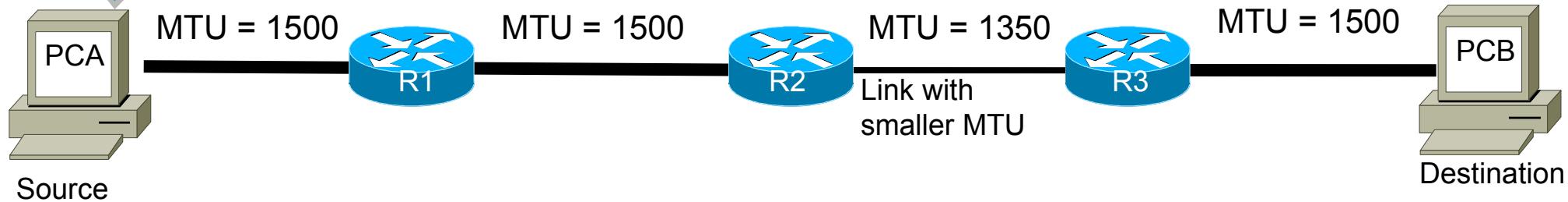
IPv4 Fragmentation



IPv6 No Fragmentation



I will use MTU of the interface.

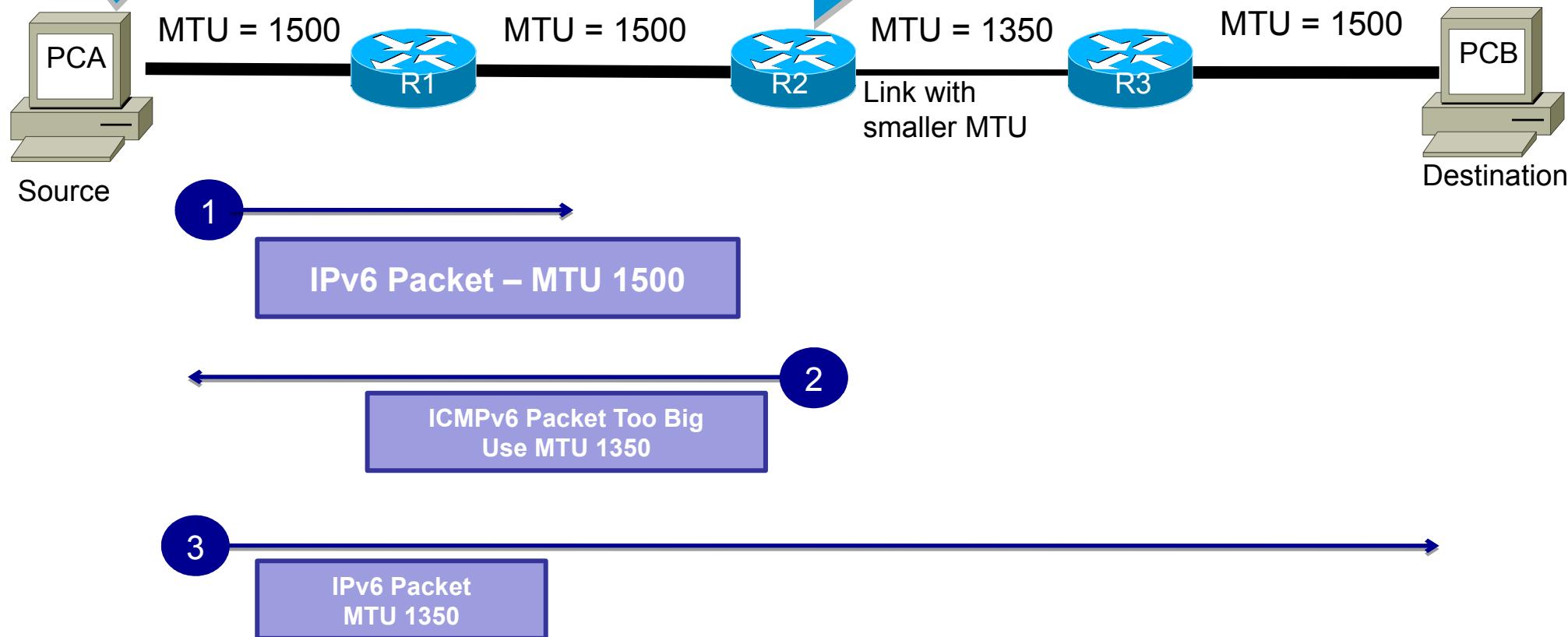


IPv6 No Fragmentation



I will use MTU of the interface.

MTU of outgoing link smaller than packet size. Drop packet. Send ICMPv6 Packet Too Big message, use MTU 1350.

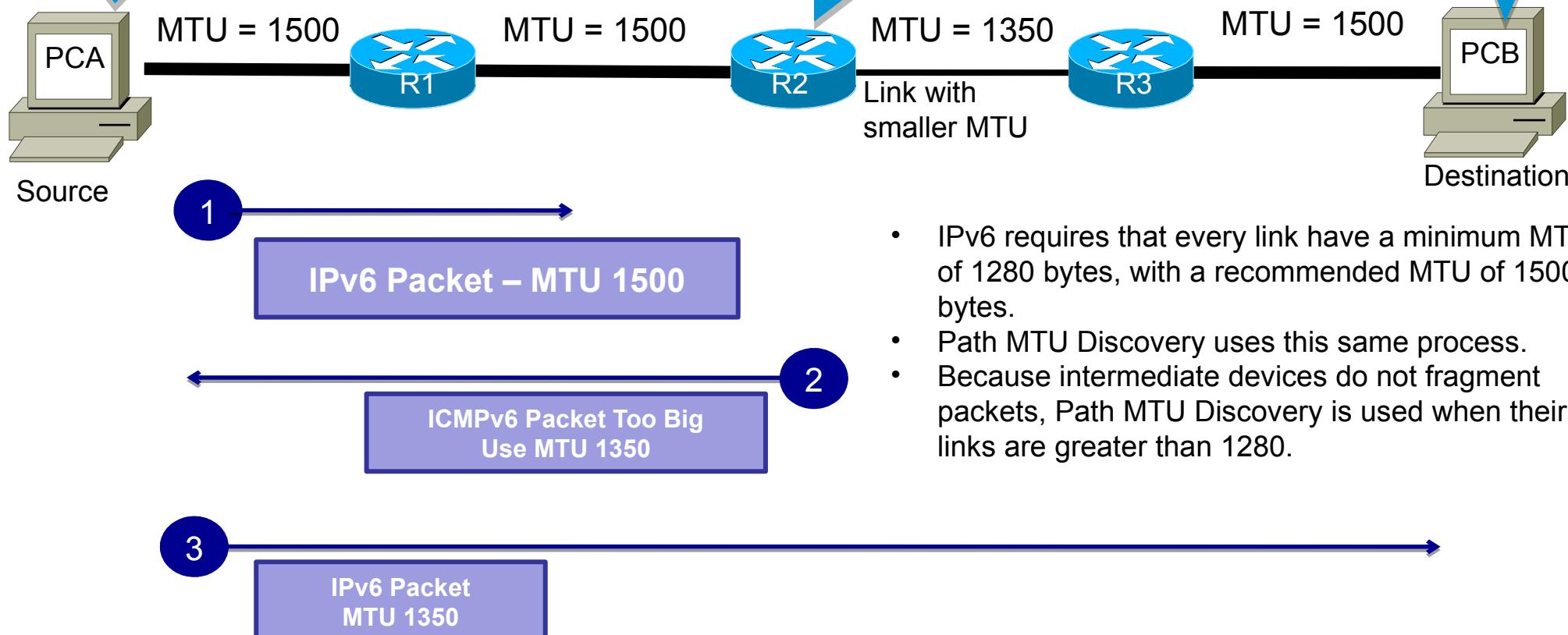


IPv6 No Fragmentation

I will use MTU of the interface.

MTU of outgoing link smaller than packet size. Drop packet. Send ICMPv6 Packet Too Big message, use MTU 1350.

Packet received. No reassembly required.



- IPv6 requires that every link have a minimum MTU of 1280 bytes, with a recommended MTU of 1500 bytes.
- Path MTU Discovery uses this same process.
- Because intermediate devices do not fragment packets, Path MTU Discovery is used when their links are greater than 1280.

IPv6 Next Header



- **IPv4 Protocol**
 - **IPv6 Next Header**
 - For both protocols, the field indicates the type of header following the IP header.

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		

IPv6

IPv6 Next Header



- **IPv4 Protocol**
 - **IPv6 Next Header**
 - For both protocols, the field indicates the type of header following the IP header.

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									

IPv6

The diagram illustrates the structure of an IPv6 header. At the top, a horizontal scale shows byte positions from 0 to 64. Below this, the header fields are arranged in a grid:

	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																
IPv6 Header								Next Header								
Destination Address																

A red box highlights the "Next Header" field at position 44, which is also labeled as the "Type of Service" field. The "Source Address" and "Destination Address" fields are shown as large blue boxes spanning multiple bytes.

IPv6 Next Header



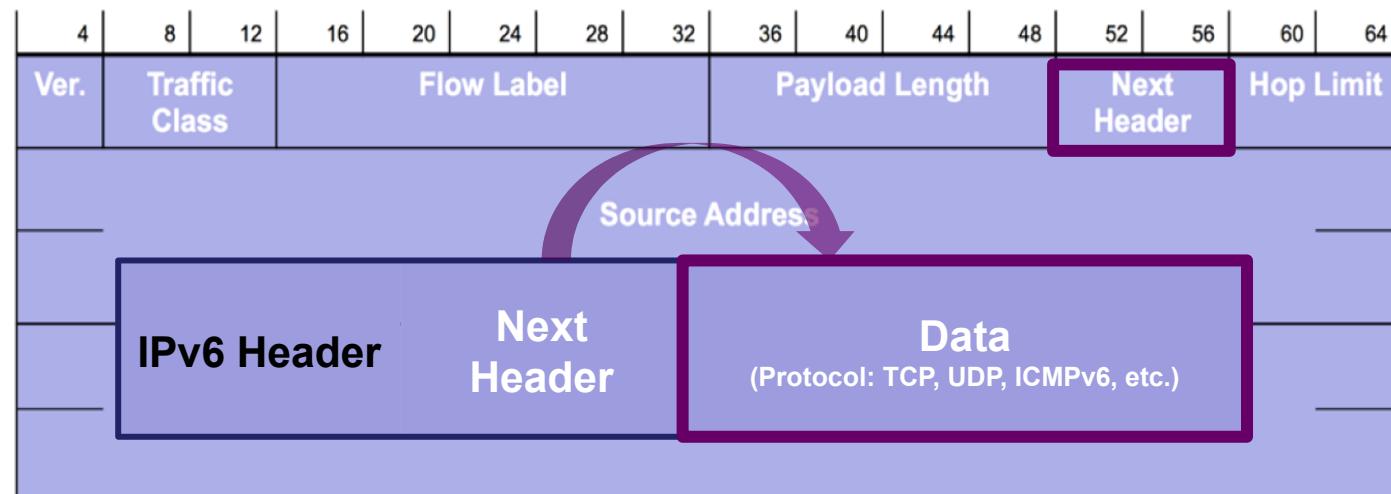
- **IPv4 Protocol**
- **IPv6 Next Header**
- For both protocols, the field indicates the type of header following the IP header.

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			

- Common values:
- 6 = TCP
- 17 = UDP
- 58 = ICMPv6
- 88 = EIGRP
- 89 = OSPF

IPv6



IPv6 Hop Limit



- **IPv4 TTL (Time to Live)**
- **IPv6 Hop Limit**
- Renamed to more accurately reflect process.
- Set by source, every router in path decrements hop limit by 1.
- When 0, drop packet.

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

IPv6

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 Source and Destination Addresses



- **IPv6 Source and Destination** addresses have the same basic functionality as IPv4.
- IPv4 – 32-bit addresses.
- IPv6 – 128-bit addresses.
- Some significant changes in IPv6.

IPv4



IPv6



IPv4 Header Checksum



- **IPv4 Header Checksum**
 - Not used in IPv6.
 - Upper-layer protocols generally have a checksum (UDP and TCP).
 - So, in IPv4 the UDP checksum is optional.
 - Because it's not in IPv6, the UDP checksum is now mandatory.

IPv6

	4	8	12	16	20	24
Ver.	Traffic Class	Flow Label				

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							

IPv6

IPv4 Options and Padding



- **IPv4 Options and Padding**
 - Not used in IPv6.
 - Variable length, optional.
 - **IPv4 Options** are handled using extension headers in IPv6.

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										

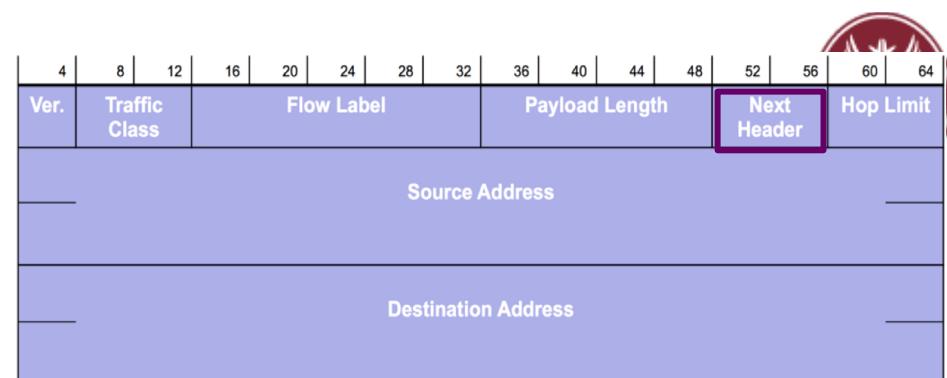
- Padding makes sure IPv4 options fall on a 32-bit boundary.
 - IPv6 header is fixed at 40 bytes.

IPv6

40 bytes =

IPv6 Extension Header

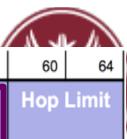
- **Next Header** identifies:
 - The protocol carried in the data portion of the packet.
 - The presence of an extension header.



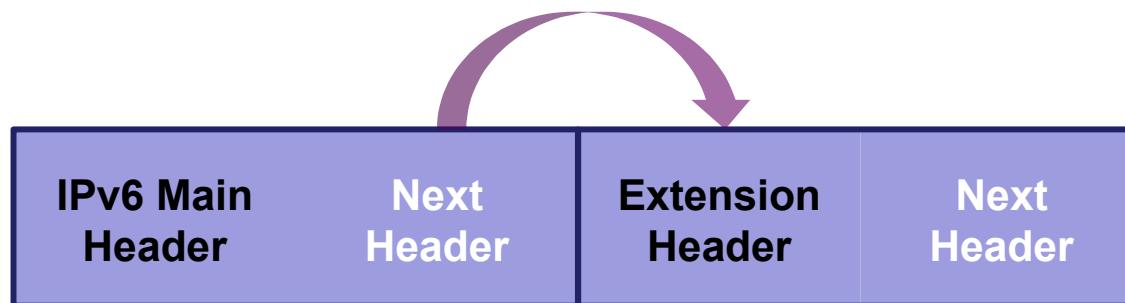
IPv6 Main
Header

Next
Header

IPv6 Extension Header

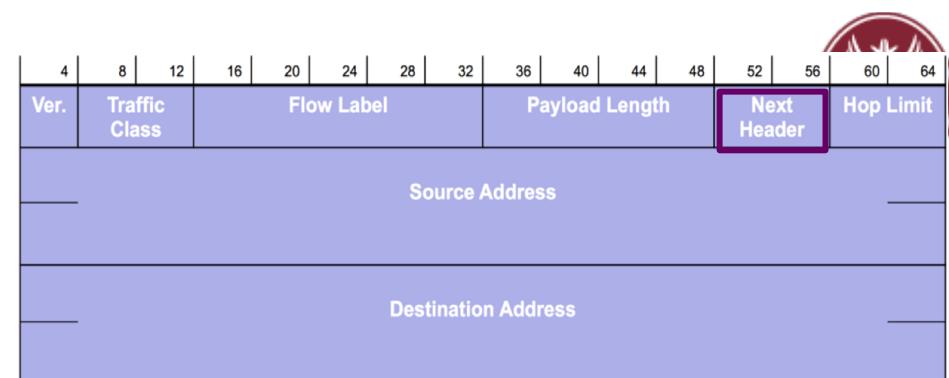


- **Next Header** identifies:
 - The protocol carried in the data portion of the packet.
 - The presence of an extension header.



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.



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 Extension Header Properties



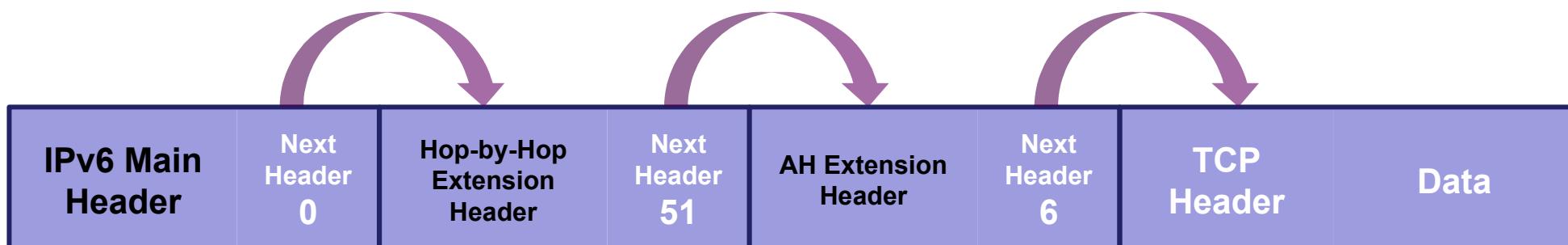
- **Flexible** (normally there are no EHs in IPv6 packets)
 - The use of EHs is optional, providing a powerful and flexible mechanism for IPv6
 - In the Basic IPv6 header, the EHs and the upper layer header (if used), are linked using the Next Header field. This is called the “**IPv6 Header Chain**”
- **Fixed** (Types and order)
 - The number of Extension Header types is fixed and standardised.
 - **Processed only at endpoints** (Except Hop-by-Hop and Routing)
 - Packet processing complexity moved from the core to the edge of the Internet for improved IPv6 performances.



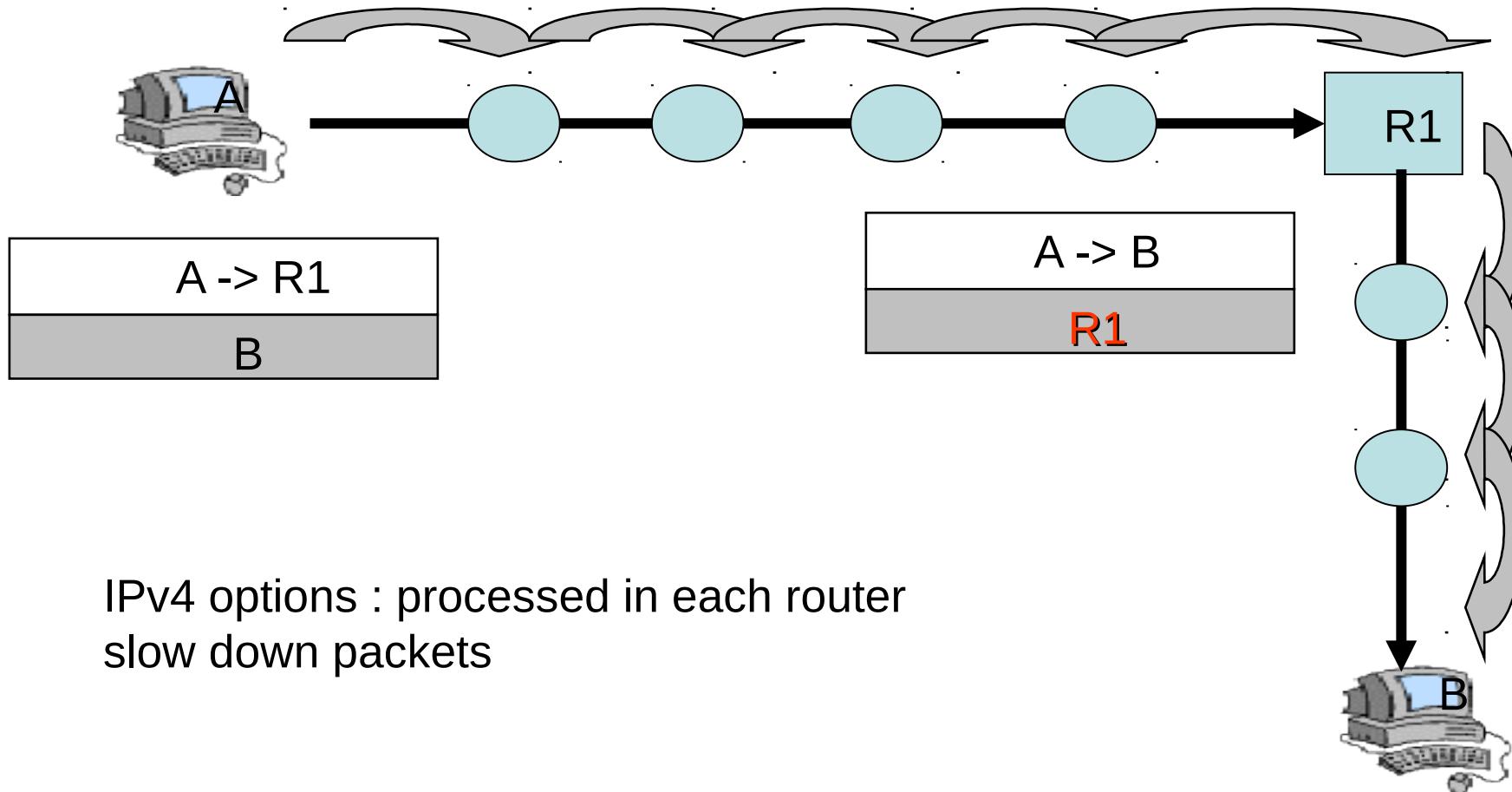


IPv6 Extension Header

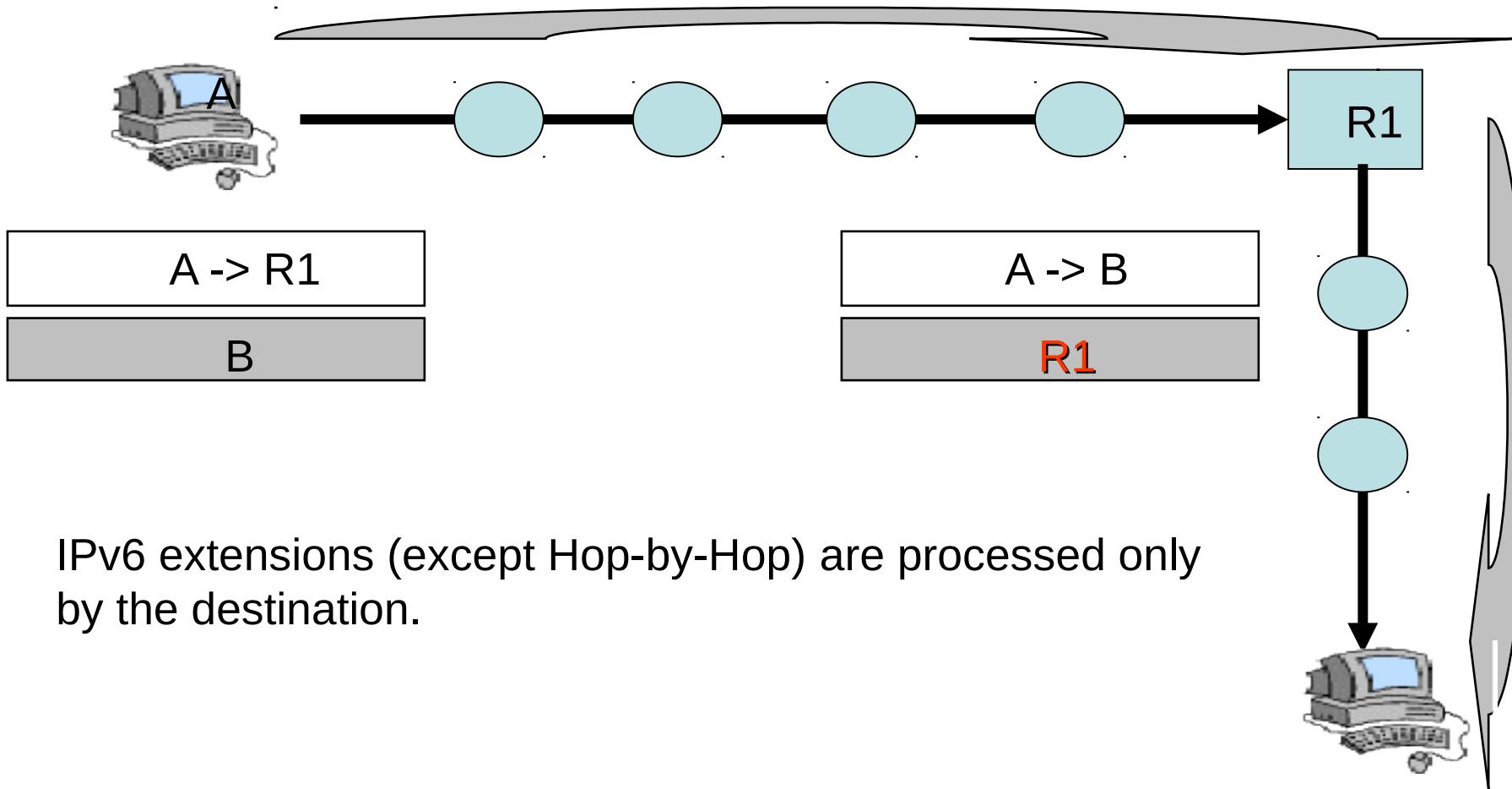
Next Header Value (Decimal)	Extension Header Name	Extension Header Description
0	Hop-by-Hop Options	Used to carry optional information, which must be examined by every router along the path of the packet.
43	Routing	Allows the source of the packet to specify the path to the destination.
44	Fragment	Used 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).



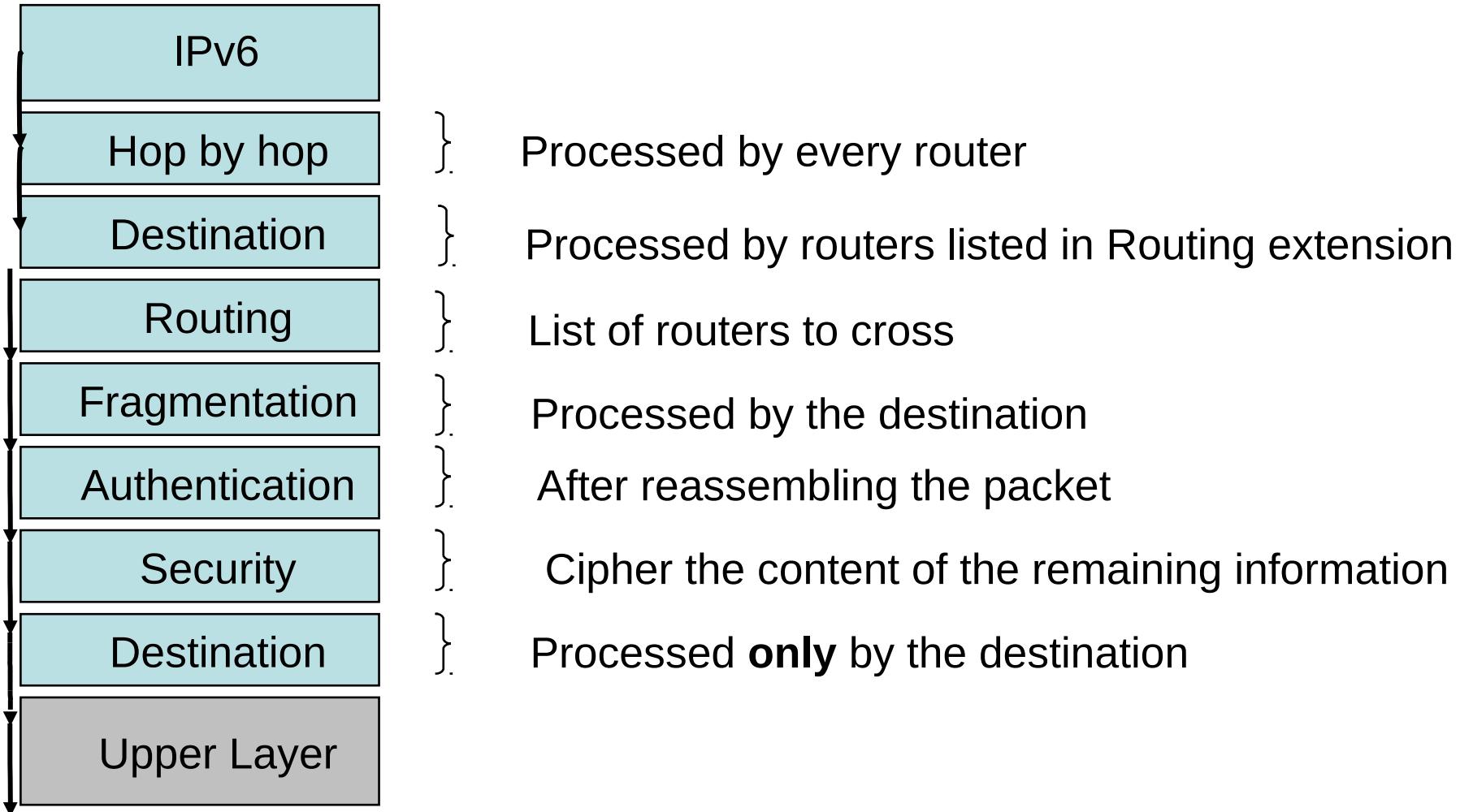
IPv4 options vs. IPv6 extensions



IPv4 options vs. IPv6 extensions



Order is important (RFC 8600 Section 4)





That's all for today

- **Questions?**
- See you on Thursday!!
- References:
 - 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)

Practical Network Defense

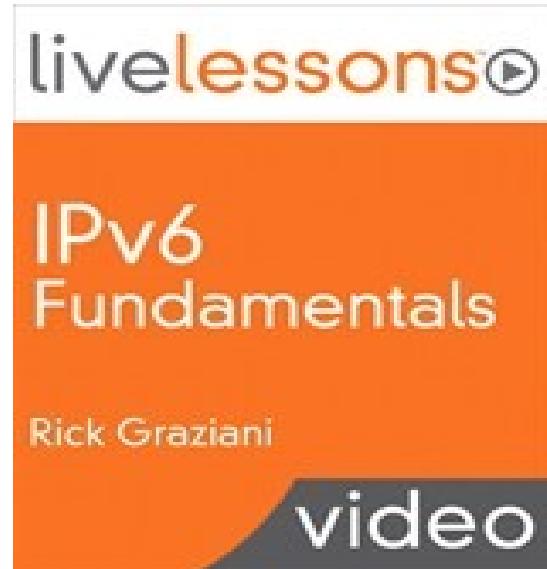
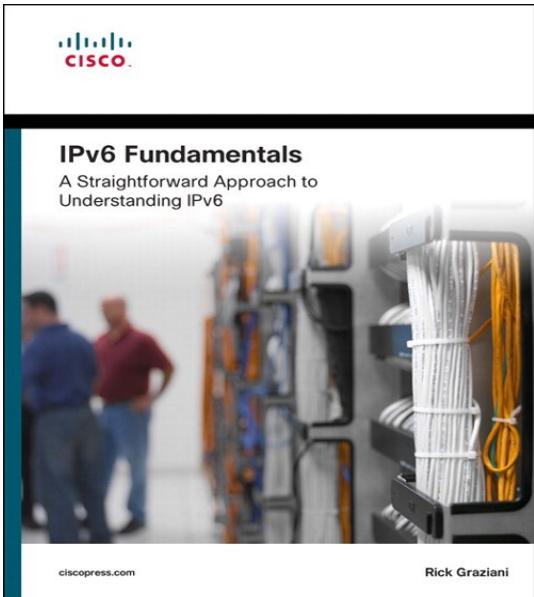
Master's degree in Cybersecurity 2021-22

IPv6: addressing and ICMPv6 lab

Angelo Spognardi
[*spognardi@di.uniroma1.it*](mailto: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



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

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

IPv6

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																									

64-bit memory word

IPv4

4	8	12	16	20	24	28	32	36	40	44	48	52	56	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																	

Similar fields

IPv6 Extension Header

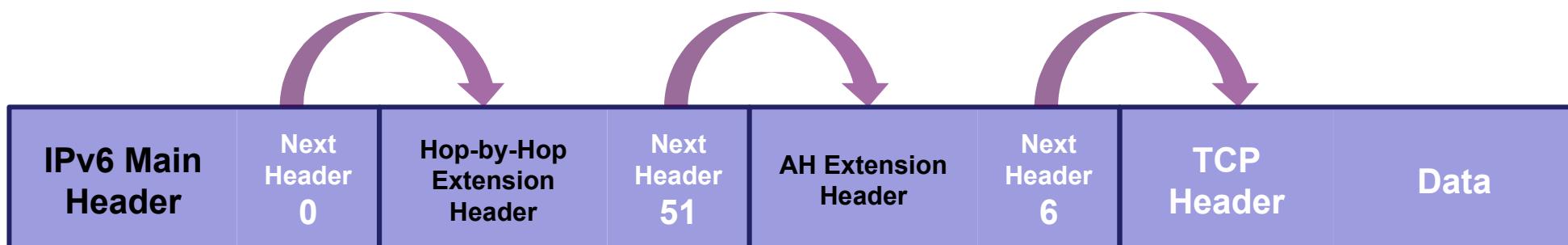
- **Next Header** identifies:
 - The protocol carried in the data portion of the packet.
 - The presence of an extension header.
- **Extension headers** are optional and follow the main IPv6 header.
- Provide flexibility and features to the main IPv6 header for future enhancements without having to redesign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



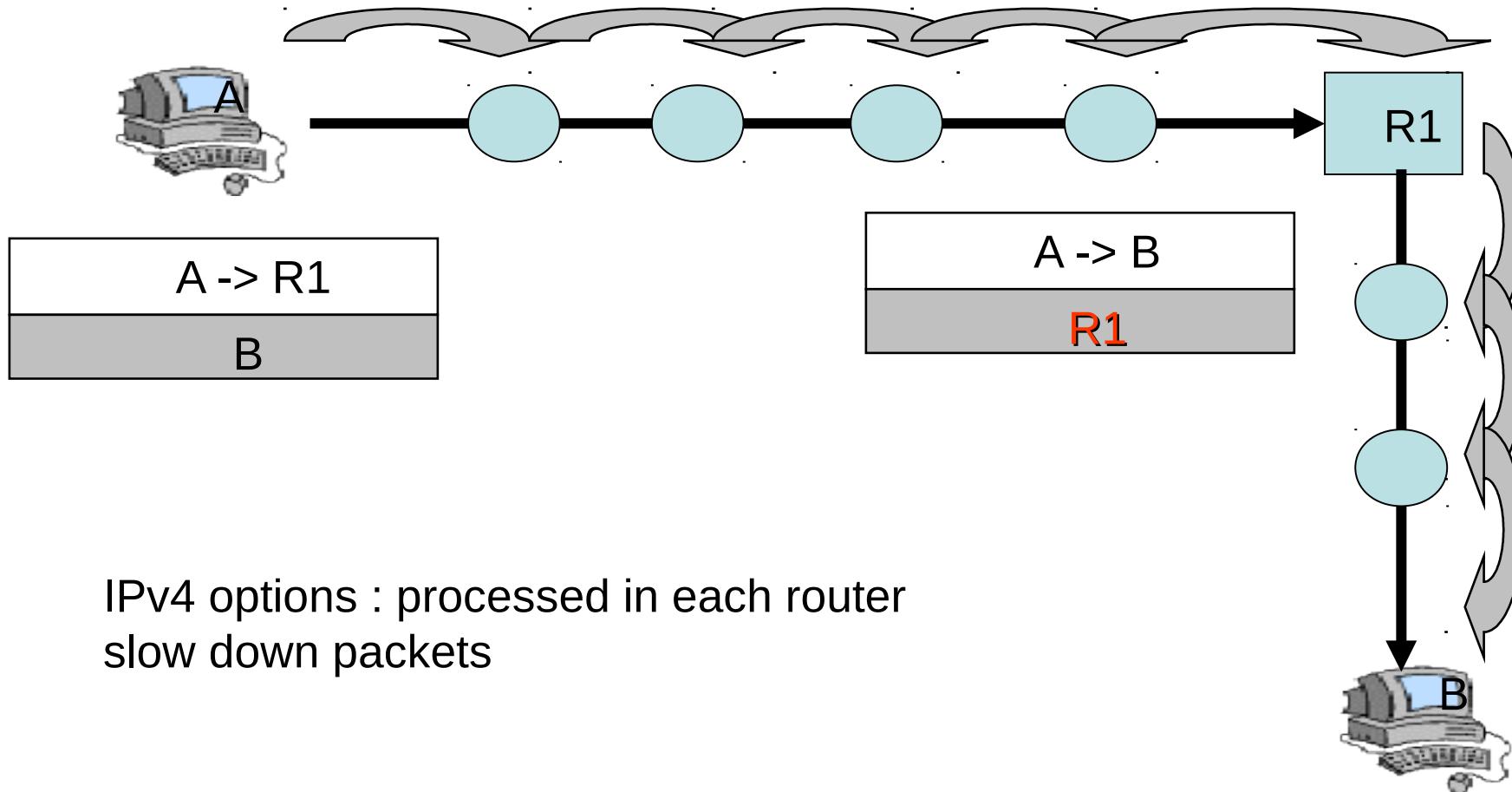


IPv6 Extension Header

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).

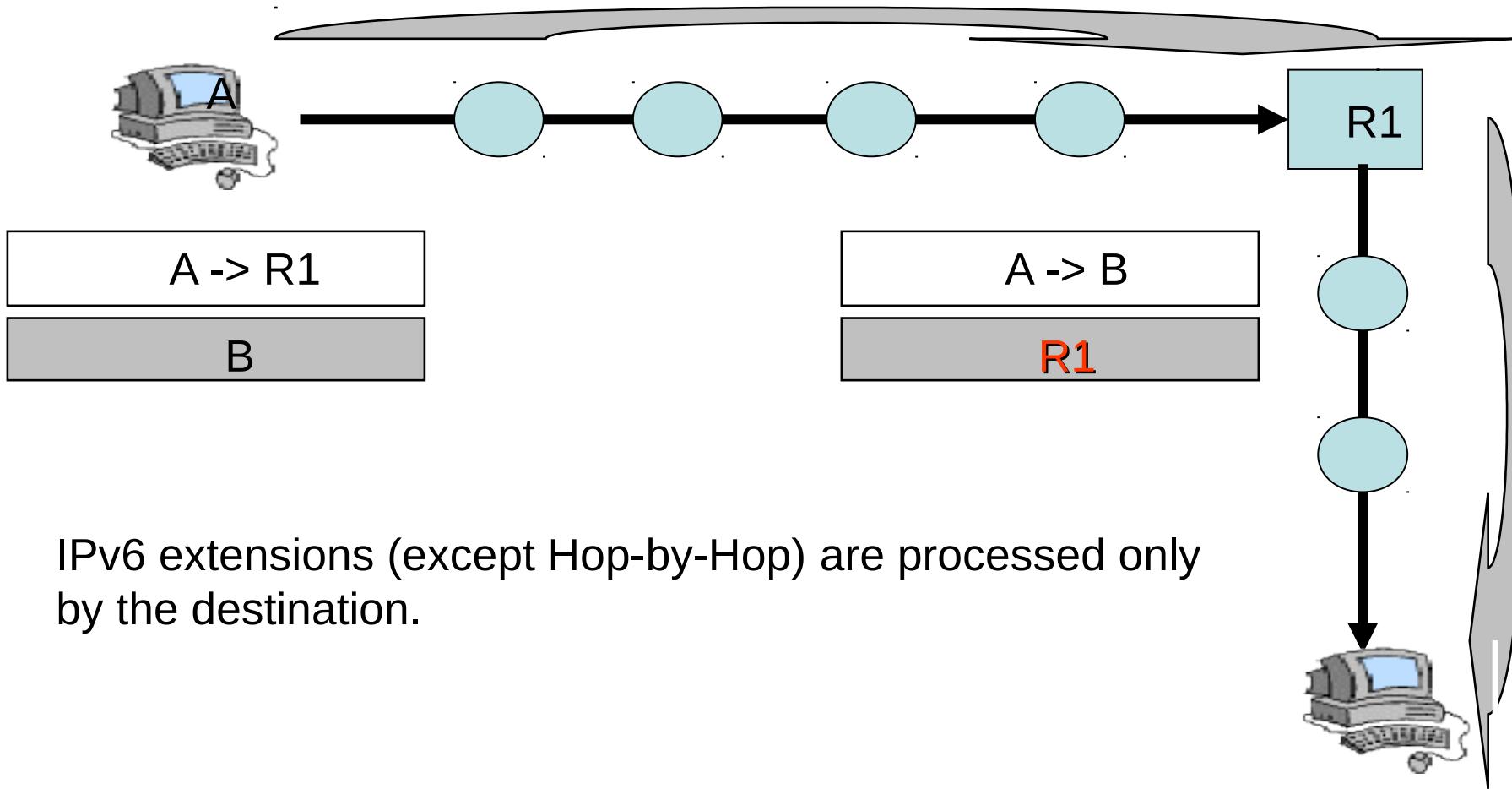


IPv4 options vs. IPv6 extensions

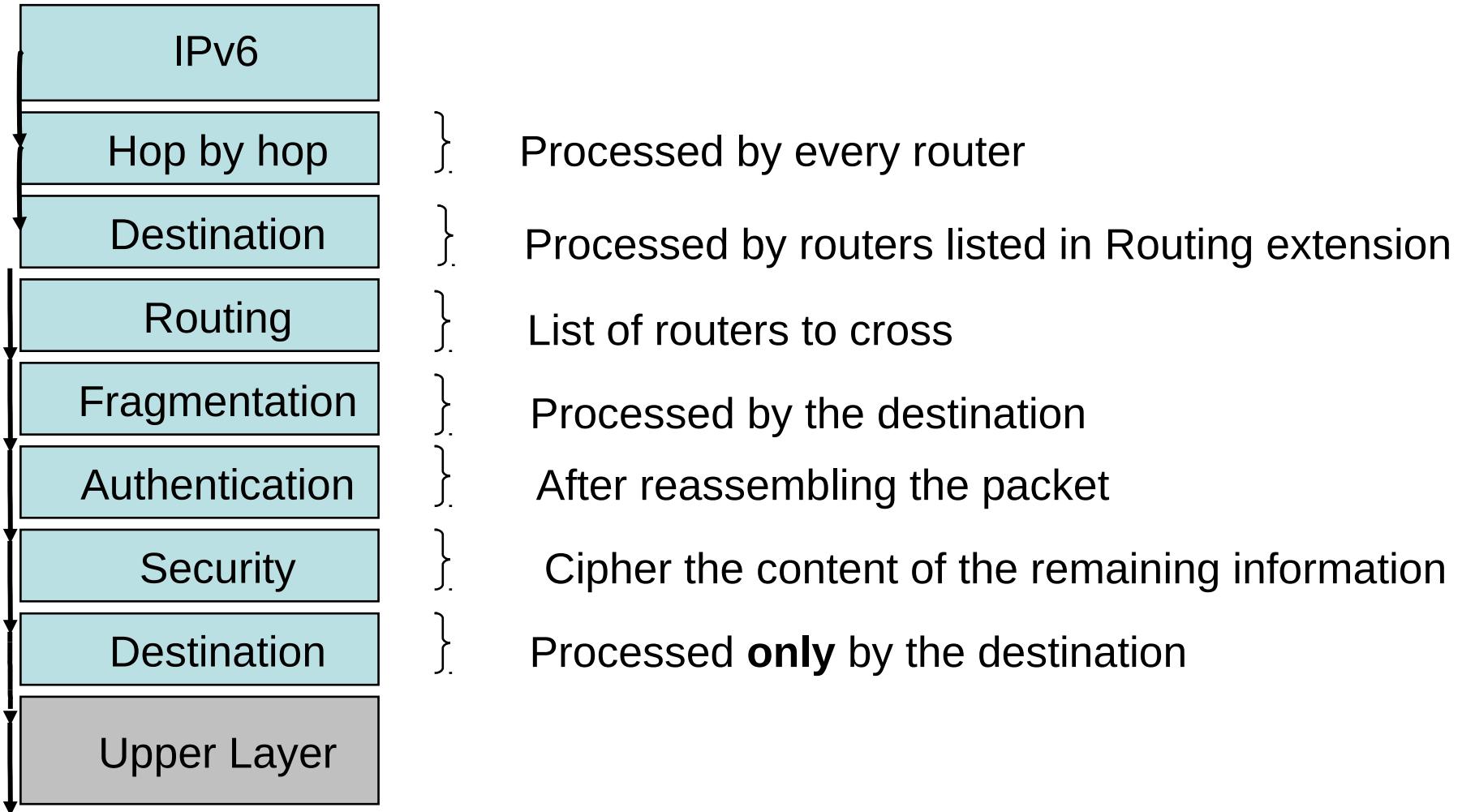


IPv4 options : processed in each router
slow down packets

IPv4 options vs. IPv6 extensions



Order is important (RFC 2460)





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>

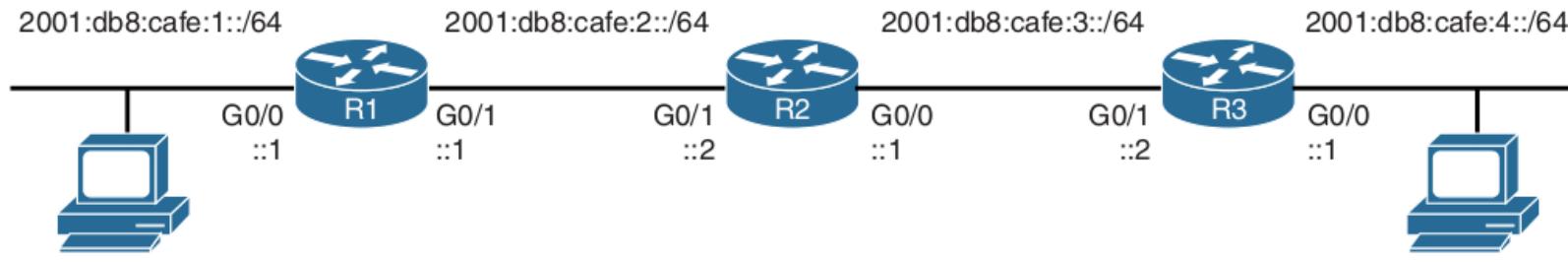


Lab activity: ex5



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 end points of a link to alter the MTU





Lab activity: ex6



Exercise 6: create an IPv6 capable connection

- The task is to create a 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)

Practical Network Defense

Master's degree in Cybersecurity 2021-22

Network traffic regulation with firewalls

Angelo Spognardi
[*spognardi@di.uniroma1.it*](mailto:spognardi@di.uniroma1.it)
Dipartimento di Informatica
Sapienza Università di Roma



Today's agenda

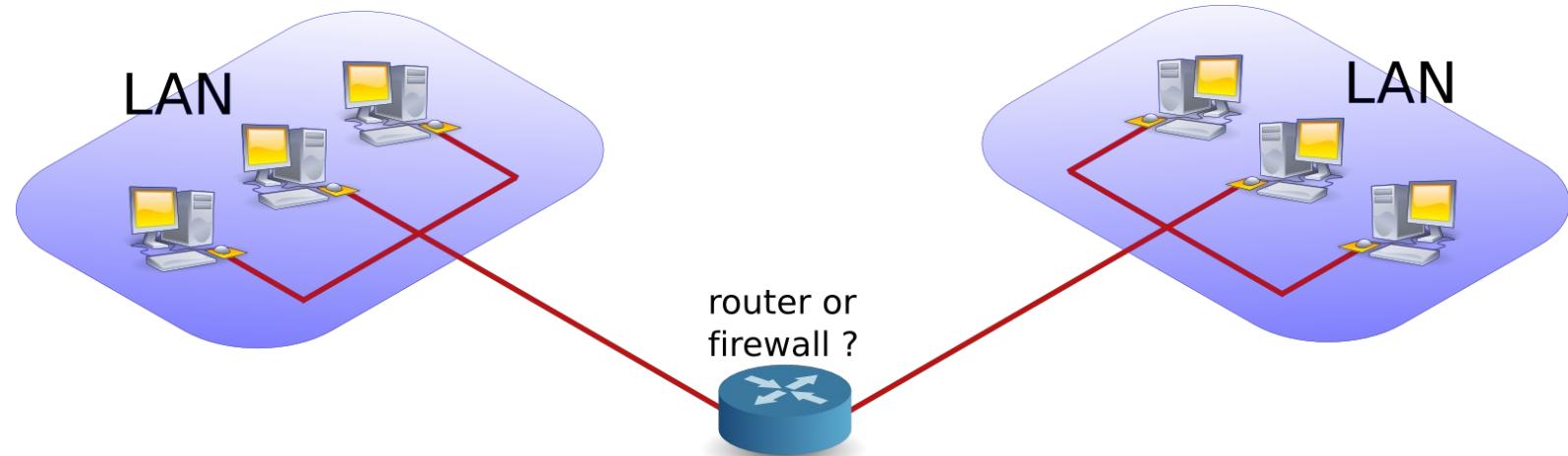
- Traffic regulation
 - Packet filtering
- Filter rules
- Stateful firewall
- Other types of firewall
 - Application-level filtering
 - Circuit-level gateway



Traffic regulation

Regulate traffic: routers and firewalls

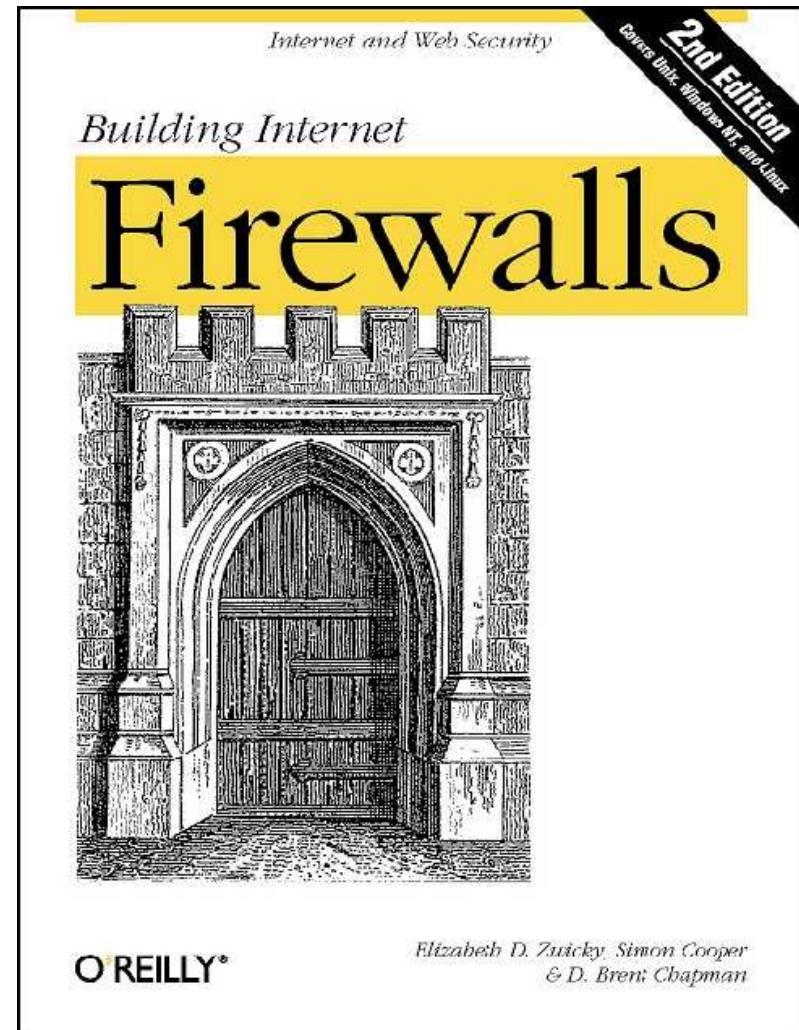
- A router is a device that connects two networks
- A firewall is a device that besides acting as a router, also contains (and implements) rules to determine whether packets are allowed to travel from one network to another
 - Router also can perform some form of screening (packet filter)





Why firewalls?

- Restricts access from the outside
 - Internet = millions of people together → bad things happen
- Prevents attackers from getting too close
- Restricts people from leaving





Secure traffic regulation

- To attain a certain level of network security, you can:
 - Regulate which traffic is allowed (sources, destinations, services, ...)
 - Protect the traffic by encryption
 - Monitor the traffic for “bad behaviour”
 - Monitor the hosts for “bad behaviour”
- The choice will depend on the security policy to be fulfilled (particularly the CIA targets).



Firewall Design & Architecture Issues

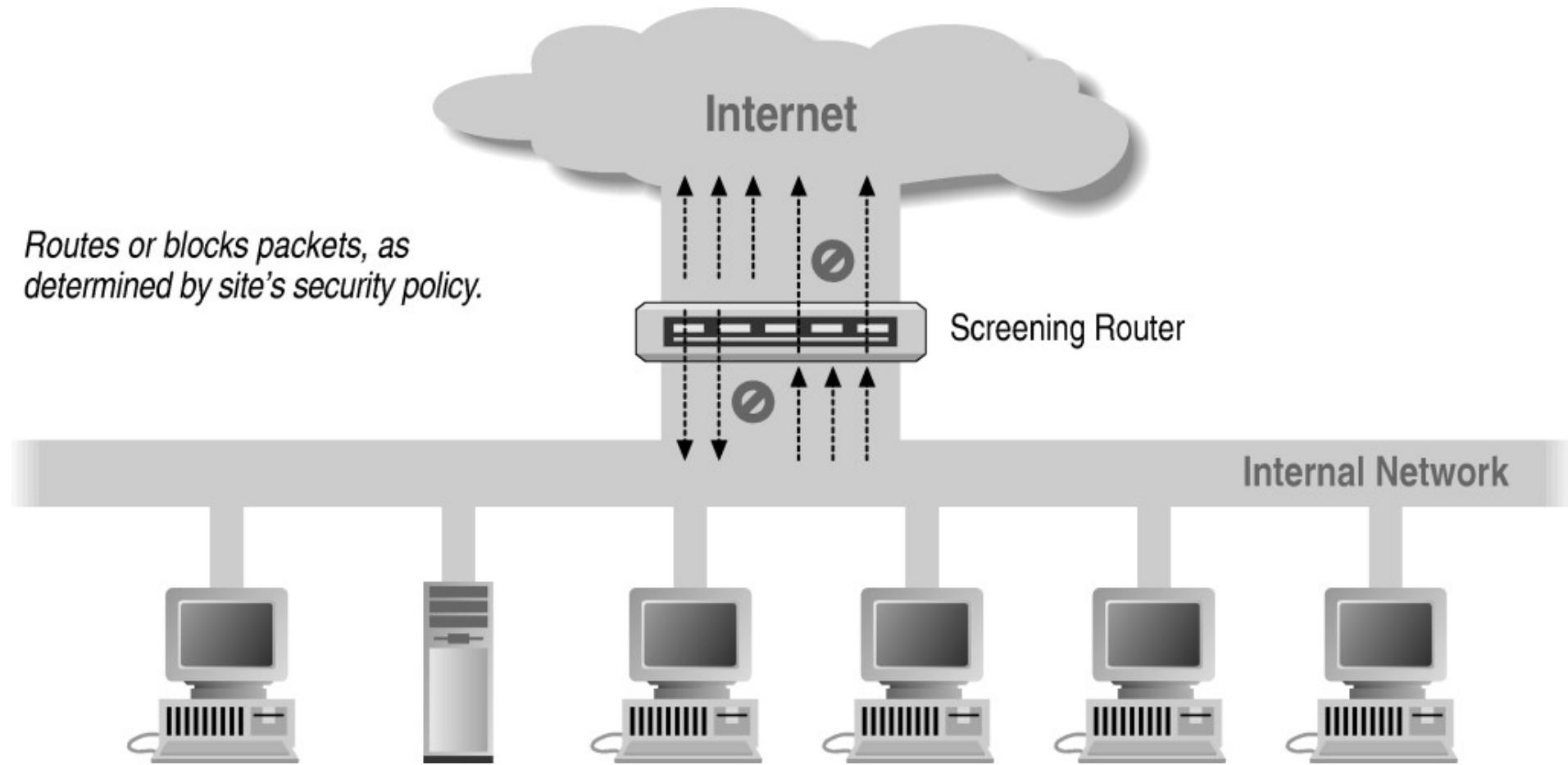
- Least privilege
- Defense in depth
- Choke point
- Weakest links
- Fail-safe stance
- Universal participation
- Diversity of defense
- Simplicity



Host based packet filters

- Kind of firewall that disciplines the traffic in/out a single host
- It specifies the packets that can be received and sent
 - Ex: iptables, windows firewall and all the so called “personal firewalls”
- Vendor products generally work per-app: each installed application has a known policy that has to obey

Screening router (ACL-based)

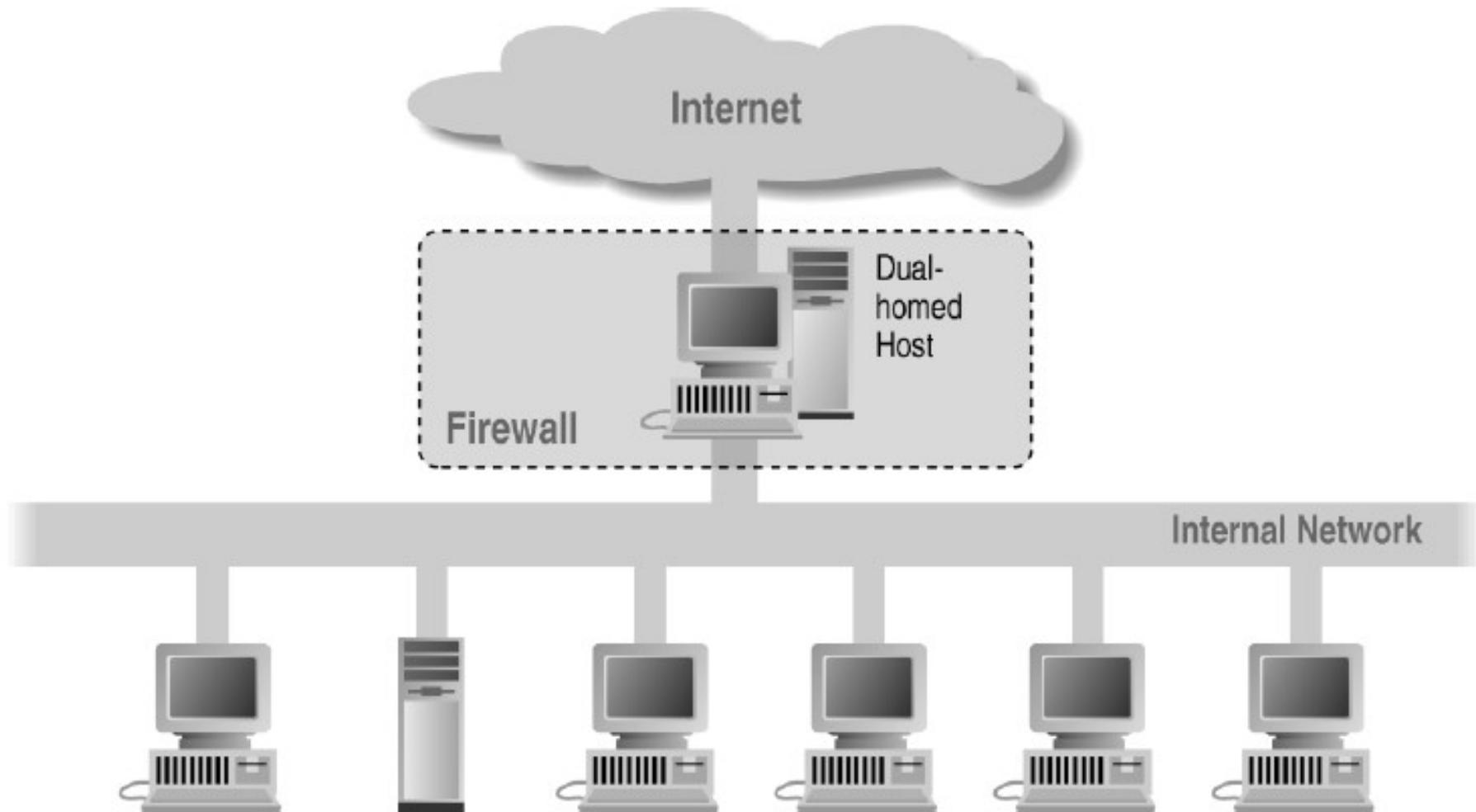




Network Access Control Lists

- List the rights for accessing/using networks
 - Extensively used in switches, routers and firewalls
- Usually distinguish between incoming and outgoing traffic, per interface/port
 - Ex: lists of IP addresses that can send packets to an interface/port
- Stateless: every packet is treated independently, without any knowledge of what has come before

Dual-homed host





Bastion host

- Hardened computer used to deal with all traffic coming to a protected network from outside
 - Hardening is the task of reducing or removing vulnerabilities in a computer system:
 - Shutting down unused or dangerous services
 - Strengthening access controls on vital files
 - Removing unnecessary accounts and permissions
 - Using “stricter” configurations for vulnerable components, such as DNS, sendmail, FTP, Apache, Tomcat, etc.
- Specially suitable for use as Application Proxy Gateways



What is a DMZ

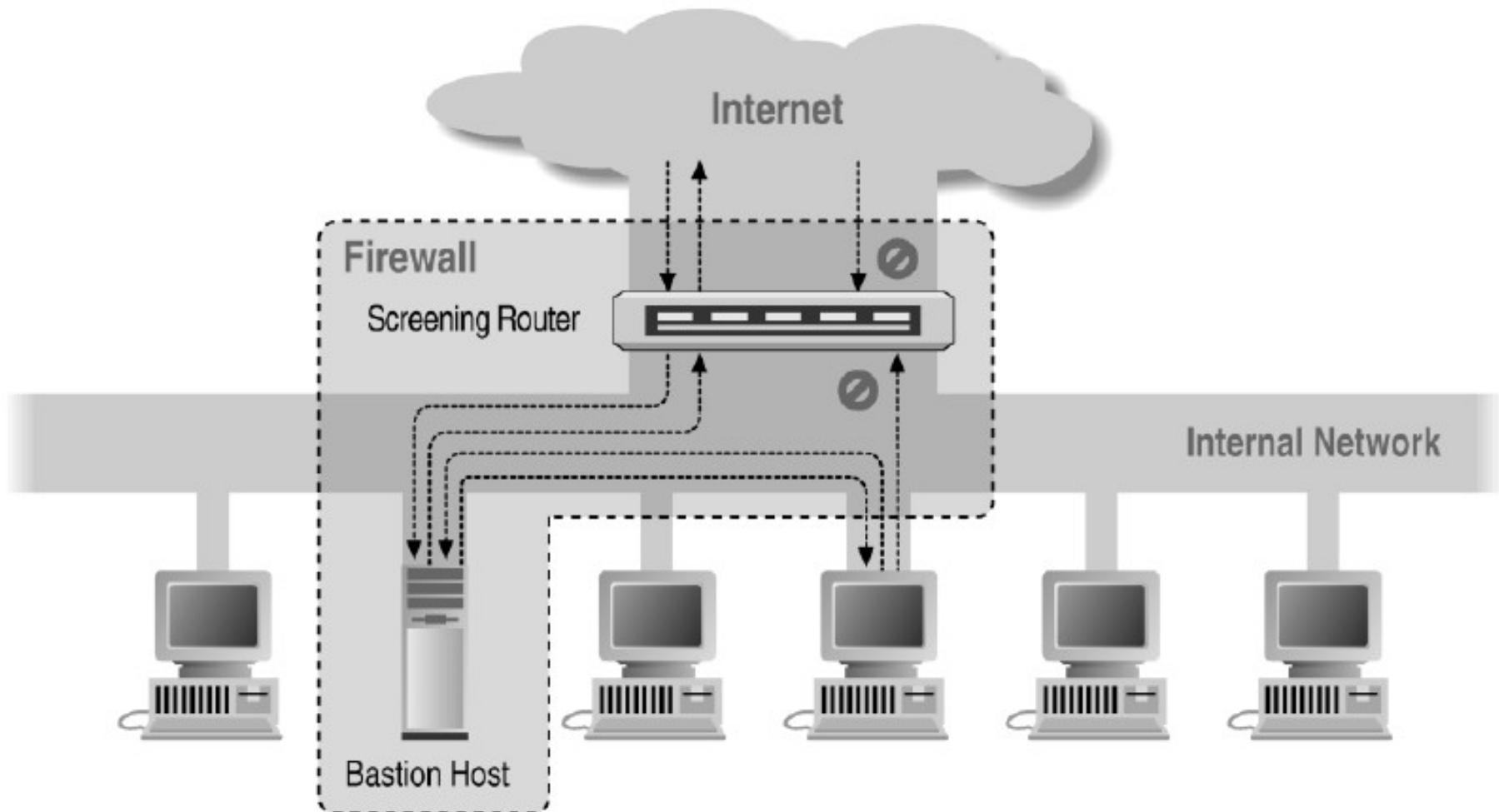
- DMZ (demilitarized zone)
 - Computer host or small network inserted as a “neutral zone” between a company’s private network and the outside public network
 - Network construct that provides secure segregation of networks that host services for users, visitors, or partners
- DMZ use has become a necessary method of providing a multilayered, **defense-in-depth** approach to security
- Reduce and regulate the access to internal (private) components of the IT system



Defense in depth

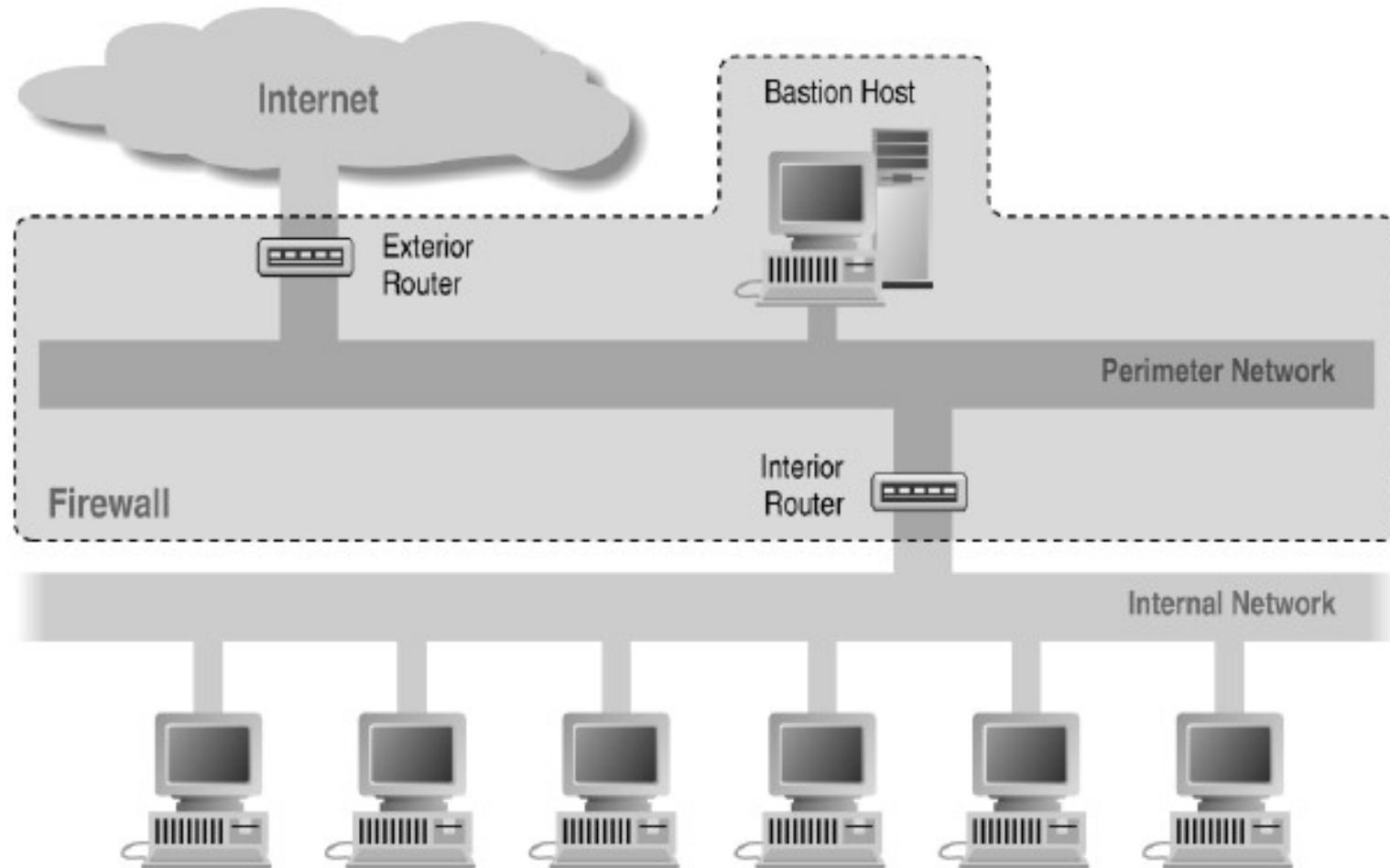
- A security approach in which IT systems are protected using **multiple overlapping** systems
 - Add redundancy to the defensive measures
 - Aim to remove the single point of failure
 - Find the right balance between complexity and multiplicity of defense measures
- In order to compromise the system, an attacker has to find multiple vulnerabilities, in different components

DMZ as a screened Host

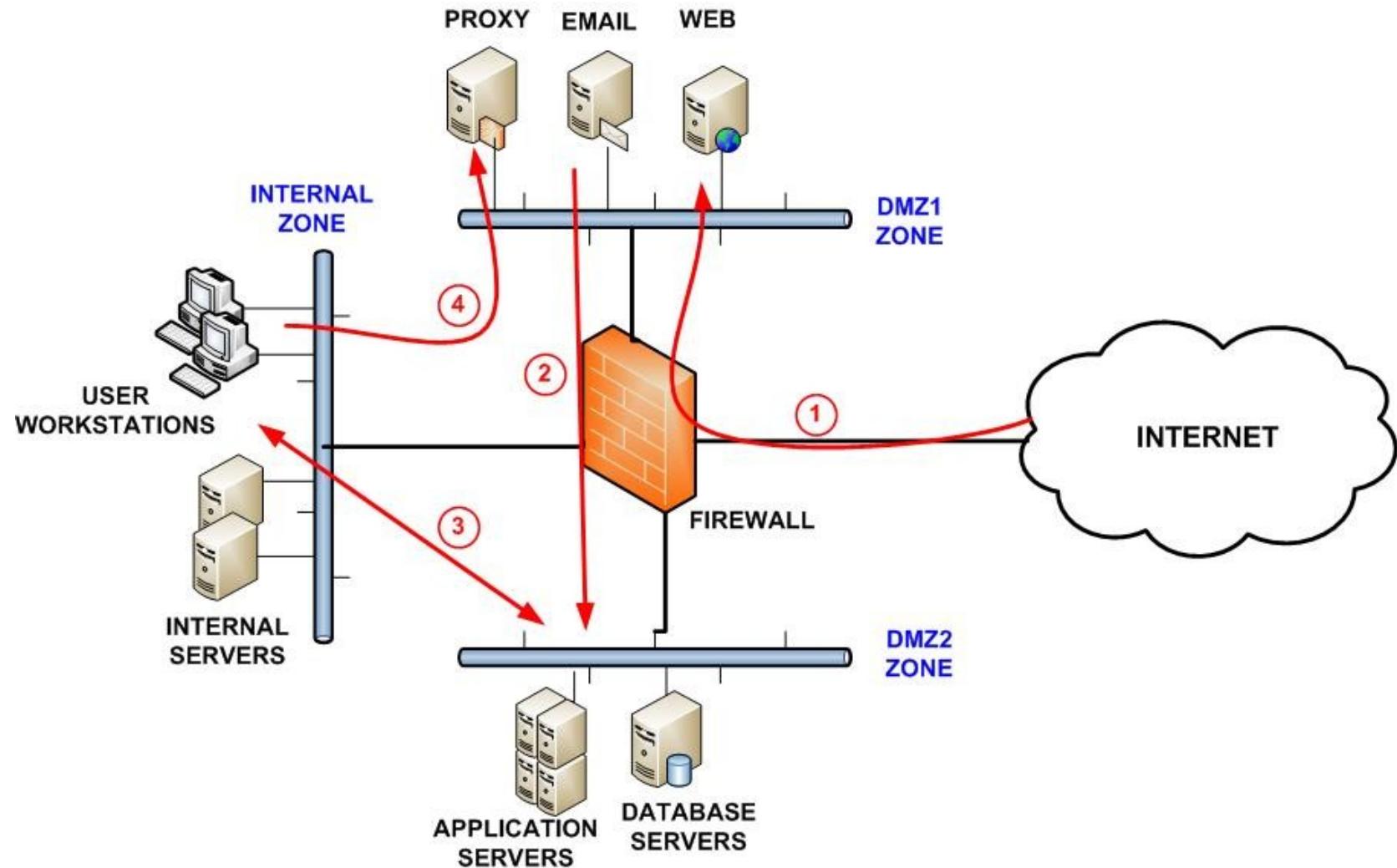




Screened Subnet Using Two Routers/Firewalls

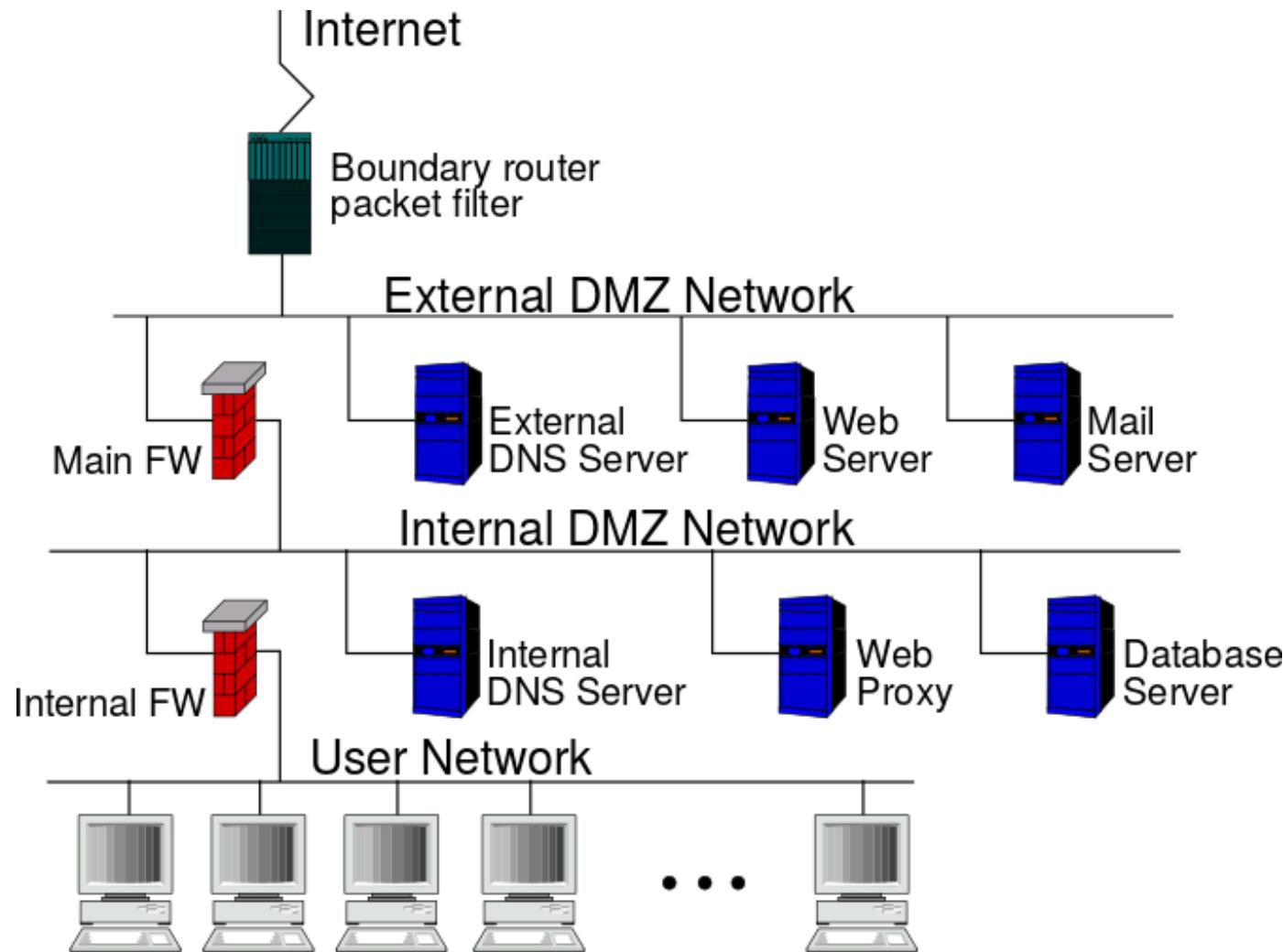


DMZ to segment the network



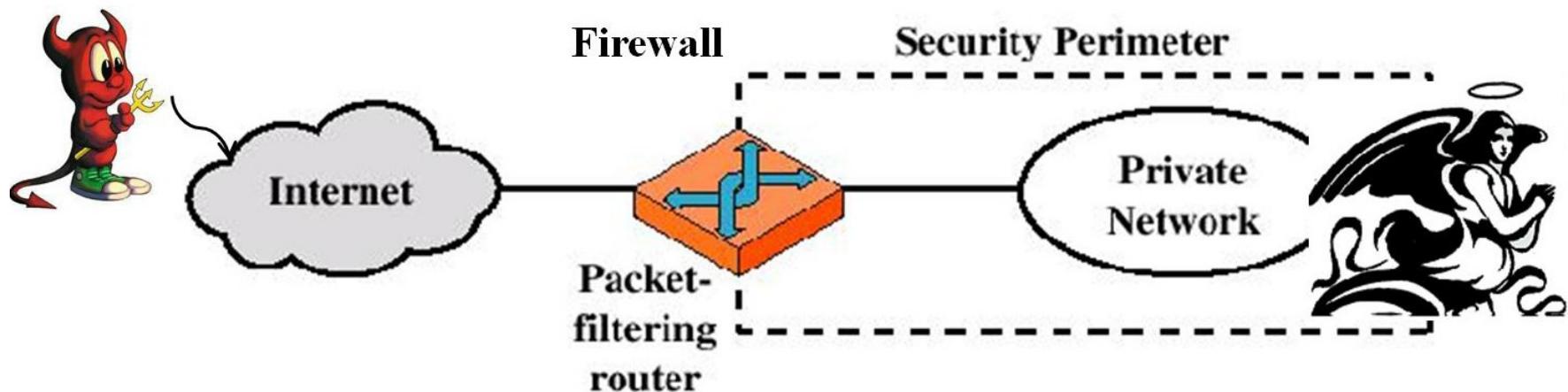


Security in depth: split DMZ



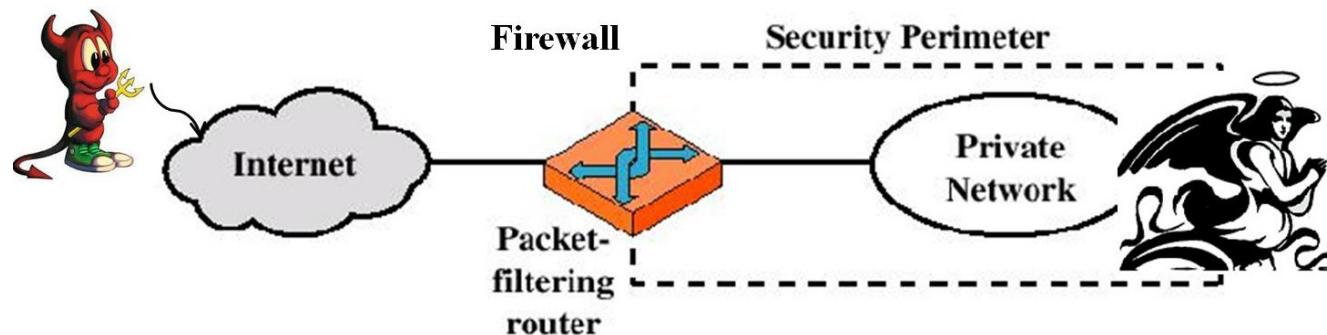
A simple plan for network security

- Use a firewall to filter ingoing and outgoing traffic between “your” network (or individual PC) and the Internet



Assumptions

1. You have security policy stating what is allowed and not allowed.
2. You can identify the “good” and the “bad” traffic by its IP-address, TCP port numbers, etc, ...
3. The firewall itself is immune to penetration.
 - A question of assurance – needs for a trusted system, secure OS etc.

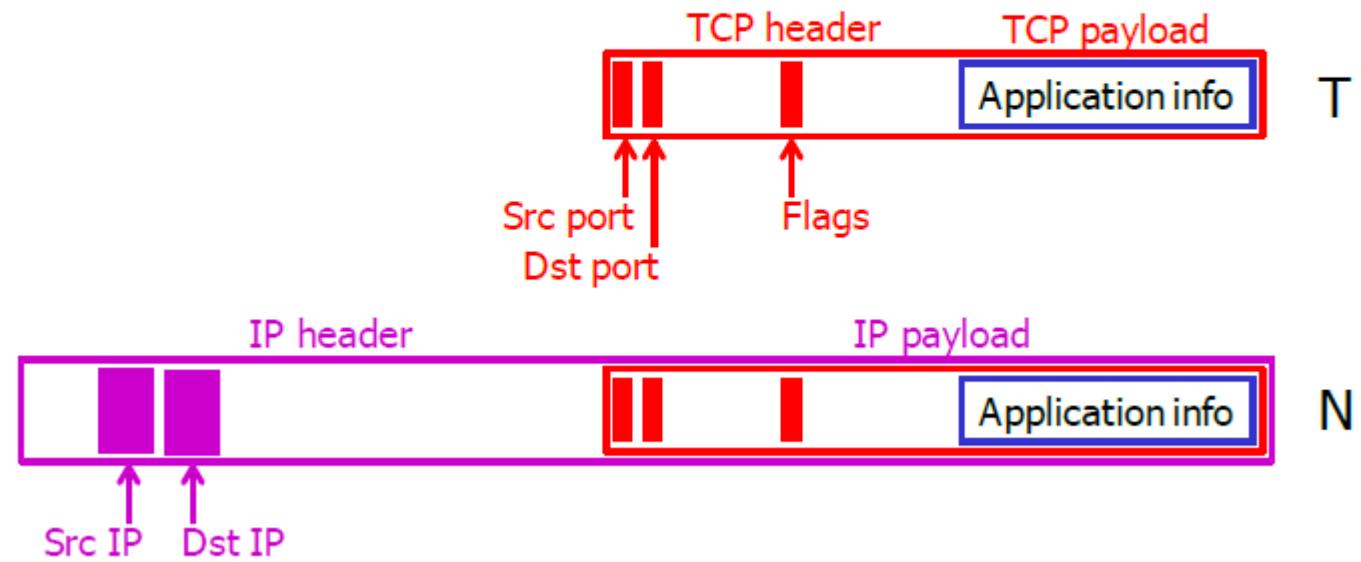
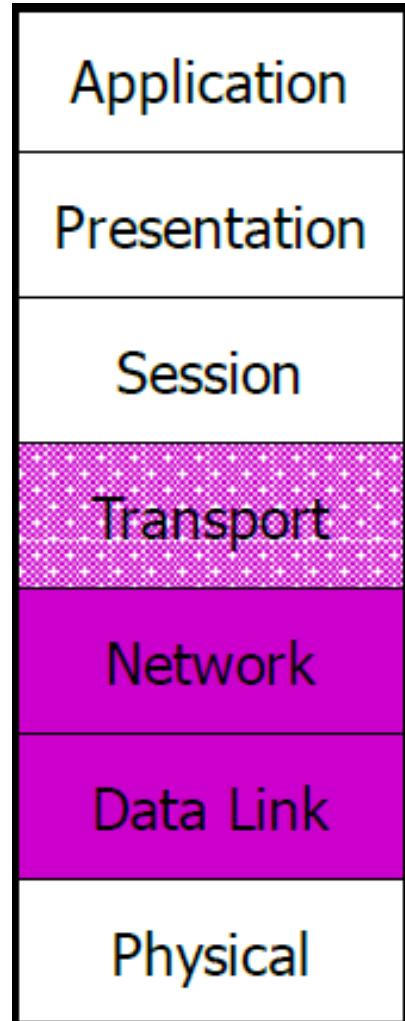


Packet filters (stateless firewall)

- Drop packets based on their source or destination addresses or port numbers or flags
- No context, only contents
- Can operate on
 - incoming interface
 - outgoing interface
 - both
- Check packets with fake IP addresses:
 - from outside (“ingress filtering”)
 - from inside (“egress filtering”)



Packet filters operating layers





Three-step process

1. Know your policy
2. Translate the policy in a formal language
 - E.g.: logical expression on packet fields
3. Rewrite the policy in terms of the firewall syntax

General mechanism:

- Rules are checked from top to bottom
- The first matching rule is applied
- One implicit rule is assumed if no rule matches
 - Block/Allow everything

action	ourhost	port	theirhost	port	comment
block	*	*	*	*	<i>default</i>



Example

- Policy:
 - allow inbound email (SMTP, port 25) only to our-gateway machine: **Mailgw**
 - refuse all traffic from a known spamming site: **demon**
- Possible rules:

action	ourhost	port	theirhost	port	comment
block	*	*	demon	*	<i>don't trust spammers</i>
allow	Mailgw	25	*	*	<i>connection to our SMTP</i>



Example, continued

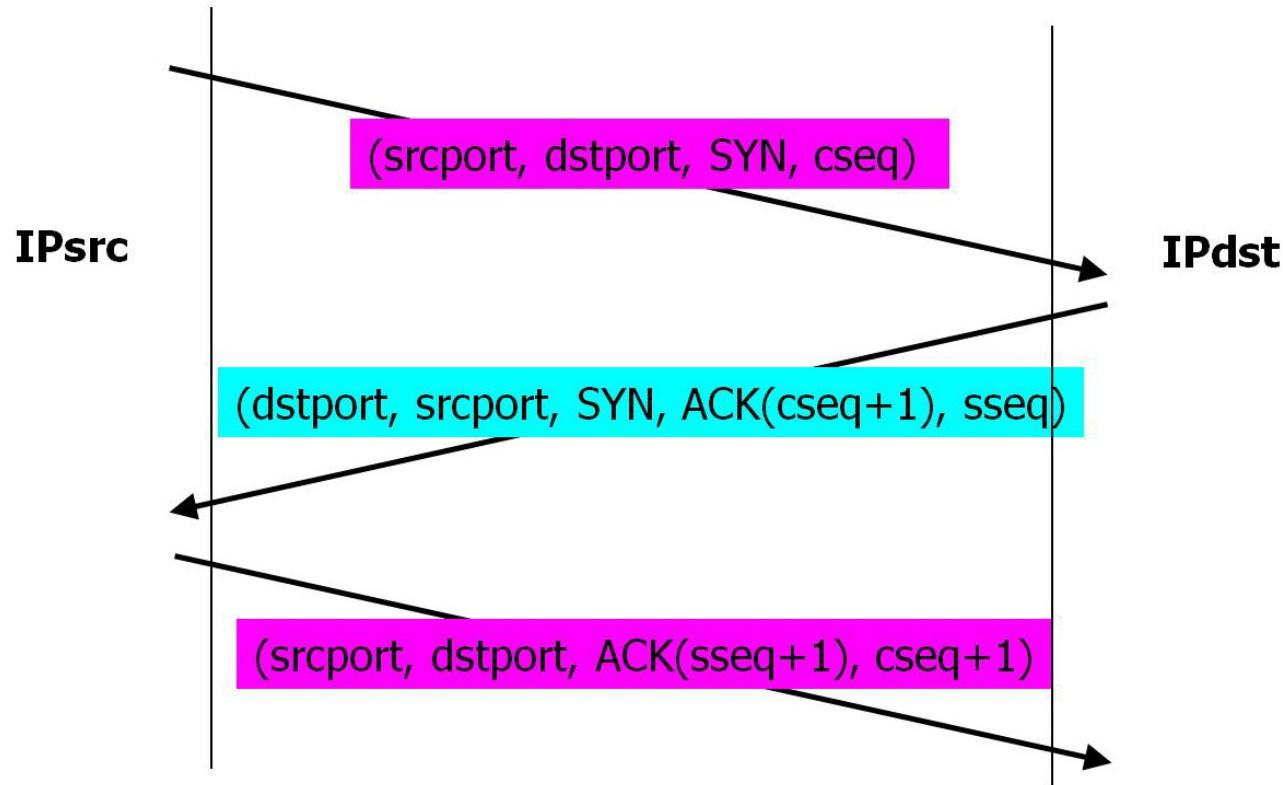
- Add the policy:
 - any inside host can send mail to the outside

action	ourhost	port	theirhost	port	comment
allow	*	*	*	25	<i>connection to their SMTP</i>

- Very bad: we can not control the type of traffic originated from port 25 and coming from the outside
- Then: rules have to specify the direction of the traffic

How to check the direction of TCP?

- Consider the TCP flags





Example with traffic direction

- We distinguish the replies to our SMTP connection considering the ACK flag

action	src	port	dest	port	flags	comment
allow	{our hosts}	*	*	25		<i>connection to their SMTP</i>
allow	*	25	*	*	ACK	<i>their replies</i>
block	*	*	*	*		<i>default</i>

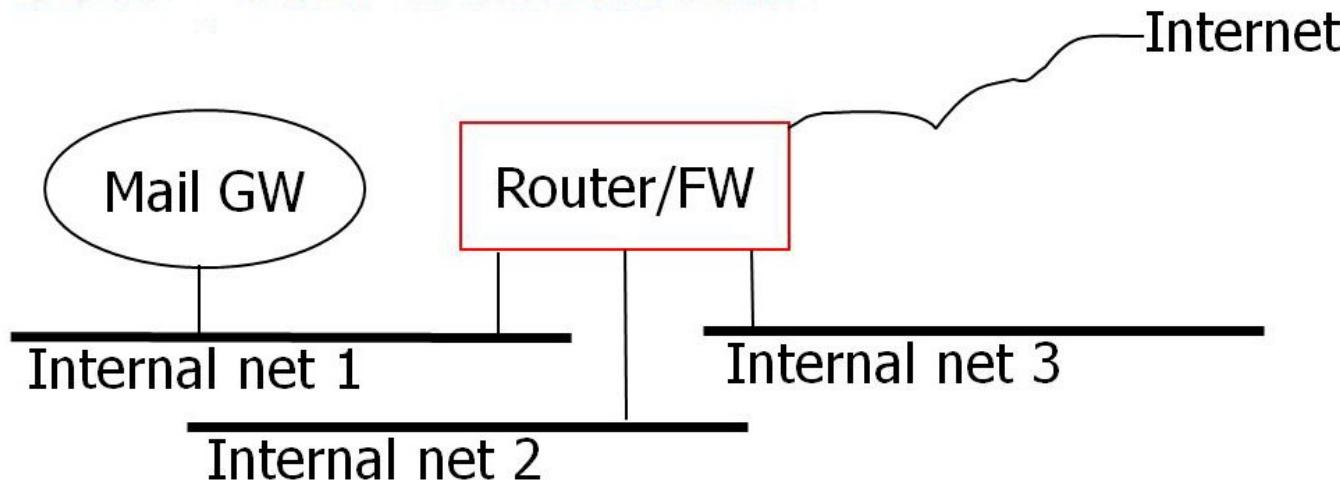
- Very easy case...



Filter rules for network firewalls

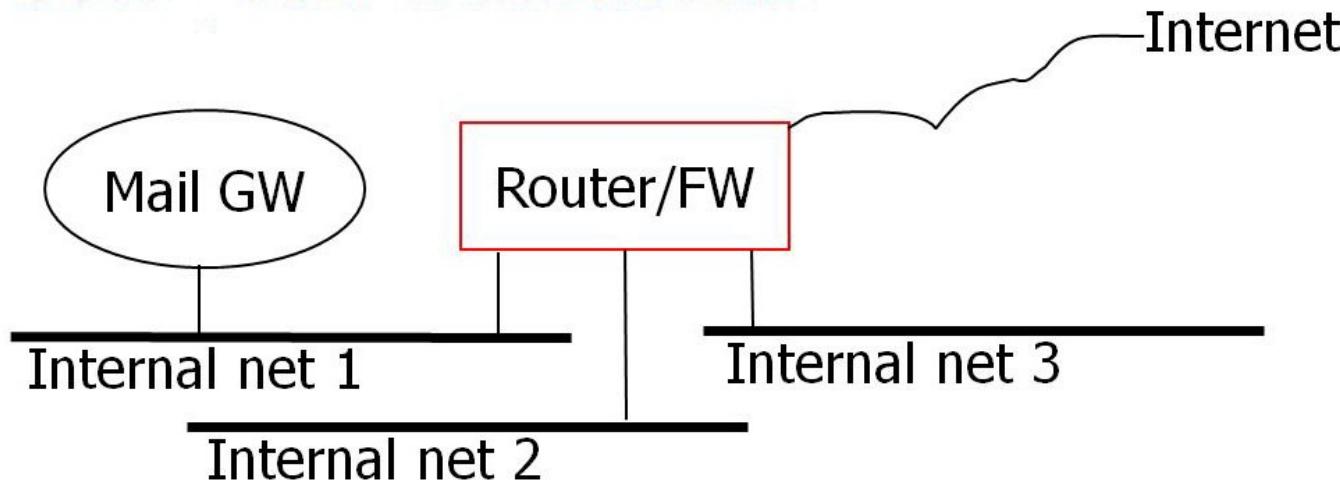


More complex network topology



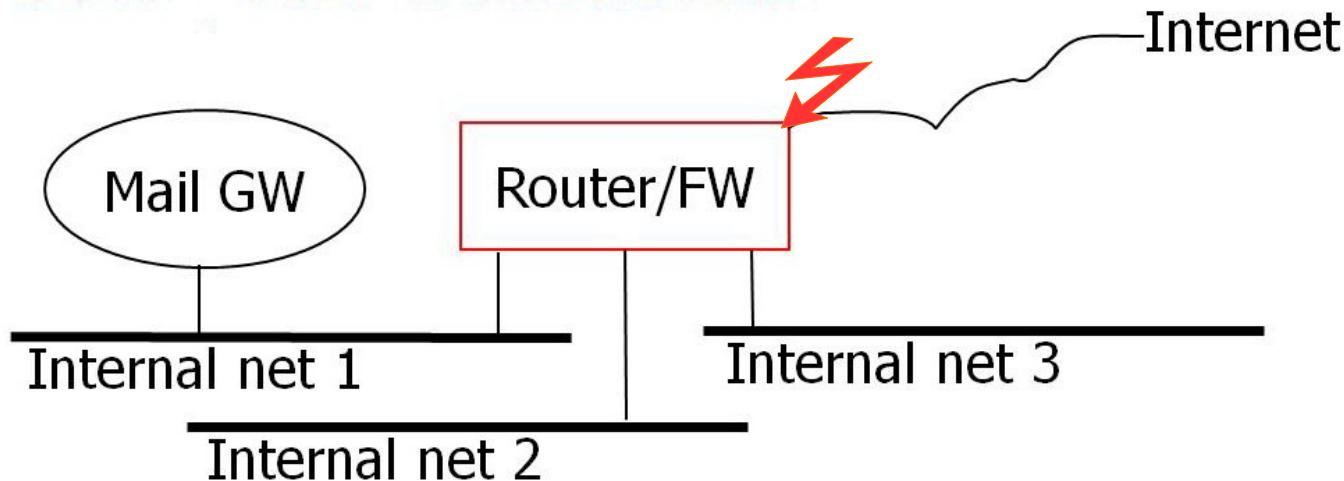
- Policy:
 - Internal Net 1 is a DMZ and only hosts Mail GW
 - Very limited connections between Mail GW and Internet (only partner servers)
 - Limited connections allowed between Mail GW and net 2 and net 3
 - Anything can pass between net 2 and net 3
 - Outgoing requests only between net 2 or net 3 and the link to the Internet

Requirements



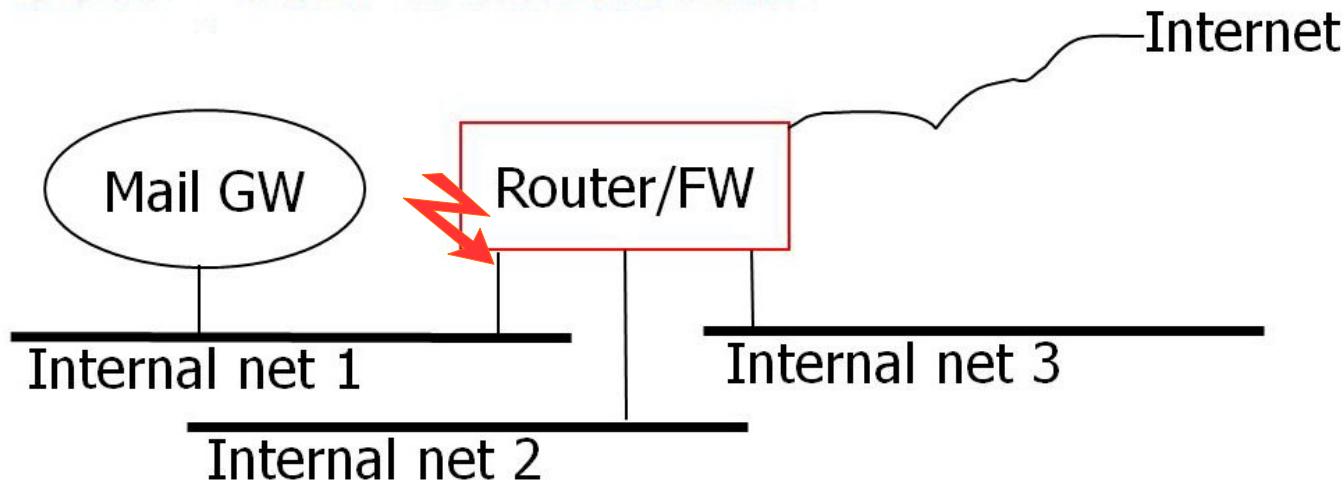
- We cannot only consider where packets have to go (destination → **egress filtering**)
 - Open access to net 2 only allowed for traffic with source address in net 3
 - No way to avoid fake source addresses (*address spoofing*) from outside
- We need to define rules based on **from where packets are arriving**, (source → **ingress filtering**)

Interface towards Internet



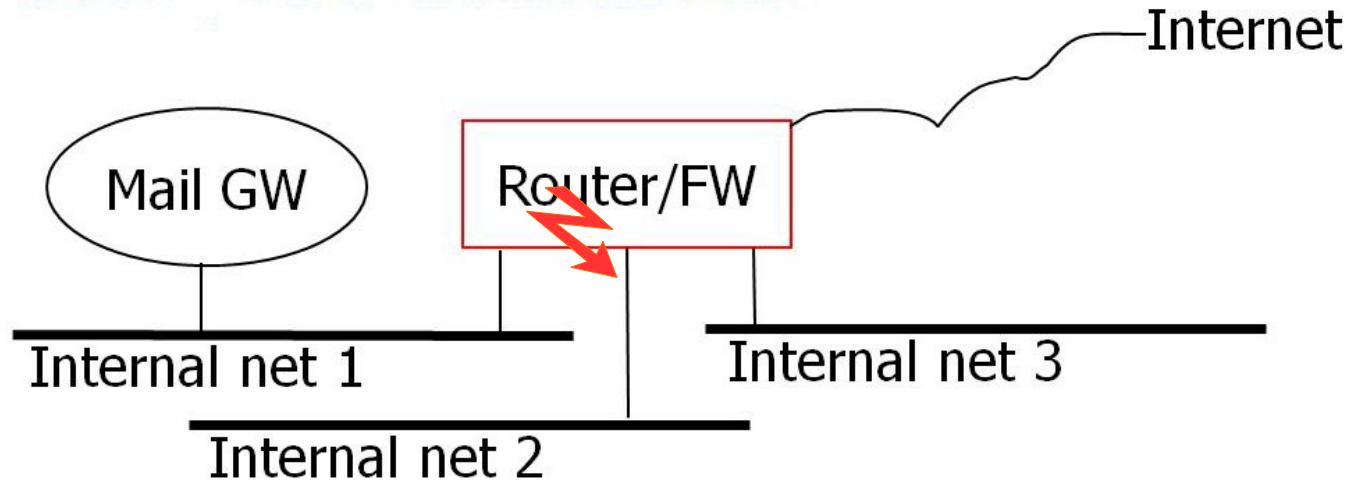
Action	IPsrc	srcport	IPdst	dstport	flags
block	"Net 1"	*	*	*	
block	"Net 2"	*	*	*	
block	"Net 3"	*	*	*	
allow	*	*	GW	25	
allow	*	*	"Net 2"	*	ACK
allow	*	*	"Net 3"	*	ACK

Interface on net 1



Action	IPsrc	srcport	IPdst	dstport	flags
allow	GW	*	"partners"	25	
allow	GW	*	"Net 2"	*	ACK
allow	GW	*	"Net 3"	*	ACK
block	GW	*	"Net 2"	*	
block	GW	*	"Net 3"	*	
allow	GW	*	*	*	

Interface on net 2 (net 3 is similar)



action	IPsrc	srcport	IPdst	dstport	flags
allow	“Net 2”	*	*	*	
block	*	*	*	*	



Problems with Packet Filters

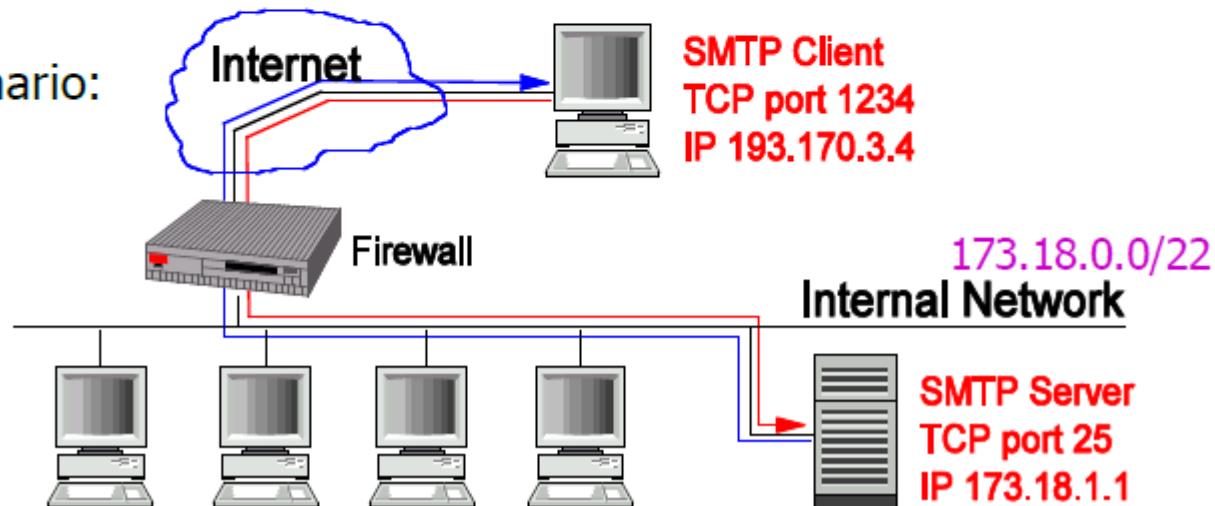
- Only a small number of parameters
 - it is (unfortunately) easy to specify filtering rules which are too specific or too general
- Payload of TCP packet is not inspected
 - No protection against attacks based on upper-layer vulnerabilities
- Limited logging ability (restricted to the few parameters used by the filter)
- No authentication facilities
- Susceptible to attacks based on vulnerabilities in various implementations of TCP and/or IP

Filter rules, 1

- Example: Filtering in- and outgoing SMTP traffic. Try rules:

Rule	In/out	IPsrc	IPdst	Proto	dstport	Action
A	Inward	External	Internal	TCP	25	Allow
B	Outward	Internal	External	TCP	>1023	Allow
C	Outward	Internal	External	TCP	25	Allow
D	Inward	External	Internal	TCP	>1023	Allow
E	*	*	*	*	*	Block

- Scenario:





Filter rules, 2

- Example: Filtering in- and outgoing SMTP traffic. Try rules:

Rule	In/out	IPsrc	IPdst	Proto	dstport	Action
A	Inward	External	Internal	TCP	25	Allow
B	Outward	Internal	External	TCP	>1023	Allow
C	Outward	Internal	External	TCP	25	Allow
D	Inward	External	Internal	TCP	>1023	Allow
E	*	*	*	*	*	Block

- Scenario:

Packet	In/out	IPsrc	IPdst	Proto	dstport	Action
1	Inward	193.170.3.4	173.18.1.1	TCP	25	Allow (A)
2	Outward	173.18.1.1	193.170.3.4	TCP	1234	Allow (B)

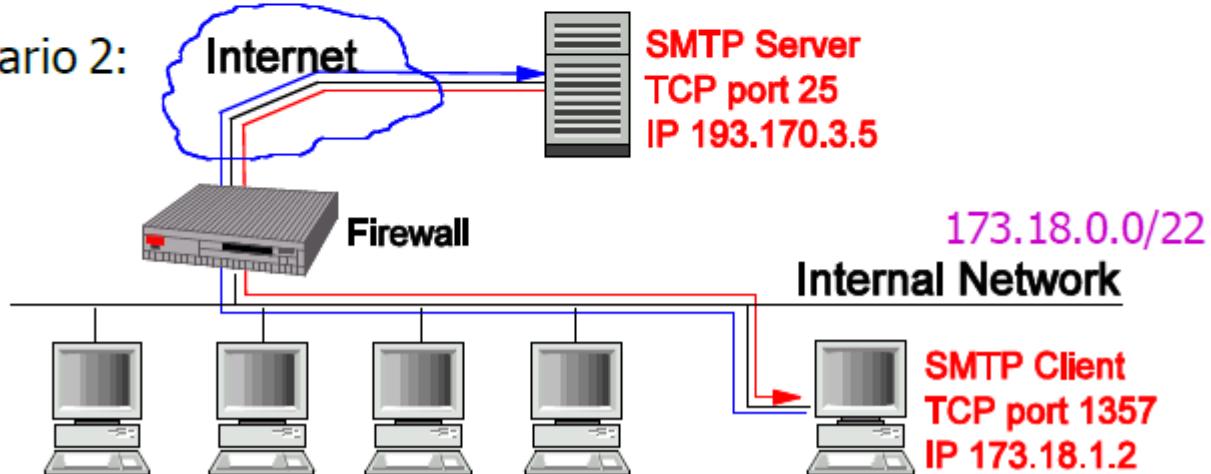
- Conclusion: This looks OK!

Filter rules, 3

- Example: Filtering in- and outgoing SMTP traffic. Try rules:

Rule	In/out	IPsrc	IPdst	Proto	dstport	Action
A	Inward	External	Internal	TCP	25	Allow
B	Outward	Internal	External	TCP	>1023	Allow
C	Outward	Internal	External	TCP	25	Allow
D	Inward	External	Internal	TCP	>1023	Allow
E	*	*	*	*	*	Block

- Scenario 2:





Filter rules, 4

- Example: Filtering in- and outgoing SMTP traffic. Try rules:

Rule	In/out	IPsrc	IPdst	Proto	dstport	Action
A	Inward	External	Internal	TCP	25	Allow
B	Outward	Internal	External	TCP	>1023	Allow
C	Outward	Internal	External	TCP	25	Allow
D	Inward	External	Internal	TCP	>1023	Allow
E	*	*	*	*	*	Block

- Scenario 2:

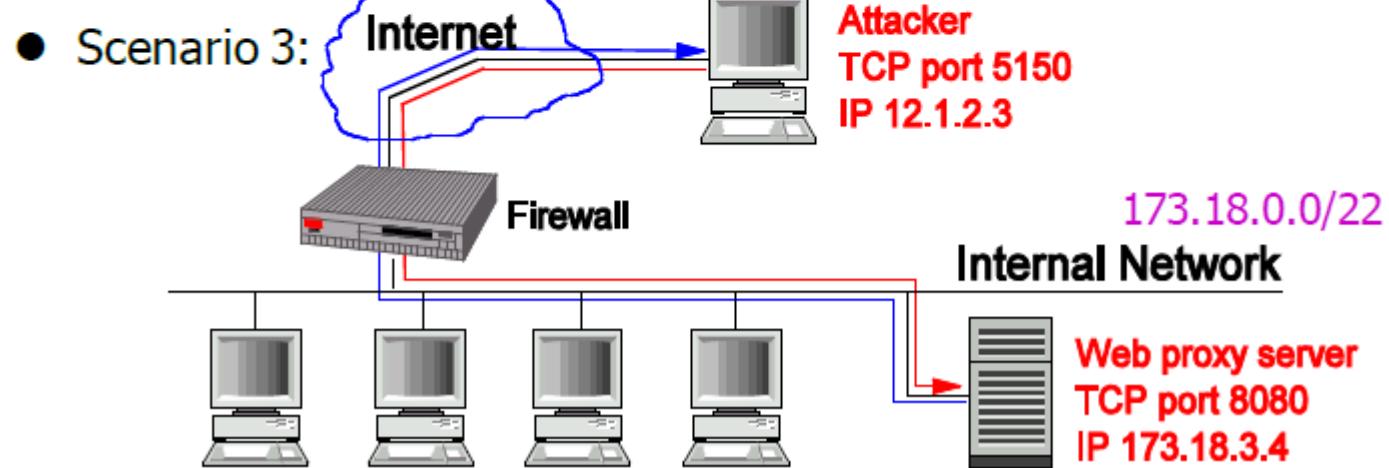
Packet	In/out	IPsrc	IPdst	Proto	dstport	Action
3	Outward	173.18.1.2	193.170.3.5	TCP	25	Allow (C)
4	Inward	193.170.3.5	173.18.1.2	TCP	1357	Allow (D)

- Conclusion: This also looks OK!

Filter rules, 5

- Example: Filtering in- and outgoing SMTP traffic. Try rules:

Rule	In/out	IPsrc	IPdst	Proto	dstport	Action
A	Inward	External	Internal	TCP	25	Allow
B	Outward	Internal	External	TCP	>1023	Allow
C	Outward	Internal	External	TCP	25	Allow
D	Inward	External	Internal	TCP	>1023	Allow
E	*	*	*	*	*	Block





Filter rules, 6

- Example: Filtering in- and outgoing SMTP traffic. Try rules:

Rule	In/out	IPsrc	IPdst	Proto	dstport	Action
A	Inward	External	Internal	TCP	25	Allow
B	Outward	Internal	External	TCP	>1023	Allow
C	Outward	Internal	External	TCP	25	Allow
D	Inward	External	Internal	TCP	>1023	Allow
E	*	*	*	*	*	Block

- Scenario 3:

Packet	In/out	IPsrc	IPdst	Proto	dstport	Action
5	Inward	12.1.2.3	173.18.3.4	TCP	8080	Allow (D)
6	Outward	173.18.3.4	12.1.2.3	TCP	5150	Allow (B)

- Conclusion: Oh, dear! That doesn't look good at all!
- Rules allow all connections where both ends use ports >1023.



Filter rules, 7

- Filtering in- and outgoing SMTP traffic. Include `srcport` in rules:

Rule	In/out	Ipsrc	IPdst	Proto	srcport	dstport	Action
A	Inward	External	Internal	TCP	>1023	25	Allow
B	Outward	Internal	External	TCP	25	>1023	Allow
C	Outward	Internal	External	TCP	>1023	25	Allow
D	Inward	External	Internal	TCP	25	>1023	Allow
E	*	*	*	*	*	*	Block

- Scenario 3:

Packet	In/out	Ipsrc	IPdst	Proto	srcport	dstport	Action
5	Inw.	12.1.2.3	173.18.3.4	TCP	5150	8080	Deny (E)
6	Outw.	173.18.3.4	12.1.2.3	TCP	8080	5150	Deny (E)

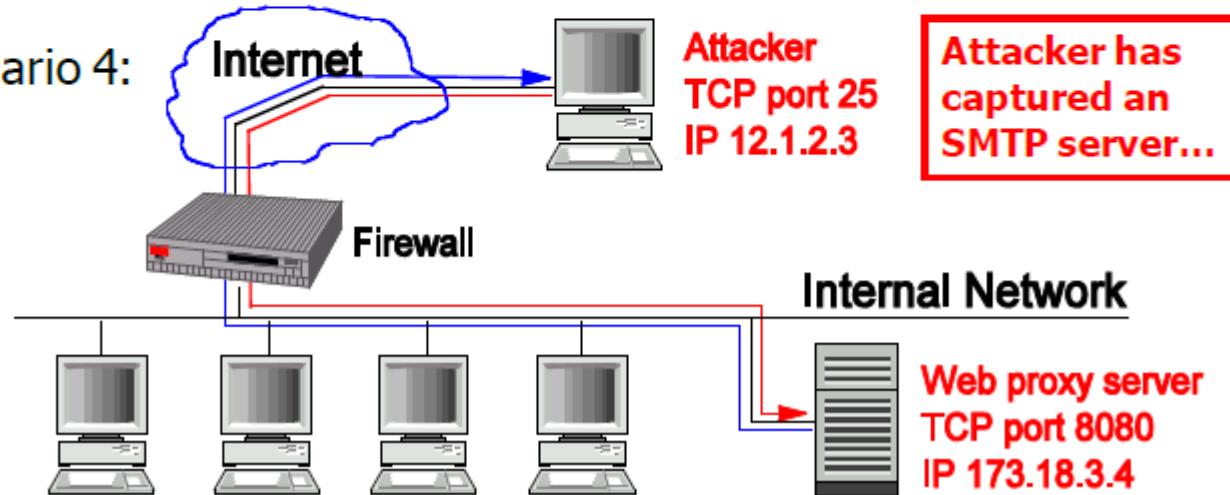
- Conclusion: This looks OK again!
- Check for yourselves that packets 1, 2, 3, 4 are treated OK.

Filter rules, 8

- Filtering in- and outgoing SMTP traffic. Include `srcport` in rules:

Rule	In/out	IPsrc	IPdst	Proto	<code>srcport</code>	dstport	Action
A	In	External	Internal	TCP	>1023	25	Allow
B	Out	Internal	External	TCP	25	>1023	Allow
C	Out	Internal	External	TCP	>1023	25	Allow
D	In	External	Internal	TCP	25	>1023	Allow
E	*	*	*	*	*	*	Block

- Scenario 4:





Filter rules, 9

- Filtering in- and outgoing SMTP traffic. Include `srcport` in rules:

Rule	In/out	IPsrc	IPdst	Proto	<code>srcport</code>	dstport	Action
A	Inward	External	Internal	TCP	>1023	25	Allow
B	Outward	Internal	External	TCP	25	>1023	Allow
C	Outward	Internal	External	TCP	>1023	25	Allow
D	Inward	External	Internal	TCP	25	>1023	Allow
E	*	*	*	*	*		Block

- Scenario 4:

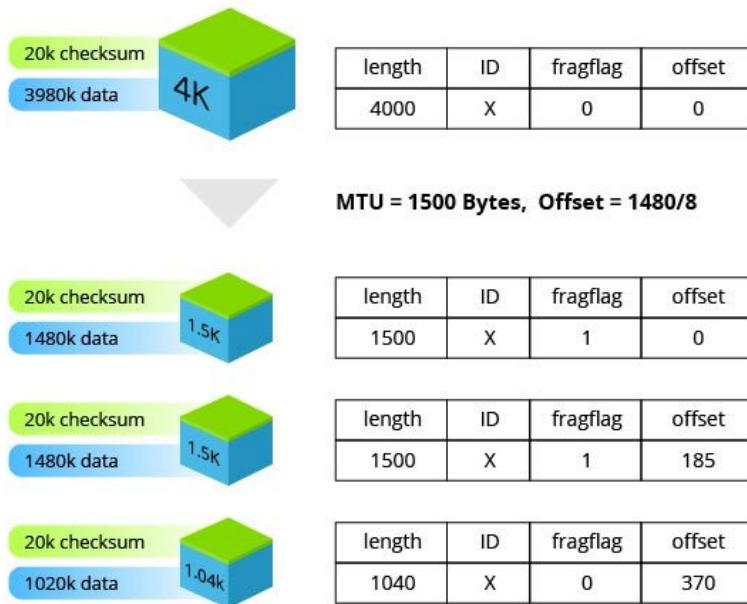
Packet	In/out	IPsrc	IPdst	Proto	<code>srcport</code>	dstport	Action
7	Inw.	12.1.2.3	173.18.3.4	TCP	25	8080	Allow (D)
8	Outw.	173.18.3.4	12.1.2.3	TCP	8080	25	Allow (C)

- Conclusion: This looks bad again!
- Need yet more information (e.g. Flags) to get desired effect: Rules B and D must require ACK flag to be set in order to accept packet.



IP fragmentation

IP Fragmentation and Reassembly (Example)



Length - The size of the fragmented datagram

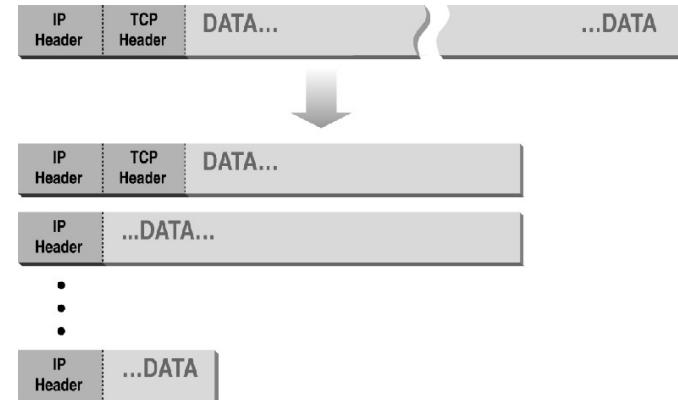
ID - The ID of the datagram being fragmented

Fragflag - Indicates whether there are more incoming fragments

Offset - Details the order the fragments should be placed in during reassembly

<https://www.incapsula.com/ddos/attack-glossary/ip-fragmentation-attack-teardrop.html>

Normal fragmentation



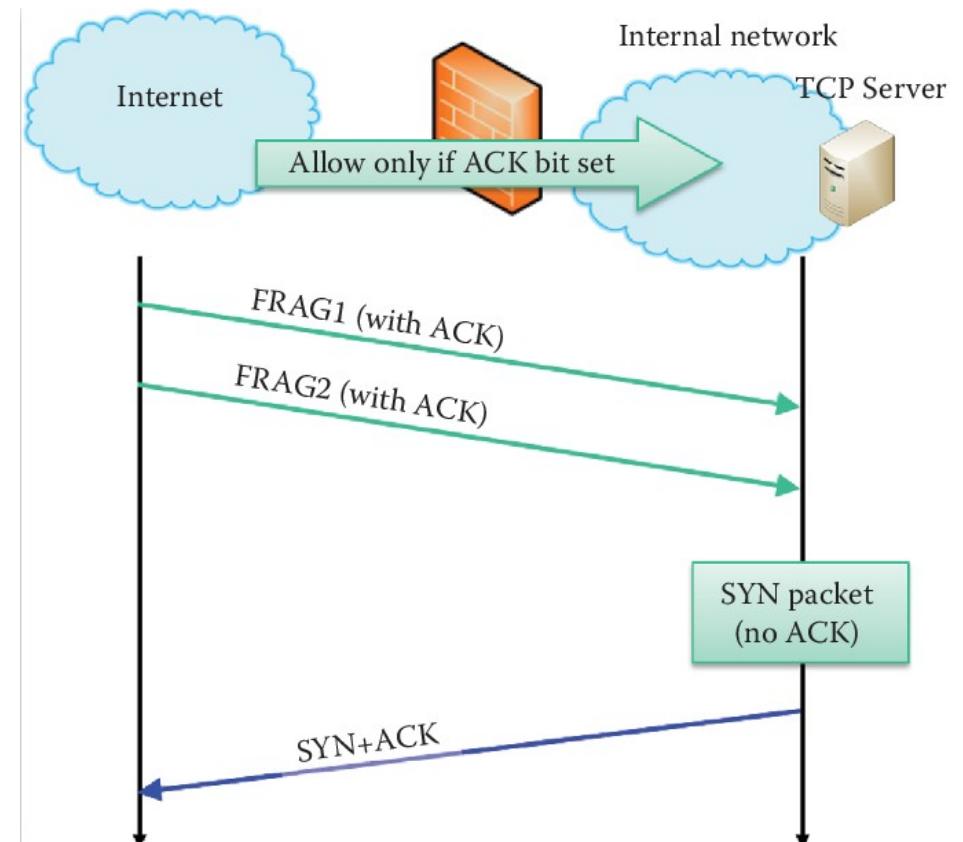
Normal



Abnormal fragmentation

Incoming TCP connections with IP frag

- Firewall blocks any incoming TCP connection
- ACK packet is allowed for outgoing packets
- Internal host reassembles a packet with the SYN bit set because two fragment offsets are chosen in order to set the SYN bit
- Attacks
 - SYN scan
 - Create TCP connection
 - SYN flood - DoS





Stateful firewalls

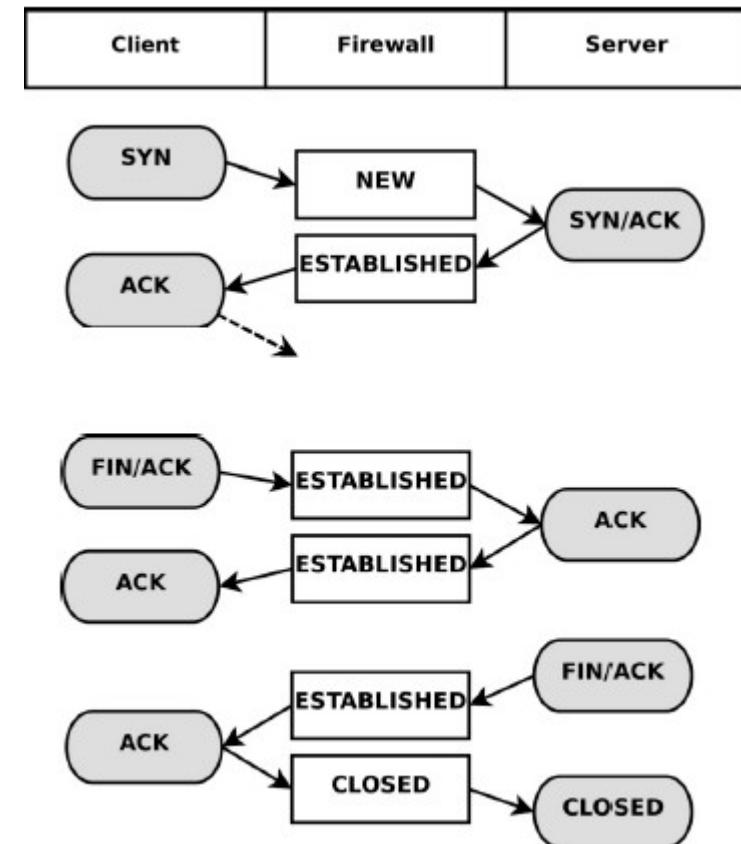
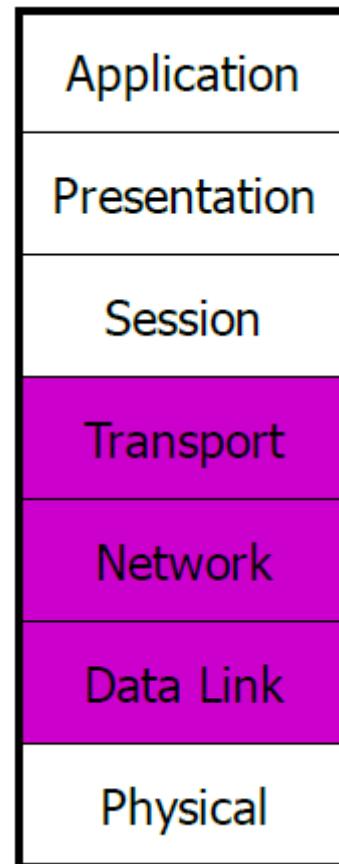


Stateful packet inspection

- Stateful Inspection Firewalls (or Dynamic Packet Filters) can keep track of established connections
- Can drop packets based on their source or destination IP addresses, port numbers and possibly TCP flags
 - Solve one major problem of simple packet filters, since they can check that incoming traffic for a high-numbered port is a genuine response to a previous outgoing request to set up a connection

Stateful firewall

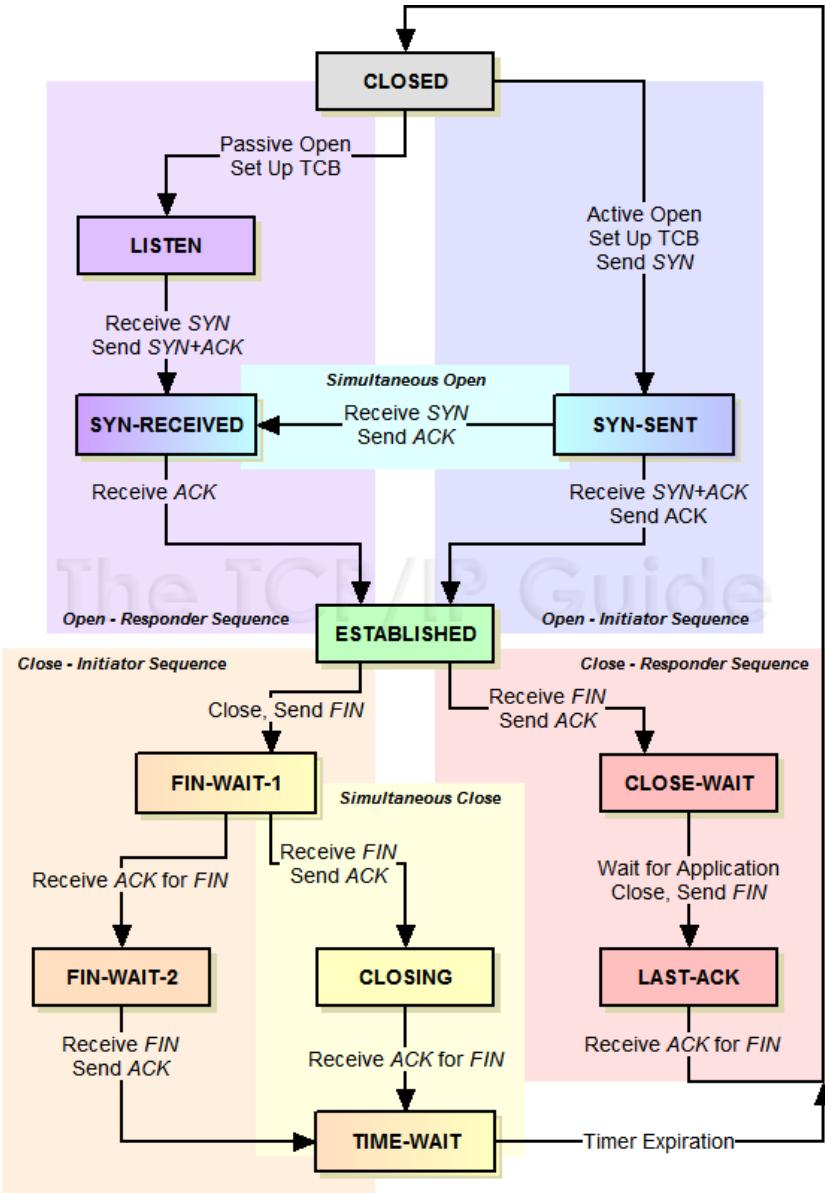
- Considered layers
- Connection tracking



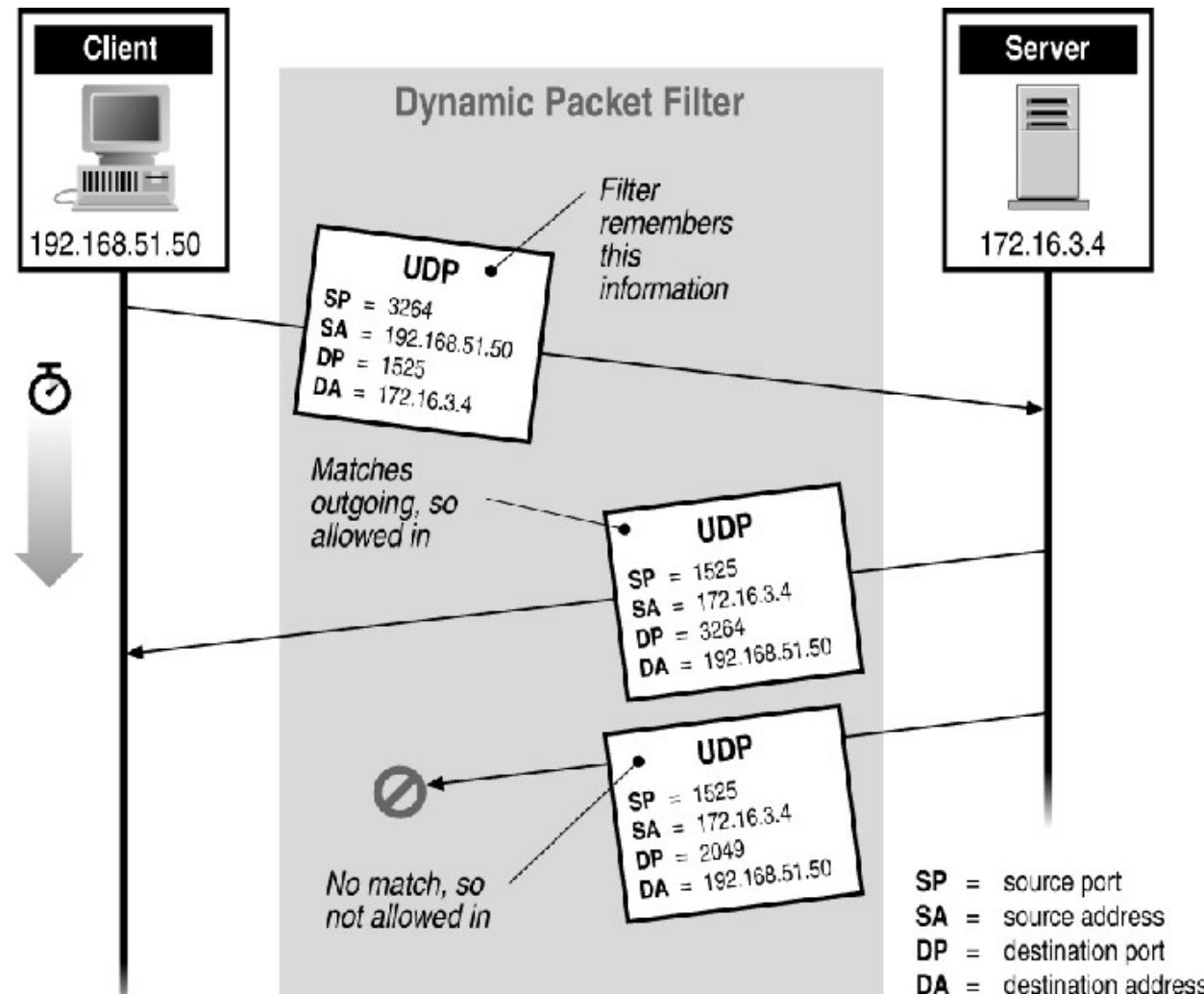
Connection tracking

Considered TCP States

- Setting up connection:
 - client calls from (high-numbered) port to port for application on server
 - server replies to (high-numbered) port on client
 - connection is considered established when the server gives correct SYN/ACK response.
- Closing connection:
 - both parties have to close the connection by sending a TCP packet with FIN flag set before connection is considered closed



Stateful firewall example

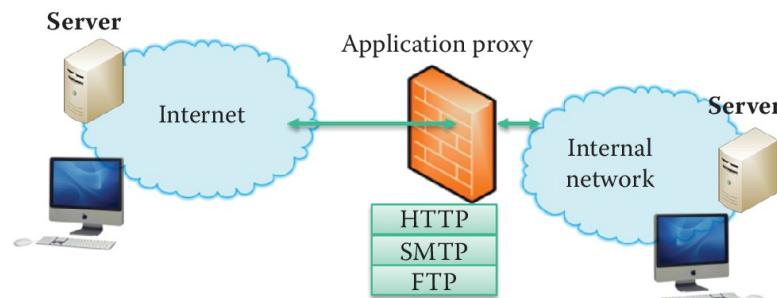




Other types of firewalls

Application-Level filtering (proxy-like)

- Deal with the details of services
- Need a separate special-purpose mechanism for each application
 - Example: mail filters, FTP, HTTP proxy
 - Big overhead, but can log and audit all activity
- Can support user-to-gateway authentication
 - Log into the proxy server with username and password
 - Example: Microsoft ISA, SQUID





Host based firewalls

- A firewall on each individual host to protect that one machine
- Selectively enable specific services and ports that will be used to send and receive traffic
 - Ex: it's unlikely that an employee would need remote SSH access to her laptop
- A host-based firewall plays a big part in:
 - reducing what is accessible to an outside attacker
 - protecting the other elements of the IT system if one of the component (ex, a process) is compromised



Application-level firewall pro and cons

- + Logging capacity
- + Intelligent filtering
- + User-level authentication
- + Protection from wrong implementations
- - Can introduce lag
- - Application-specific
- - Not always transparent



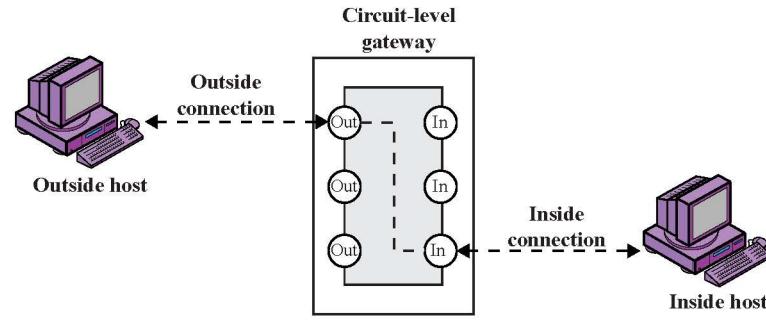
Circuit-level gateways (or generic proxy)

- Also known as a TCP relay
 - Able to deal with several protocols
- SOCKS (v5.0: Internet RFC1928) is the de facto standard
 - It also works with UDP
 - WinSock for Windows
- SOCKS performs at Layer 5 of the OSI model
 - The Session Layer above transport layer
 - TOR (the onion routing): socks-like interface
- The client connect to a proxy that relays its connections in a protocol-independent manner
- Provide user-authentication
- Usually no content filtering



TCP relay

- Splices and relays TCP connections
 - Does not examine the contents of TCP segments
 - Can use ACL (like packet filtering, i.e. dst IP/dst port)
 - Less control than application-level gateway
- Client applications must be adapted for SOCKS
 - “Universal” interface to circuit-level gateways
- Example: `ssh -D 12345 <remote_host>`
 - More on this when talking about tunneling





Common firewall weaknesses

- No content inspection causes the problems
 - Software weakness (e.g. buffer overflow, and SQL injection exploits)
 - Protocol weakness (WEP in 802.11)
- No defense against
 - Denial of service
 - Insider attacks
- Firewall failure has to be prevented
 - Firewall cluster for redundancy



NG-Firewalls

- Next Generation firewalls try to include additional features
- Not only traffic filtering, but also:
 - Intrusion Detection System
 - VPN gateway
 - Deep Packet Inspection
 - Traffic shaping



Summary!



Summary

- Traffic regulation: routers and firewall
 - Decide the packets that can pass through the node
- Firewall architectures: where they go in the network?
 - Network segmentation and DMZ
- Types of firewalls:
 - Host firewall, stateless, stateful, application-gateway, circuit-gateway
- Stateless firewall weaknesses
 - No state, IP fragmentation



That's all for today

- **Questions?**
- See you next lecture!
- Resources:
 - “Building internet firewalls”, Elizabeth D. Zwicky, Simon Cooper, D. Brent Chapman, O'Reilly 2nd ed. (old but still very useful for educational purposes)
 - https://docstore.mik.ua/orelly/networking_2ndEd/fire/index.htm
 - “Firewalls and Internet security: repelling the wily hacker”, William R. Cheswick, Steven M. Bellovin, Aviel D. Rubin, Addison-Wesley 2nd ed.

Practical Network Defense

Master's degree in Cybersecurity 2021-22

Network traffic regulation with iptables

Angelo Spognardi
[*spognardi@di.uniroma1.it*](mailto:spognardi@di.uniroma1.it)
Dipartimento di Informatica
Sapienza Università di Roma



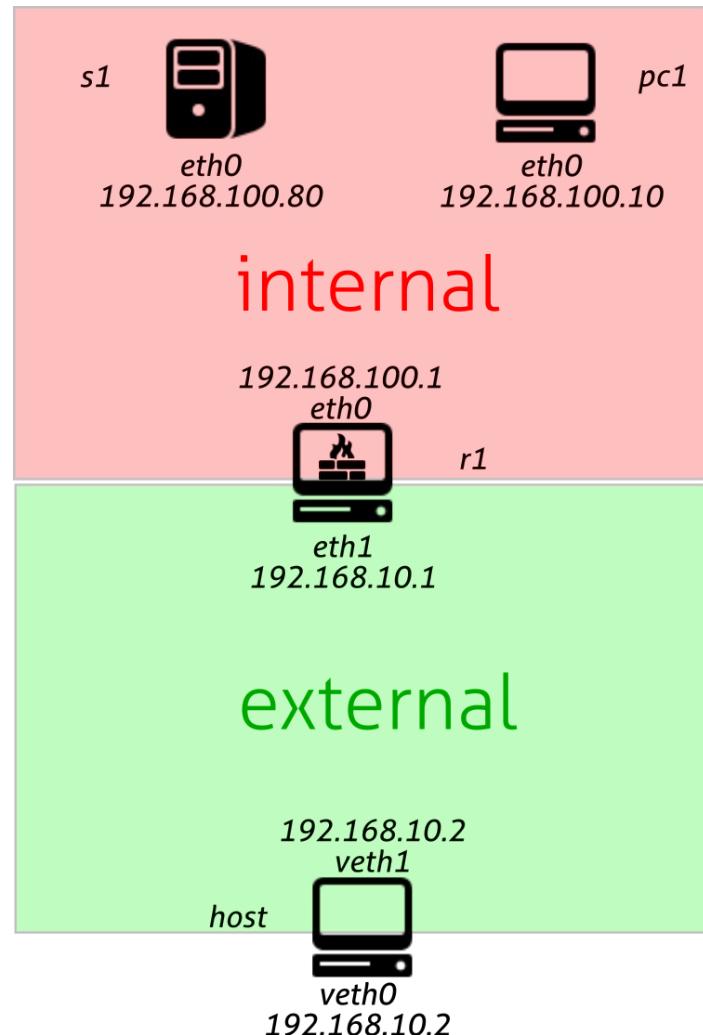
Lab activity

iptables



Network setup

- Use lab4/ex1
- Connect from the host machine so that it is in the external network
- Add a route towards internal via r1-eth1





First Demo

Objective: **block any ping to our pc1**

- Start capturing with wireshark
- Firstly, verify we can ping from s1 and from host
- Then raise our firewall, using iptables

```
iptables -A INPUT -p icmp --icmp-type echo-request -j DROP
```

- Verify we cannot ping pc1 anymore, but we can ping the others
- Check with tcpdump what's going on...
- When done, clean iptables rules

```
iptables -F
```



Second demo

Objective: **exclude any service but HTTP on s1**

- Start capturing with wireshark
- Firstly, verify we can connect from host and pc1 to host to ssh and web server (through the different ports)
- Then raise our firewall on s1, using iptables
`iptables -A INPUT -p tcp --destination-port 80 -j ACCEPT`
`iptables -A INPUT -j REJECT`
- Verify we cannot reach s1 any more (with ssh)
- Check with wireshark what's going on...
- When done, clean iptables rules
`iptables -F`



Iptables

- It is the implementation of a packet filtering firewall for Linux that runs in kernel space
 - It is the evolution of ipchains and ipfw. Coming successor will be nftables
- iptables tool inserts and deletes rules from the kernel's packet filtering table
- It can also operate at the Transport layer (TCP/UDP)
- Old but still extremely valuable tutorial:

www.frozenthux.net/iptables-tutorial/iptables-tutorial.html



Iptables fundamentals

- The rules are grouped in **tables**
 - For now, we focus on the **FILTER** table
- Each table has different **CHAINS** of rules
- Each packet is subject to each rule of a table
- Packet fates depend on the **first matching rule**
- To see chains and rules of the filter table

`iptables -L`

or (better)

`iptables -L -n -v --line-numbers`

Filter table

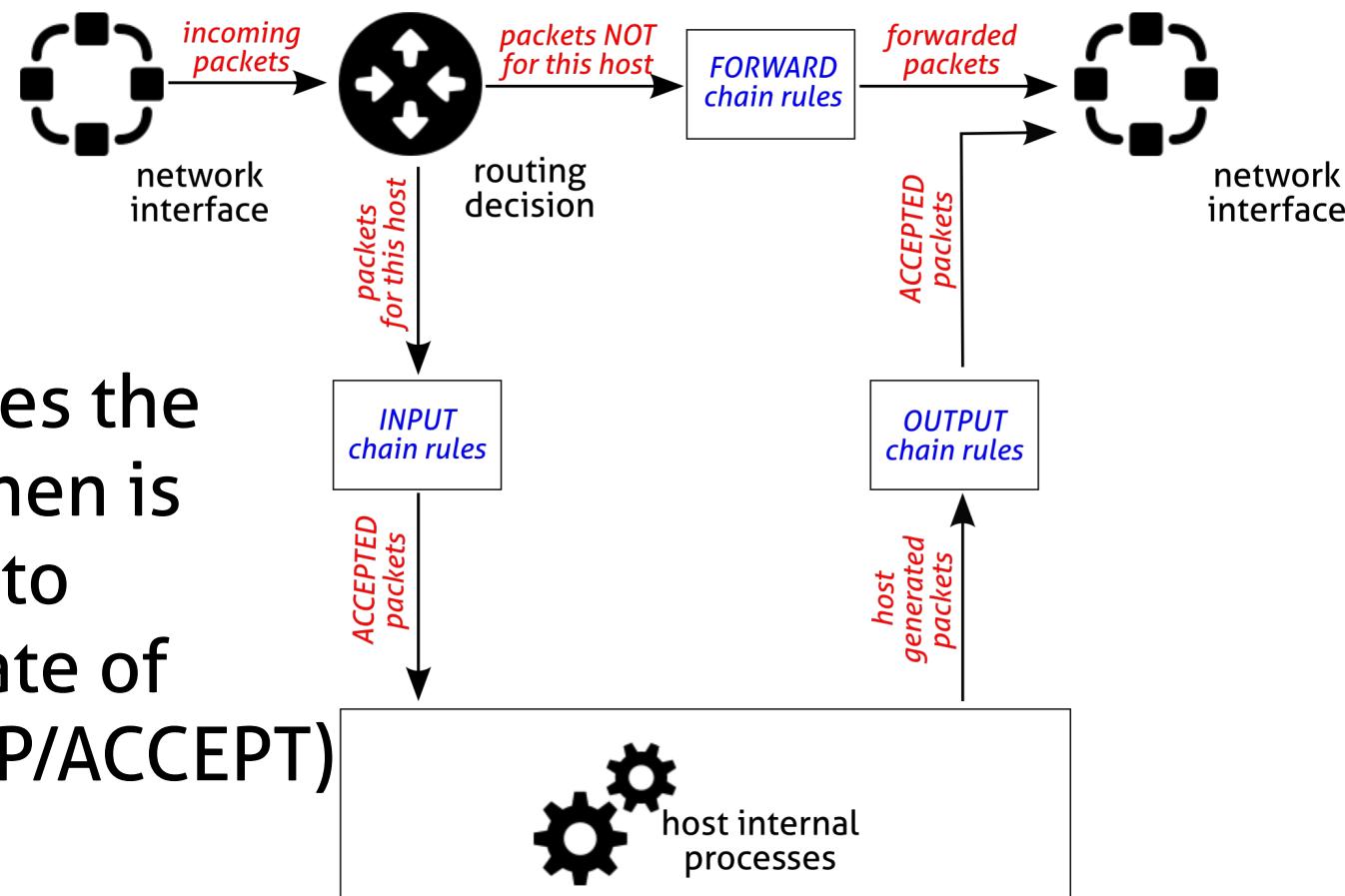
- Three built-in rule chains:

- INPUT

- OUTPUT

- FORWARD

- If a packet reaches the end of a chain, then is the chain policy to determine the fate of the packet (DROP/ACCEPT)





Create and save a rule set

- You can save in a shell script the sequence of the iptables commands
 - Typical structure of iptables_rules.sh

```
#!/bin/bash

# flush (clean) the filter table
iptables -t filter -F

# allow only service XX
iptables ...
```

- Or you can use the built in commands
 - iptables-save > iptables_rules.bk
 - iptables-restore < iptables_rules.bk



Useful iptables command switches

iptables switches	Description
-t table	Specifies the table (filter if not given)
-j target	Jump to the target (it can be another chain)
-A chain	Append a rule to the specified chain
-F	Flush a chain
-P policy	Change the default policy
-p protocol	Match the protocol type
-s ip-address	Match the source IP address
-d ip-address	Match the destination IP address
-p tcp --sport port	Match the tcp source port (also works for udp)
-p tcp --dport port	Match the tcp destination port (also works for udp)
-i interface-name	Match input interface (from which the packet enters)
-o interface-name	Match output interface (on which the packet exits)



Review the rulesets of demos

```
iptables -A input -p icmp -icmp-type echo-request -j DROP
```

```
iptables -A input -p tcp --destination-port 80 -j ACCEPT  
iptables -A input -j REJECT
```

- We can specify different “targets” (this is a subset):
 - **ACCEPT**: the packet is handed over to the end application or the operating system for processing
 - **DROP**: the packet is blocked.
 - **REJECT**: the packet is blocked, but it also sends an error message to the source host of the blocked packet
 - reject-with <qualifier> <qualifier> is an ICMP message
 - **LOG**: the packet is sent to the syslog daemon for logging.
 - iptables continues processing with the next rule in the table.
 - You can't log and drop at the same time → use two rules (--log-prefix "reason")



Other useful iptables command switches

iptables switches	Description
-p tcp --sport port	Match the tcp source port
-p tcp --dport port	Match the tcp destination port
-p udp --sport port	Match the udp source port
-p udp --dport port	Match the udp destination port
--icmp-type type	Match specific icmp packet types
-m <i>module</i>	Uses an extension module
-m state --state s	Enable connection tracking. Match a packet which is in a specific state: NEW: the packet is the start of a new connection ESTABLISHED: the packet is part of an established connection RELATED: the packet is the starting of a related connection (like FTP data) INVALID: the packet could not be identified
-m multiport ...	Enable specification of several ports with one single rule



Modules examples

- Allow both port 80 and 443 for the webserver on inside:

```
iptables -A FORWARD -s 0/0 -i eth0 -d 192.168.1.58 -o eth1 -p TCP \
--sport 1024:65535 -m multiport --dport 80,443 -j ACCEPT
```

- The return traffic from webserver is allowed, but only of sessions are established:

```
iptables -A FORWARD -d 0/0 -o eth0 -s 192.168.1.58 -i eth1 -p TCP \
-m state --state ESTABLISHED -j ACCEPT
```

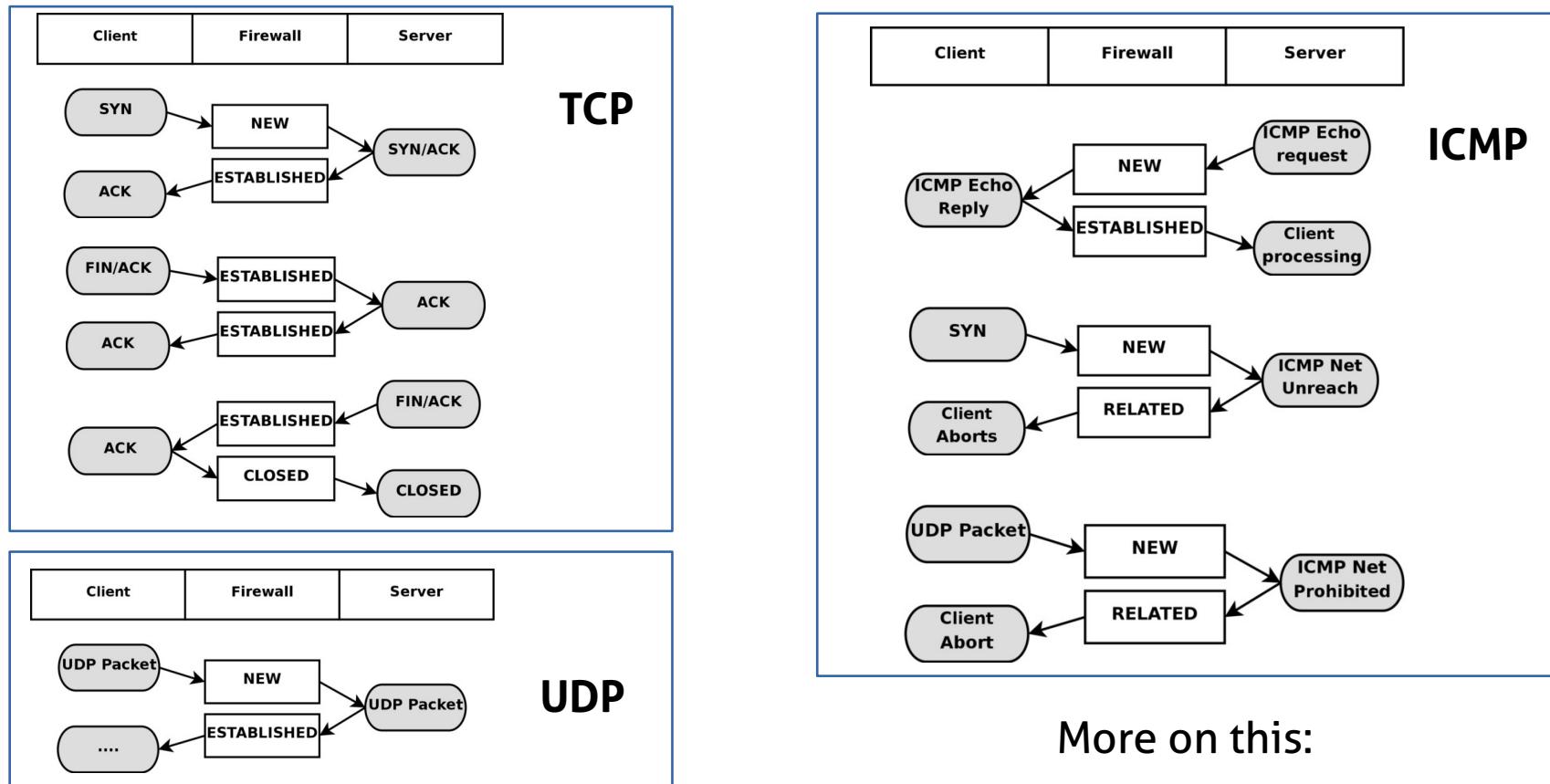
- If sessions are used, you can reduce an attack called half open

Half open is known to consume server all free sockets (tcp stack memory) and is sensed as a denial of service attack, but it is not.

Sessions are usually waiting 3 minutes.

More on the conntrack module

- Clever use of logic to recognize connections, even with connection-less protocols (UDP, ICMP...)



More on this:

<https://www.frozentux.net/iptables-tutorial/iptables-tutorial.html#STATEMACHINE>



Lab activity



Main tasks

- Iptables and ip6tables
- Reference links:
 - Linux ipv6 configuration: ipv6 sysctl
 - <https://www.kernel.org/doc/Documentation/networking/ip-sysctl.txt>
 - Iptables reference manual
 - www.frozentux.net/iptables-tutorial/iptables-tutorial.html



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>
- You can move in the directory and run `lstart`
 - **NOTE:** launch docker first or the first `lstart` attempt can (...will...) fail



Lab activity: ex1, ex2



Useful hints before starting the labs

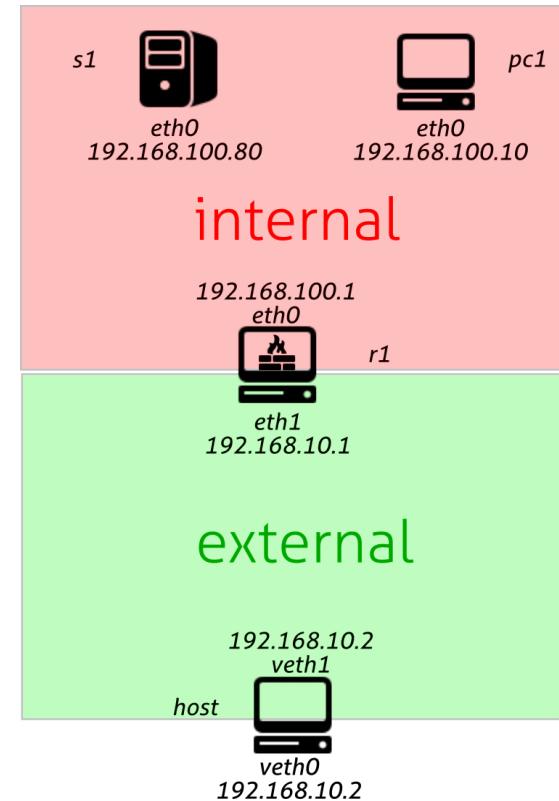
- Connect the kathara-host using the connect-lab.sh script or adding ip addresses

```
./connect-lab.sh 192.168.10.2/24 external  
ip addr add 2001:db8:cafe:2::2/64 dev veth0
```

- You can add aliases to the /etc/hosts file in every host

```
echo 2001:db8:cafe:2::80/64 s1 >>/etc/hosts
```

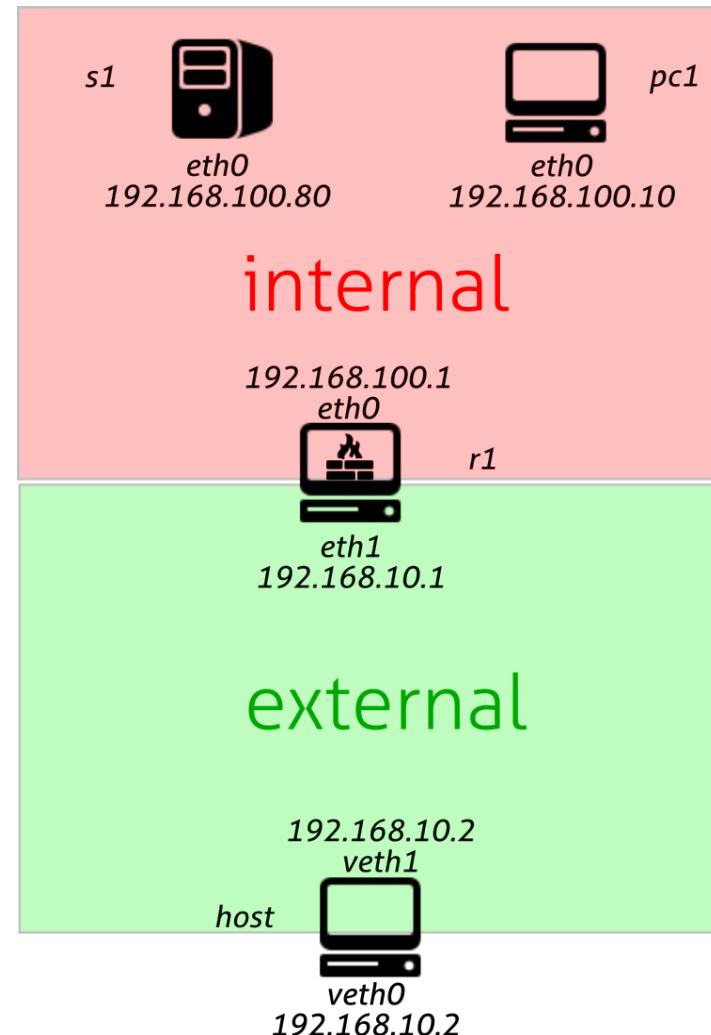
- On s1 http and ssh services are running:
- You can connect to s1 using the login *user:password*
 - ssh user@192.168.100.80





Exercise 1: pnd-labs/lab4/ex1

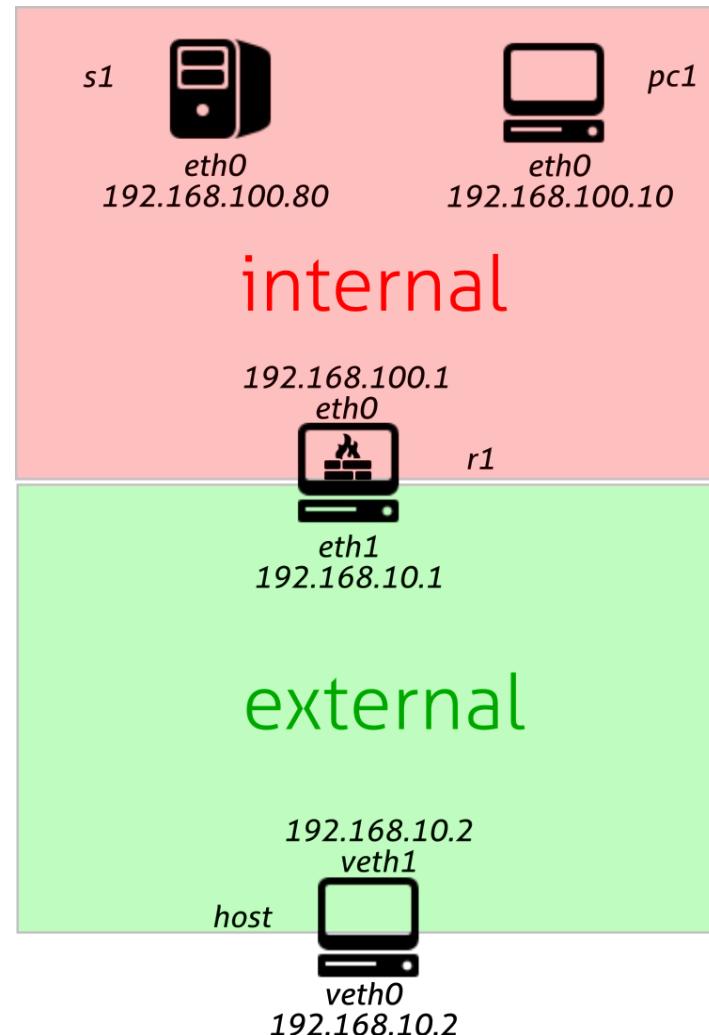
- Start with the previous setting
- Protect the internal network from the external network
 - Configure r1 to only allow HTTP traffic to s1
- Try with other services or ports, also with pc1
 - Ex: ping, ssh, http on different ports





Exercise 2: pnd-labs/lab4/ex2

- Extend ex1 with IPv6
- Repeat the same exercise with the IPv6 addressing
- The internal network is 2001:db8:cafe:1::/64
- The external network is 2001:db8:cafe:2::/64
- You have to use ip6tables

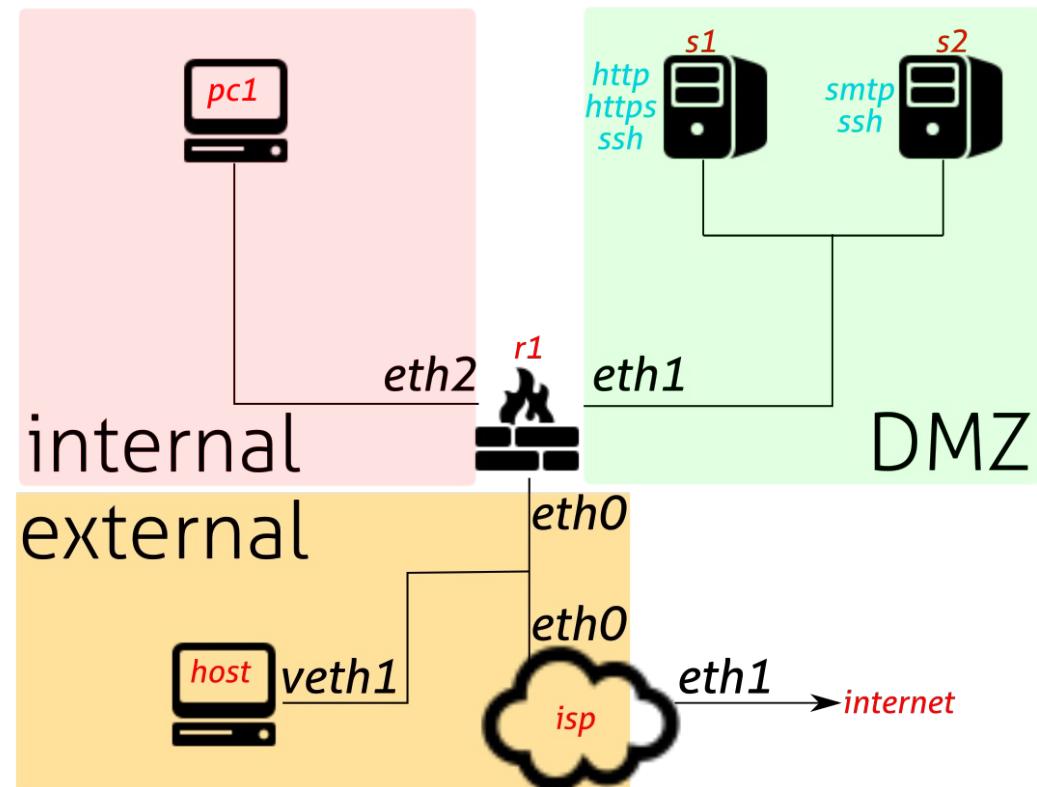




Lab activity: ex3

Exercise 3: pnd-labs/lab4/ex3

- A firewall to protect an internal lan and a DMZ with two servers
- DMZ can be accessed from outside but cannot initiate any connection
- Only internal hosts can also reach DMZ via ssh
- Use both IPv4 and IPv6





That's all for today

- **Questions?**
- See you next lecture!
- Resources:
 - “Building internet firewalls”, Elizabeth D. Zwicky, Simon Cooper, D. Brent Chapman, O'Reilly 2nd ed.
 - https://docstore.mik.ua/orelly/networking_2ndEd/fire/index.htm
 - “Firewalls and Internet security: repelling the wily hacker”, William R. Cheswick, Steven M. Bellovin, Aviel D. Rubin, Addison-Wesley 2nd ed.
 - www.frozentux.net/iptables-tutorial/iptables-tutorial.html

Practical Network Defense

Master's degree in Cybersecurity 2021-22

Network traffic regulation NAT

Angelo Spognardi
[*spognardi@di.uniroma1.it*](mailto:spognardi@di.uniroma1.it)
Dipartimento di Informatica
Sapienza Università di Roma



Today's agenda

- NAT
- iptables
 - Tables
 - Chain priorities
 - Logging

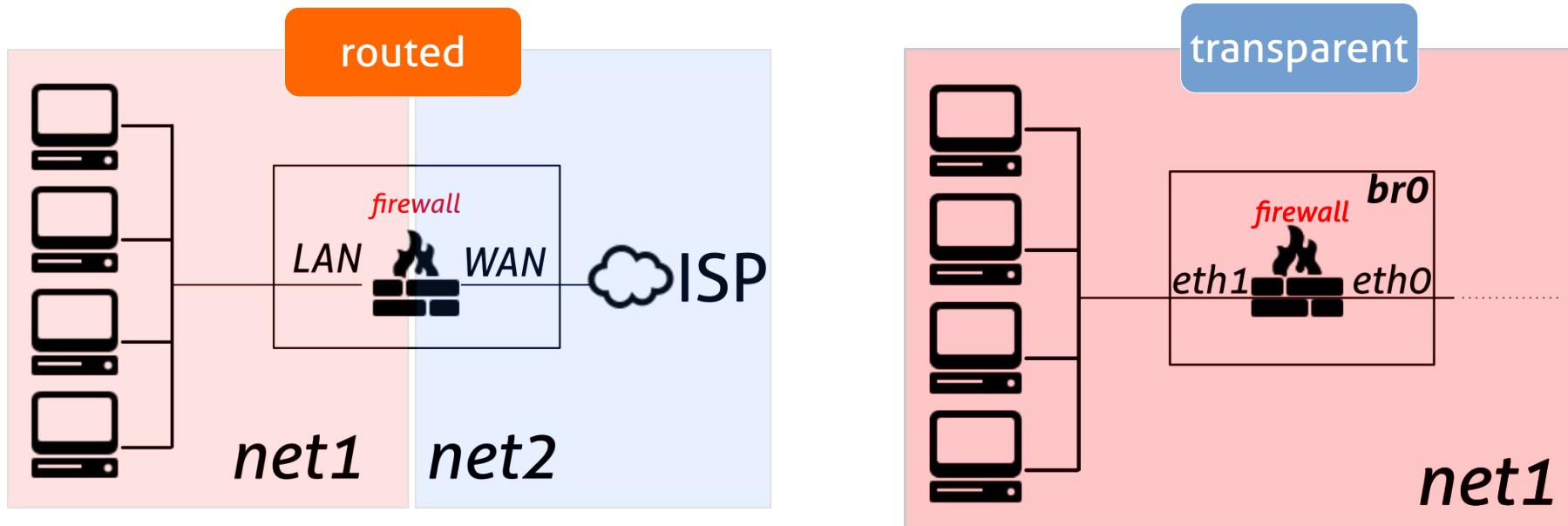


NAT

Network Address Translation

Routed vs Transparent firewall modes

- In routed mode, a firewall is a hop in the routing process
 - It IS a router responsible of its own (internal) networks
- In transparent mode, a firewall works with data at Layer 2 and is not seen as a router hop to connected devices
 - It can be implemented using bridged NICs



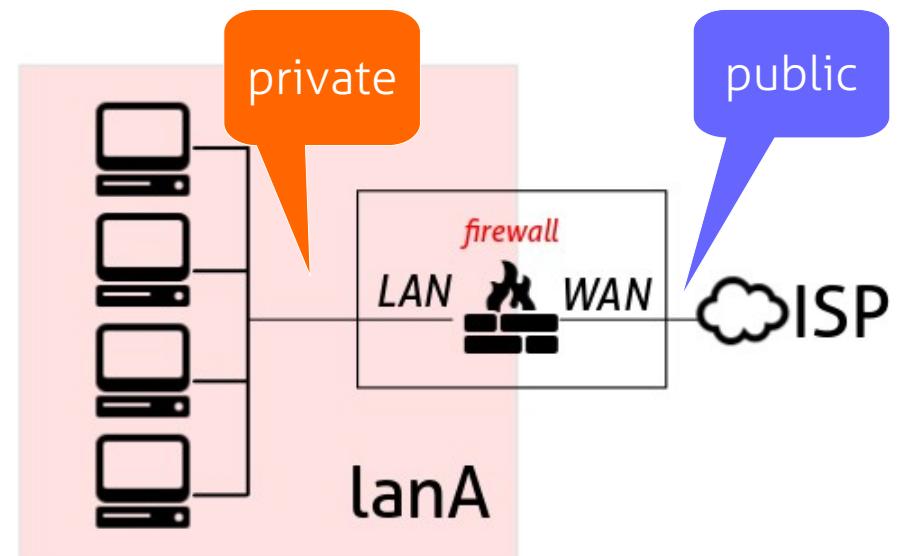


Routable IP Addressing

- Routable addresses need to be unique on the Internet to be **PUBLICLY** reachable → **public IP addresses**
- Non-routable addresses → **Special-Purpose Address IP addresses**
 - Private addresses (<https://www.iana.org/go/rfc1918>):
 - 10.0.0.0 - 10.255.255.255 (10/8 prefix)
 - 172.16.0.0 - 172.31.255.255 (172.16/12 prefix)
 - 192.168.0.0 - 192.168.255.255 (192.168/16 prefix)
 - Loopback addresses (<https://www.iana.org/go/rfc1122>):
 - 127.0.0.0–127.255.255.255 (127/8 prefix)
 - Shared address space (<https://www.iana.org/go/rfc6598>):
 - 100.64.0.0–100.127.255.255 (100.64/10 prefix)
 - Full list: [iana website](#)

Network Address Translation (NAT)

- Translate the address (f.e.: between incompatible IP addressing)
- Informally speaking, connecting to the Internet a LAN using un-routable in-house LAN addresses
- NAT in a routed firewall:
 - Can filter requests from hosts on WAN side to hosts on LAN side
 - Allows host requests from the LAN side to reach the WAN side
 - Does not expose LAN hosts to external port scans



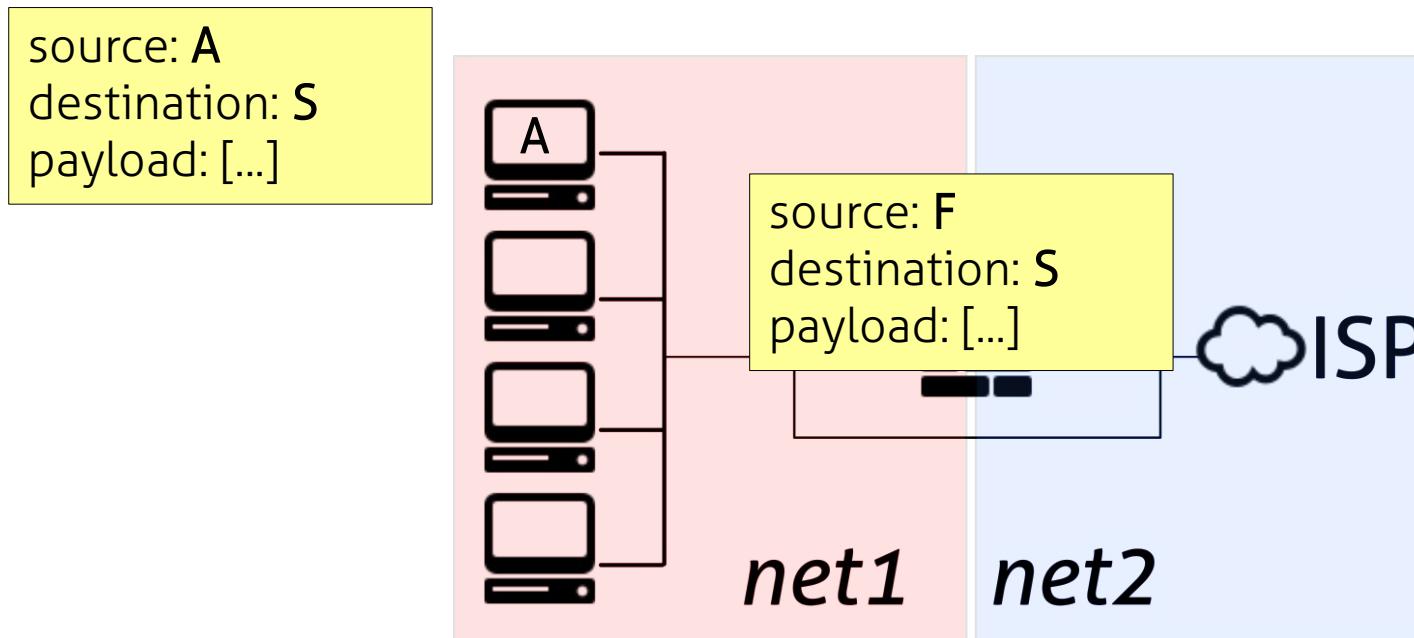


NAT goals

- A private network uses just one IP address provided by ISP to connect to the Internet
- Private networks use private IP addresses provided by IETF
- Can change address of devices in private network without notifying outside world
- Can change ISP without changing the address of devices in private network
- Devices inside private network are not explicitly addressable by external network, or visible by outside world (a security plus)

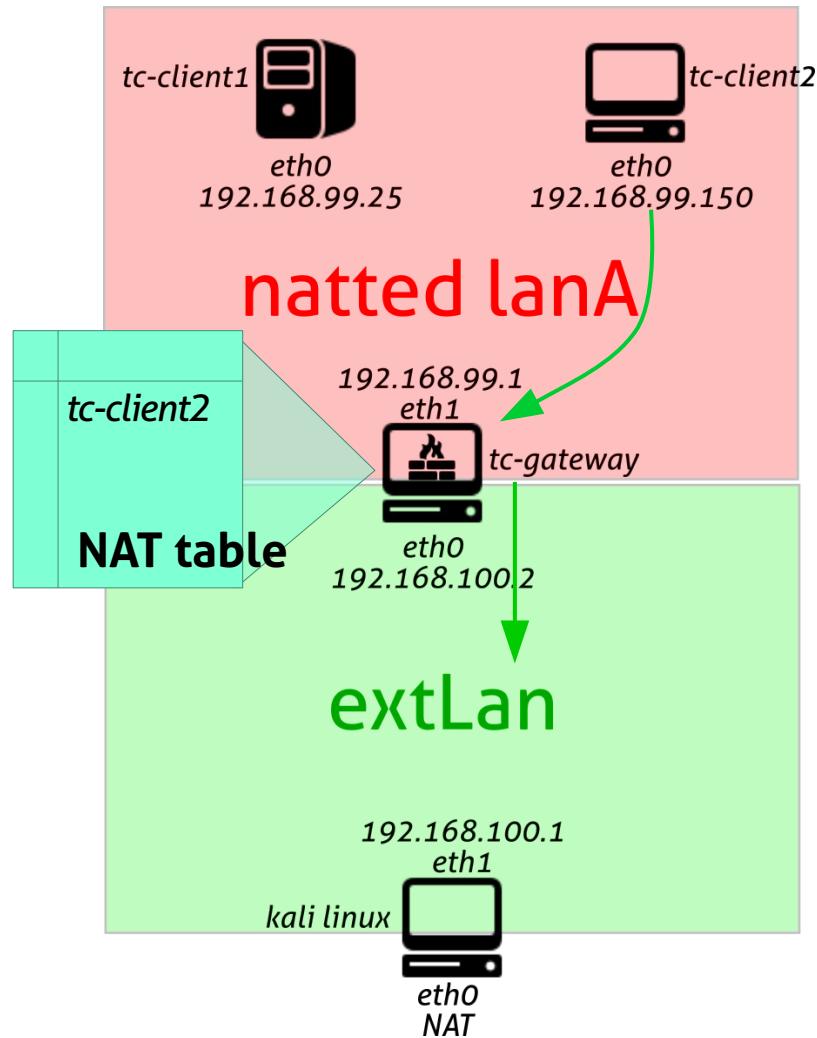
Source NAT (SNAT)

- Translate **outgoing** IP source packet headers from the internal host addresses to the WAN IP address
- The session is **masqueraded** as coming from the NATting device



Source NAT (SNAT)

- The firewall/router:
 - Forwards replies from the client service request by the external server to the client
 - Enables the client-server session or connection to continue on another port as requested by the external server, forwarding any responses by the server to the client (the RELATED connections)
- The NAT table is where associations between requests and internal IP addresses are kept





Types of NAT: Basic and NAPT

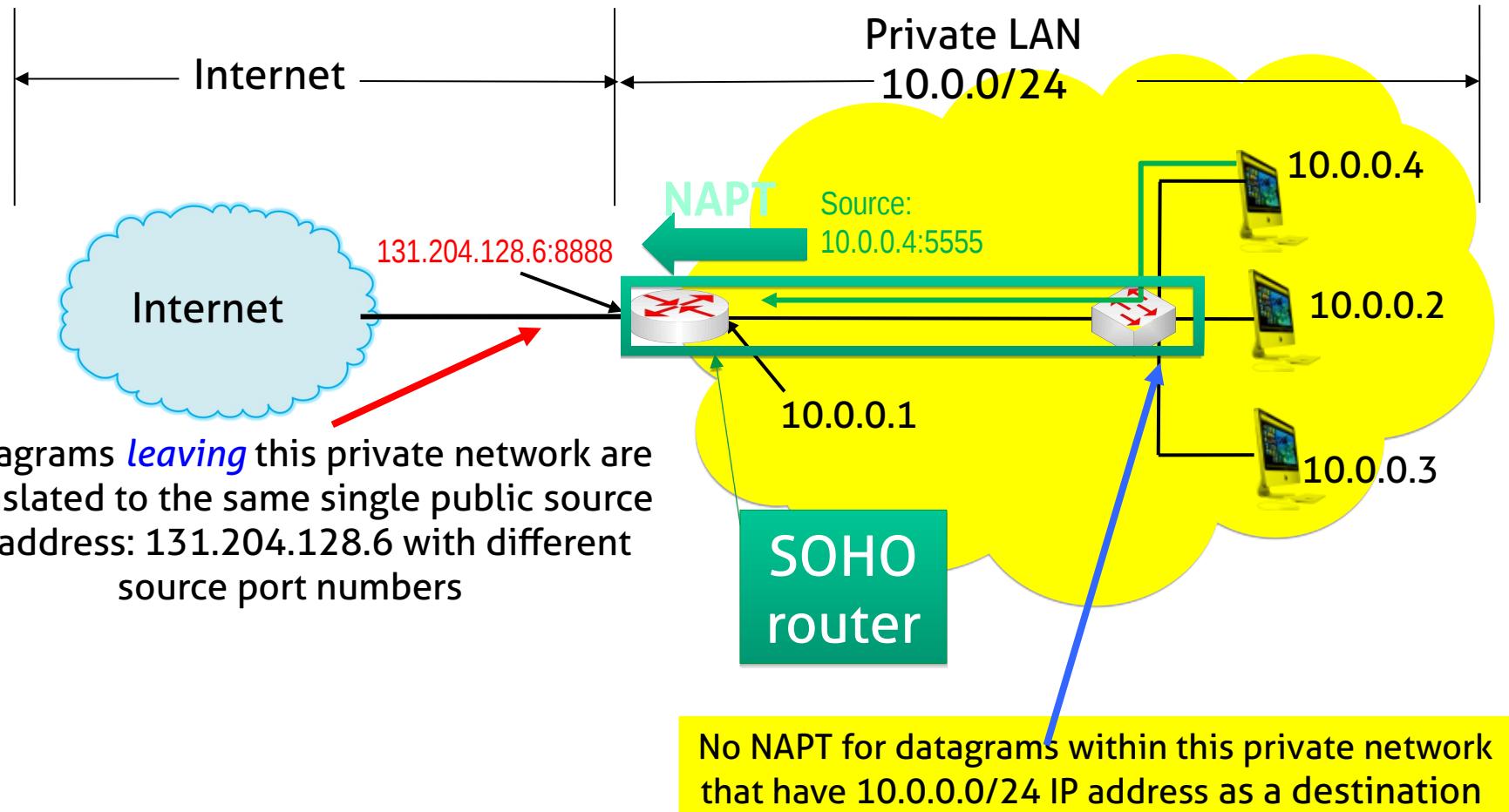
- Basic NAT: a block of external/public IP addresses are set aside for translating the addresses of hosts within a private domain as they originate sessions to the external domain
 - For packets outbound from the private network, the source IP address and related fields such as IP, TCP, UDP and ICMP header checksums are translated
 - For inbound packets, the destination IP address and the checksums as listed above are translated
- However, multiple external/public IP addresses are difficult to obtain due to the shortage of IPv4 addresses
- Network Address Port Translation (NAPT)
- The NAPT also translates transport identifiers, e.g., TCP and UDP port numbers as well as ICMP query identifiers



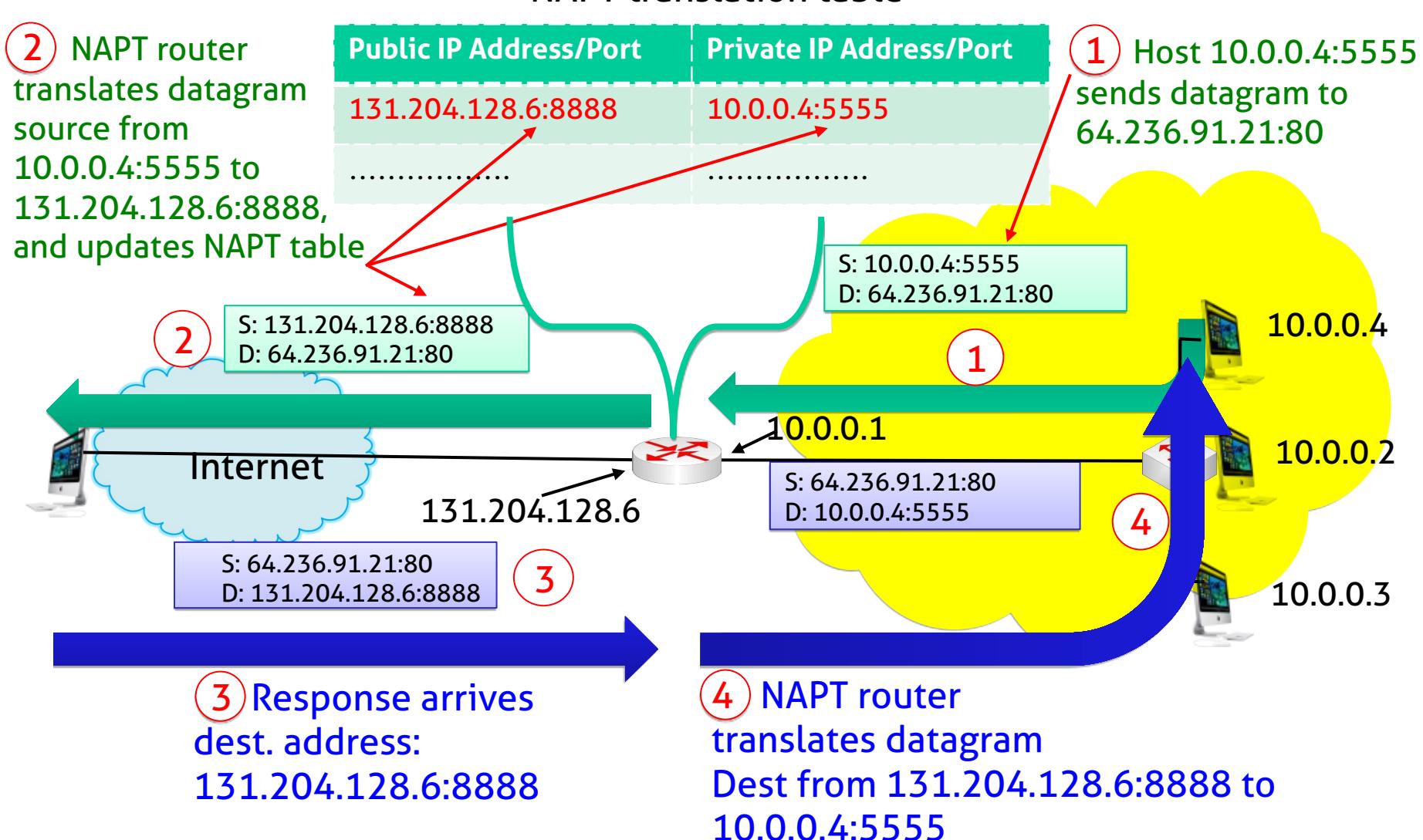
Network Address Port Translation

- Multiplex a number of private hosts into a single external/public IP address
- The IP address binding extends to transport level identifiers such as TCP/UDP ports
 - For most of the small office and home (SOHO) routers
 - The private network usually relies on a single IP address, supplied by the ISP to connect to the Internet
 - SOHO can change ISPs without changing the private IP addresses of the devices within the network
- The terms NAT and NAPT are used interchangeably in the literature
 - However the RFCs, such as RFC 3022, use the term NAPT when port numbers are involved in translation
 - Cisco refers to NAPT as PAT, i.e. Port Address Translation

NAPT: leaving packets

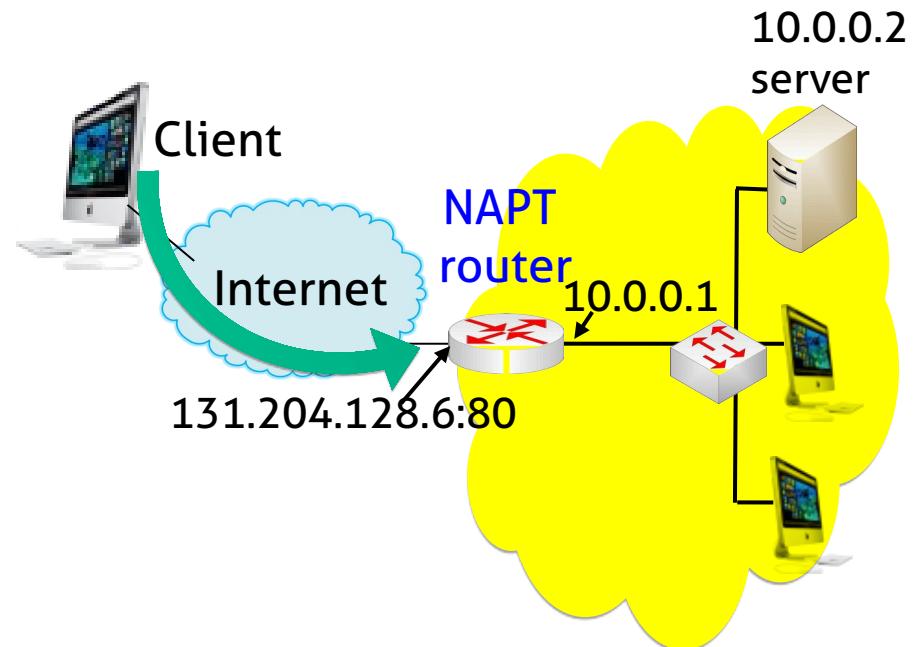


NAPT: responses



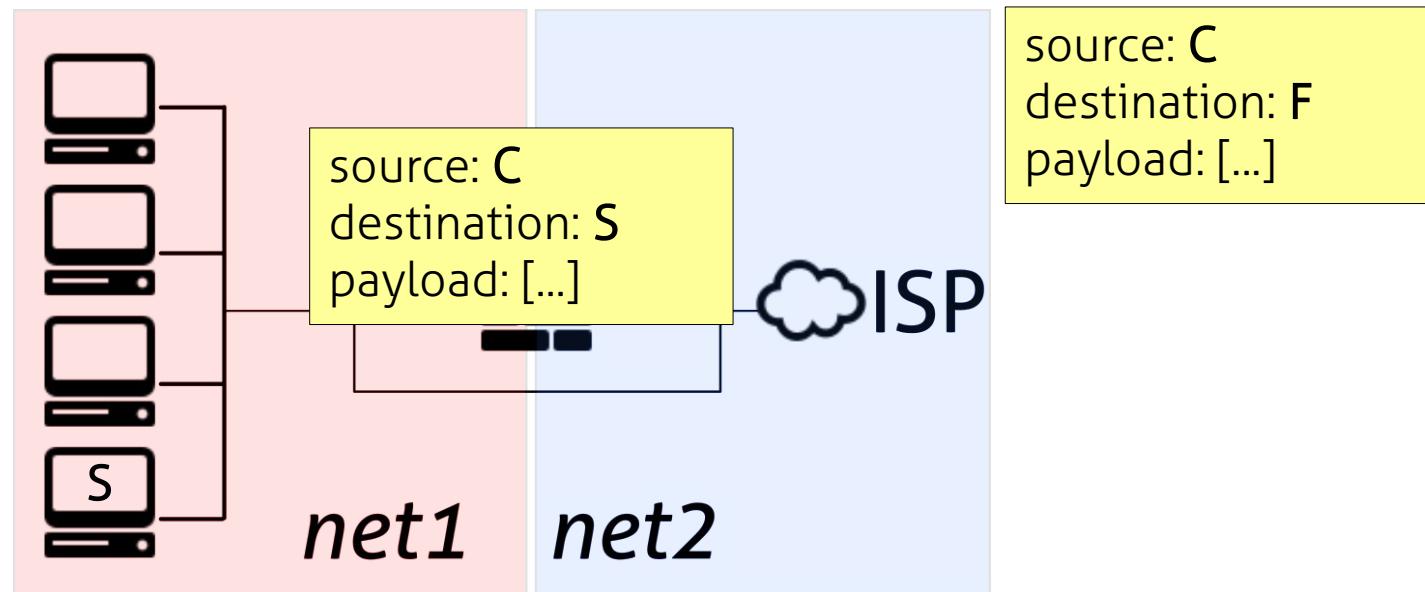
NAPT for Incoming Requests

- NAPT router blocks all incoming ports by default
- Many applications have had problems with NAPT in the past in their handling of incoming requests
- Four major methods
 - Application Level Gateways (ALGs)
 - Static port forwarding
 - Universal Plug and Play (UPnP) Internet Gateway Device (IGD) protocol
 - Traversal Using Relays around NAT (TURN)



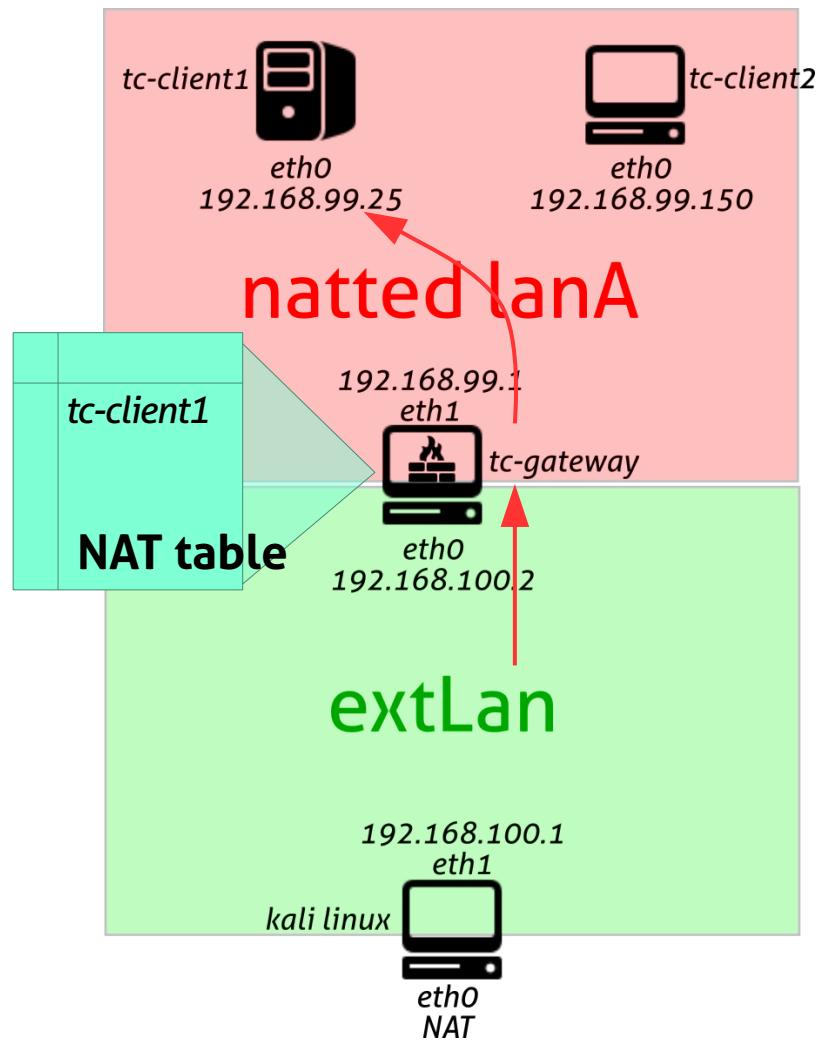
Destination NAT (DNAT)

- Enables servers located **inside** the firewall/router LAN to be accessed by clients located outside.
- The service appears to be hosted by the firewall/router



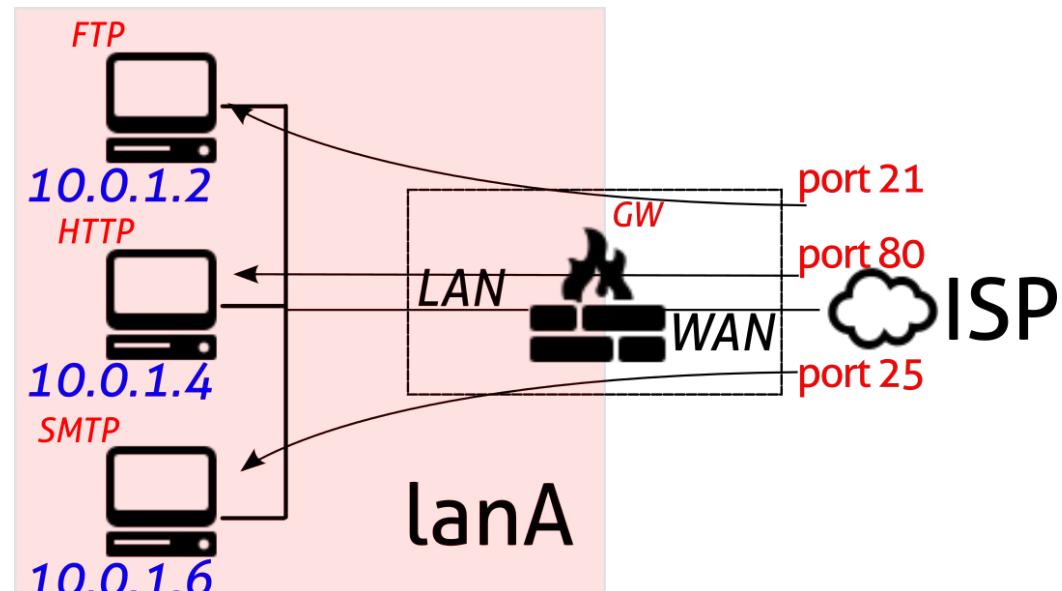
Destination NAT (DNAT)

- The firewall/router uses the NAT table to:
 - Translate incoming packets from the firewall/router WAN IP address to the internal address of the server
 - Forward the replies to the client service request from the internal server to the external client
 - Enables the client-server session or connection to continue on another port as requested by the internal server, forwarding any responses by the client to the server (the RELATED connections)



More on DNAT

- It is also called port forwarding or Virtual Server
- According to the port accessed from the external interface, the packets can be forwarded towards different internal hosts.
- Ex:
 - SMTP (port 25) to host A
 - HTTP (port 80) to host B
 - FTP (port 21) to host C





NAT port forwarding

Change port numbers on DNAT sessions, to enable an internal server to provide a particular service

- Can make use of different or differently-configured server programs to respond to internal LAN requests and to external WAN requests
- Example:
 - a host might be configured to provide outgoing SMTP service for the LAN on port 25 and incoming SMTP service on port 2525
 - the firewall will translate the port numbering for DNAT'ed incoming SMTP requests from 25 to 2525 and will also translate outgoing responses on this port intelligently



NAT pros and cons



NAT one-size-fits-all

- Artifacts of NAT devices values:
 - The need to establish state before anything gets through from outside to inside solves one set of problems
 - The expiration of state to stop receiving any packets when finished with a flow solves a set of problems
 - The ability for nodes to appear to be attached at the edge of the network solves a set of problems.
 - The ability to have addresses that are not publicly routed solves yet another set (mostly changes where the state is and scale requirements for the first one).



Perceived benefits of NAT and impact on IPv4 (RFC 4864)

- Simple Gateway between Internet and Private Network
- Simple Security Due to Stateful Filter Implementation
- User/Application Tracking
- Privacy and Topology Hiding
- Independent Control of Addressing in a Private Network
- Global Address Pool Conservation
- Multihoming and Renumbering with NAT



RFC 4864, IPv4 vs IPv6

Goal	IPv4	IPv6
Simple Gateway as default router and address pool manager	DHCP - single address upstream DHCP - limited number of individual devices downstream	DHCP-PD - arbitrary length customer prefix upstream SLAAC via RA downstream
Simple Security	Filtering side effect due to lack of translation state	Explicit Context Based Access Control (Reflexive ACL)
Local Usage Tracking	NAT state table	Address uniqueness
End-System Privacy	NAT transforms device ID bits in the address	Temporary use privacy addresses
Topology Hiding	NAT transforms subnet bits in the address	Untraceable addresses using IGP host routes /or MIPv6 tunnels
Addressing Autonomy	RFC 1918	RFC 3177 & 4193
Global Address Pool Conservation	RFC 1918 << 2^{48} application end points topology restricted	$17 * 10^{18}$ subnets $3.4 * 10^{38}$ addresses full port list / addr unrestricted topology
Renumbering and Multihoming	Address translation at border	Preferred lifetime per prefix & multiple addresses per interface



Applications not working with NAT (RFC 3027)

- Applications that have realm-specific (public or private) IP address information in payload
- Bundled session applications
- Peer-to-peer applications
- IP fragmentation with NAPT enroute
- Applications requiring retention of address mapping
- Applications requiring more public addresses than available
- Encrypted protocols like IPsec, IKE, Kerberos



Possible mitigation

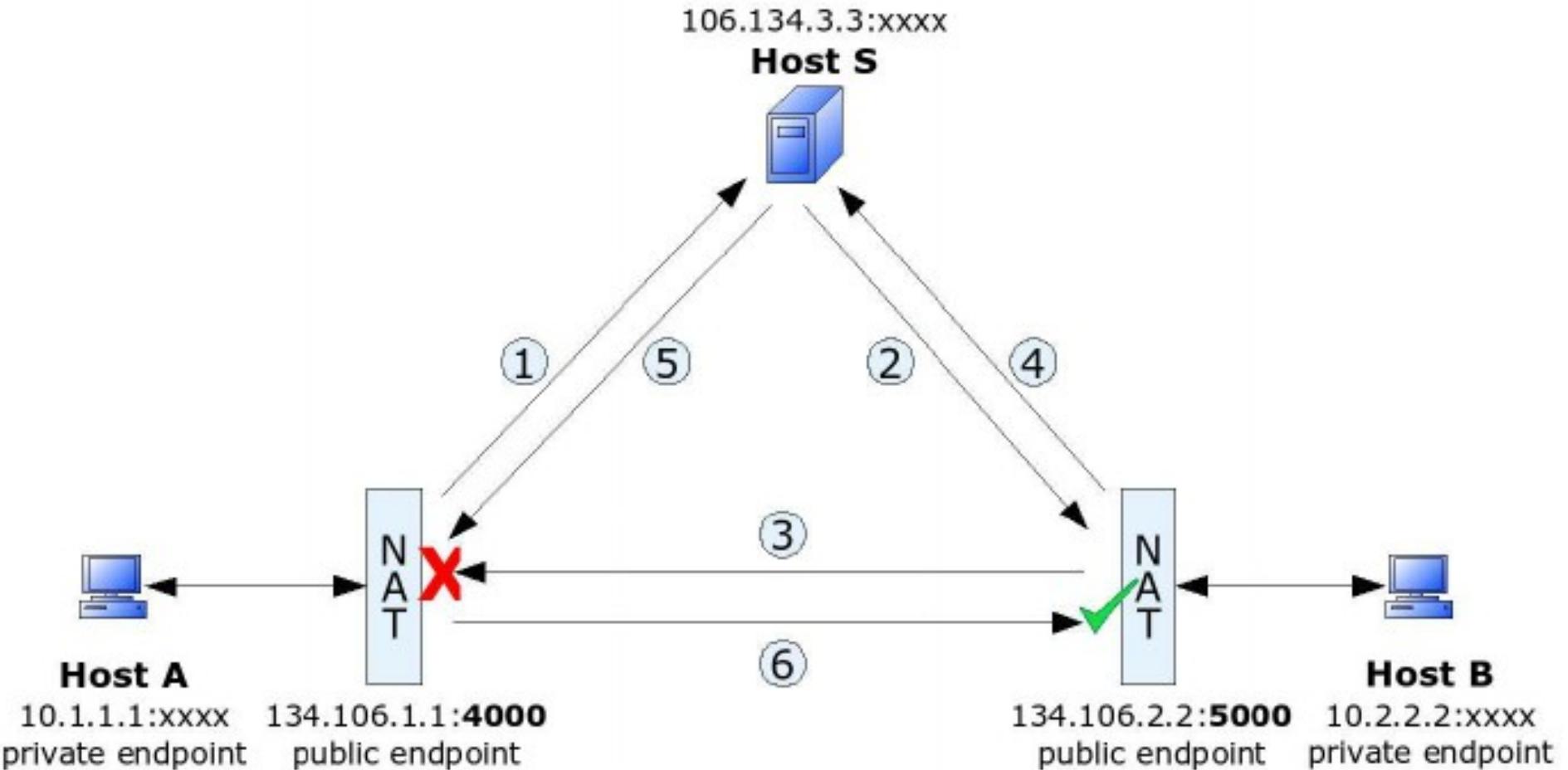
- Trivial: write applications that do not have IP info in the payload
- Use NAT in conjunction with Application Layer Gateways
 - Gateways specific for making NAT working with a given application
 - Ex: FTP, SIP, SMTP
 - Look for specific application traffic and perform IP/port translation also in the payload
- Use Interactive Connectivity Establishment (ICE)
 - Using STUN servers and TURN servers
- Unlikely to find a solution for encrypted application traffic



Hole punching

- A method for establishing bidirectional connections between Internet hosts, both in private networks using NAT
 - Working depends on the NAT implementation details
- Main idea:
 - find the public IP address of the other peer and initiate a connection to create a NAT state, so that replies can be correctly passed
- STUN: Session Traversal Utilities for NAT is used for this purpose, discovery and help two peers to communicate

Example of third party for NAT traversal





Example: UDP hole punching

- 1) Host A submits the request for communication with Host B to Host S
- 2) Host S submits the public endpoint (port 4000) of Host A to Host B
- 3) Host B sends a packet via its public-private endpoint (port 5000) to the public endpoint of Host A (4000).
 - 1) The NAT system of Host A rejects the packet, but a NAT session on B's side is now established ("a hole is punched"), ready to translate any packets from A's public endpoint (port 4000) to B's public endpoint (port 5000) into B's corresponding private endpoint
- 4) Host B notifies Host S and awaits an incoming connection from Host A
- 5) Host S submits the public endpoint of Host B (port 5000) to Host A
- 6) Host A uses its public-private endpoint (port 4000) to establish a connection with Host B's public endpoint (port 5000)

This technique appears to work with 82 percent UDP and 64 percent TCP NAT systems

B. Ford. *Peer-to-peer communication across network address translators*, 2005. USENIX Annual Technical Conference



iptables Tables and chains



Iptables fundamentals

- The rules are grouped in “tables”
- Each table has different “chains” of rules and different possible “targets” (packet fates)
- Each packet is subject to each rule of a chain
- Packet fates depend on the **first matching rule**
- To see chains and rules of the filter table

`iptables -L`

or for extended output

`iptables -L -n -v --line-numbers`

Filter table

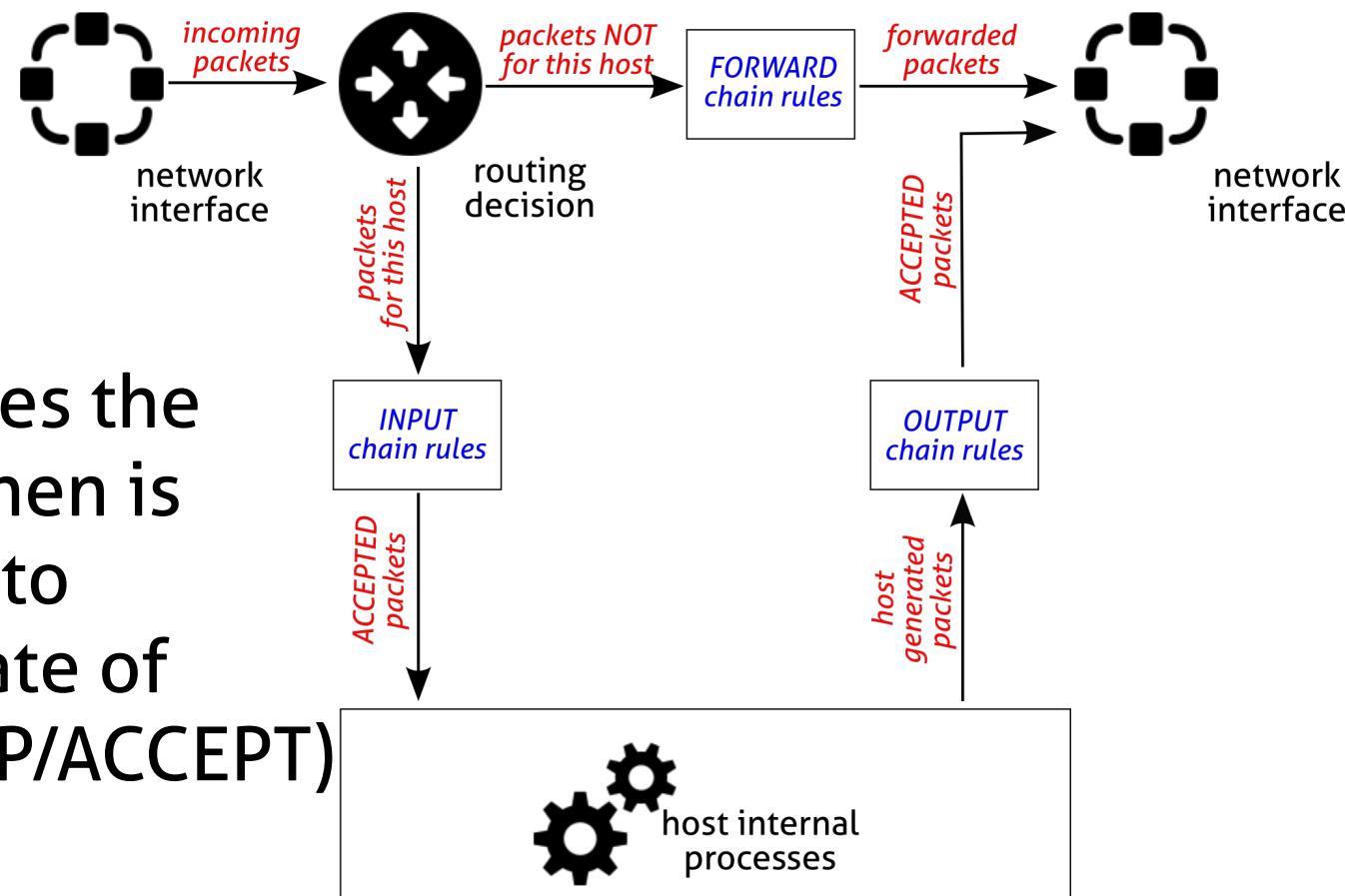
- Three built-in rule chains:

- INPUT

- OUTPUT

- FORWARD

- If a packet reaches the end of a chain, then is the chain policy to determine the fate of the packet (DROP/ACCEPT)





Useful iptables command switches

iptables switches	Description
-t table	Specifies the table (filter if absent)
-j target	Jump to the target (it can be another chain)
-A chain	Append a rule to the specified chain
-F	Flush a chain
-P policy	Change the default policy
-p protocol	Match the protocol type
-s ip-address	Match the source IP address
-d ip-address	Match the destination IP address
-p tcp --sport port	Match the tcp source port (also works for udp)
-p tcp --dport port	Match the tcp destination port (also works for udp)
-i interface-name	Match input interface (from which the packet enters)
-o interface-name	Match output interface (on which the packet exits)



Other useful iptables command switches

iptables switches	Description
-p tcp --sport port	Match the tcp source port
-p tcp --dport port	Match the tcp destination port
-p udp --sport port	Match the udp source port
-p udp --dport port	Match the udp destination port
--icmp-type type	Match specific icmp packet types
-m <i>module</i>	Uses an extension module
-m state --state s	Enable connection tracking. Match a packet which is in a specific state: NEW: the packet is the start of a new connection ESTABLISHED: the packet is part of an established connection RELATED: the packet is the starting of a related connection (like FTP data) INVALID: the packet could not be identified
-m multiport ...	Enable specification of several ports with one single rule

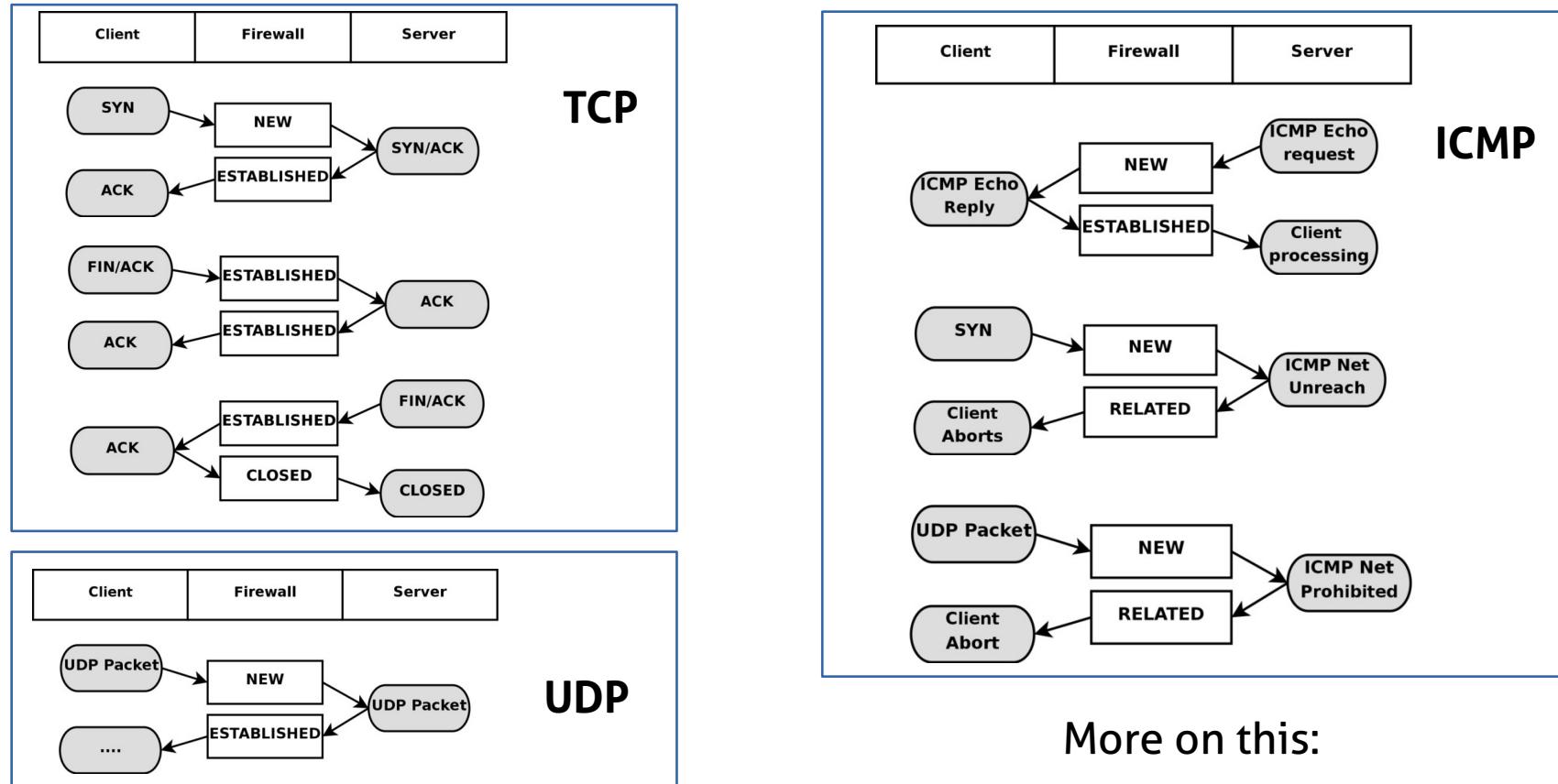


Even more useful iptables command switches

iptables switches	Description
-f	Match fragments which are not the first of a packet
--syn	Match packets which start a TCP connection It is equivalent to --tcp-flags SYN,RST,ACK SYN
-n	Used to avoid name substitution for IP hosts and ports
! (==not)	Used to negate the match. Can be used with many flags. Example: all the source addresses but a specific one -s ! ip_address
-m conntrack -ctstate...	Enable connection tracking (superset of state)
-m mac --mac-source	Match packets with a specific source MAC address
-m limit	Limit the number of packets for a specific time period --limit and --limit-burst iptables -A FORWARD -p tcp --syn -m limit --limit 1/s -j ACCEPT

More on the conntrack module

- Clever use of logic to recognize connections, even with connection-less protocols (UDP, ICMP...)



More on this:

<https://www.frozentux.net/iptables-tutorial/iptables-tutorial.html#STATEMACHINE>



iptables: four built-in tables

1.MANGLE: manipulate bits in TCP header

2.FILTER: packet filtering

3.NAT: network address translation

4.RAW: exceptions to connection tracking

- When present RAW table has the highest priority
- Used only for specific reasons
- Default: not loaded



MANGLE table

- Used for IP header manipulation (like ToS, TTL,...)
 - Should not be used for FILTERING
 - Should not be used for NAT
- Five chains to alter:
 - PREROUTING: incoming packets before a routing decision
 - INPUT: packets coming into the machine itself
 - OUTPUT: locally generated outgoing packets
 - FORWARD: packets being routed through the machine
 - POSTROUTING: packets immediately after the routing decision



FILTER table

- Used for filtering packets
- Three built-in rule chains:
 - INPUT: incoming packets intended for the filtering machine
 - OUTPUT: outgoing packets
 - FORWARD: for the packets that are not intended for this machine
 - The FORWARD chain only used when the machine is configured as a router (net.ipv4.ip_forward to 1 in the /etc/sysctl.conf file)



NAT table

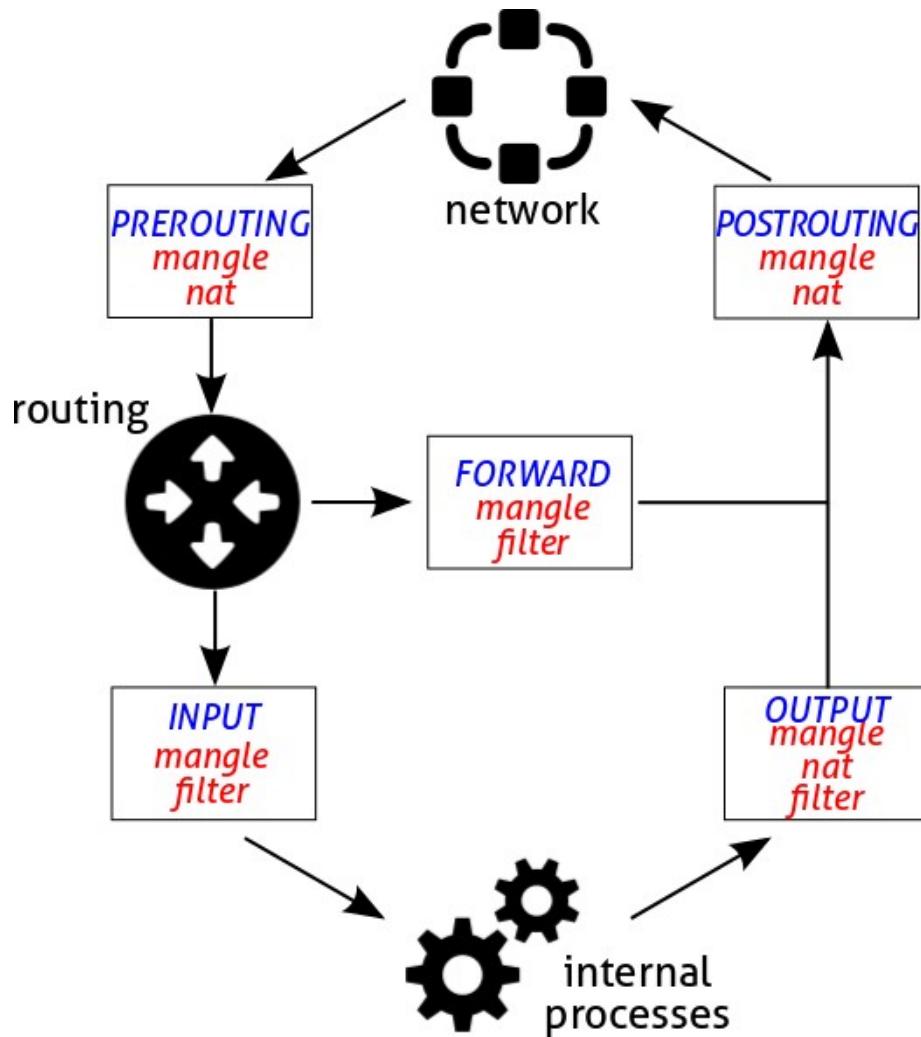
- Used for NAT (Network Address Translation): to translate the packet's source field or destination field
 - Only the first packet in a stream will hit this table (the rest of the packets will automatically have the same action)
- Special targets (*packet fates/actions*):
 - DNAT: destination nat
 - SNAT: source nat
 - MASQUERADE: dynamic nat (when fw interface address is dynamically assigned)
 - REDIRECT: redirects the packet to the machine itself



NAT'ing targets

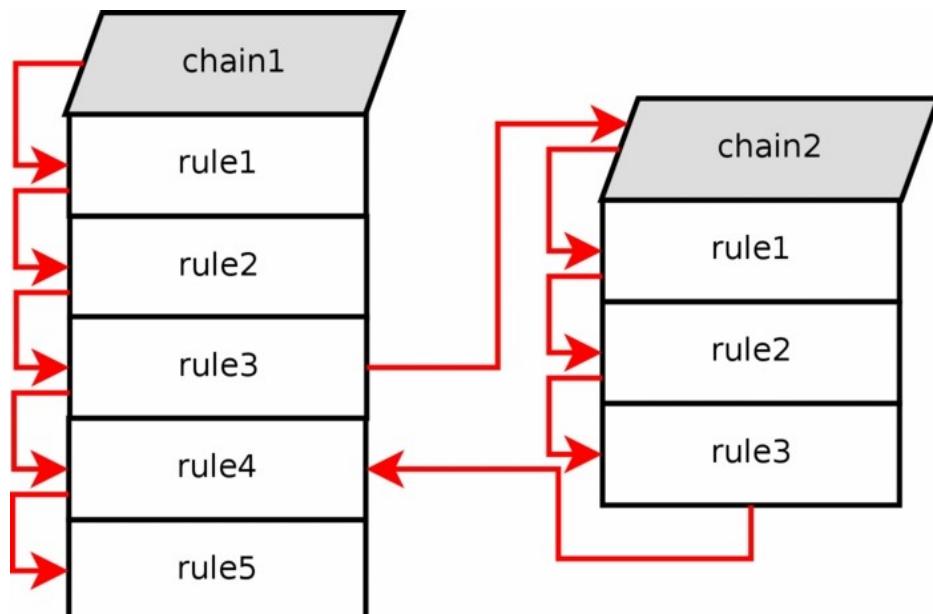
- DNAT: Destination address translation
 - Transform the destination IP of incoming packets
 - Used in PREROUTING chain
- SNAT: Source address translation
 - Transform the source IP of outgoing packets
 - Can be done one-to-one or many-to-one
 - Used in POSTROUTING chain
- MASQUERADE: like SNAT but the source IP is taken from the dynamically assigned address of the interface

Chain and table priorities



- MANGLE>NAT>FILTER
- RAW>MANGLE
 - Not shown in the picture
 - Only used during PREROUTING and OUTPUT

User defined chains



- It is possible to specify a jump rule to a different chain within the same table
- The new chain must be user specified
- If the end of the user specified chain is reached, the packet is sent back to the invoking chain



iptables logging

- LOG as possible target
 - "non-terminating target", i.e. rule traversal continues at the next rule
 - to log dropped packets, use the same DROP rule, but with LOG target
- When this option is set for a rule, the Linux kernel will print some information on all matching packets (like most IP header fields) via the kernel log (where it can be read with dmesg or syslogd(8))
 - log-level level: specifies the type of log (emerg, alert, crit, err, warning, notice, info, debug)
 - log-prefix prefix: add further information to the front of all messages produced by the logging action



Log example

- Log forwarded packets
 - `iptables -A FORWARD -p tcp -j LOG \ --log-level info --log-prefix "Forward INFO"`
- Log and drop invalid packets
 - `iptables -A INPUT -m conntrack --ctstate \ INVALID -j LOG --log-prefix "Invalid packet"`
 - `iptables -A INPUT -m conntrack --ctstate \ INVALID -j DROP`



That's all for today

- **Questions?**
- See you tomorrow
- Resources:
 - “Building internet firewalls”, Elizabeth D. Zwicky, Simon Cooper, D. Brent Chapman, O'Reilly 2nd ed.
 - https://docstore.mik.ua/orelly/networking_2ndEd/fire/index.htm
 - “Firewalls and Internet security: repelling the wily hacker”, William R. Cheswick, Steven M. Bellovin, Aviel D. Rubin, Addison-Wesley 2nd ed.
 - www.frozentux.net/iptables-tutorial/iptables-tutorial.html
 - Local Network Protection for IPv6 (<https://tools.ietf.org/html/rfc4864>)

Practical Network Defense

Master's degree in Cybersecurity 2021-22

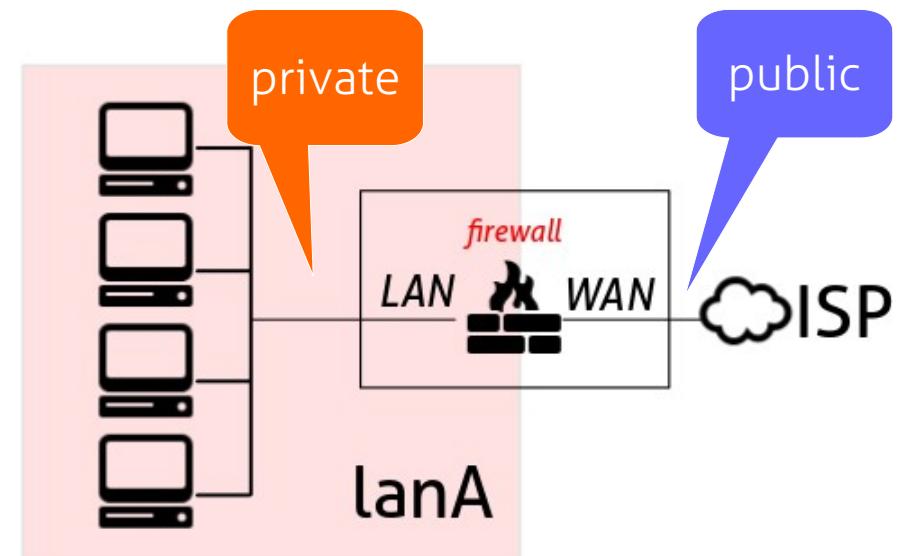
Iptables and NAT

Angelo Spognardi
[*spognardi@di.uniroma1.it*](mailto:spognardi@di.uniroma1.it)

*Dipartimento di Informatica
Sapienza Università di Roma*

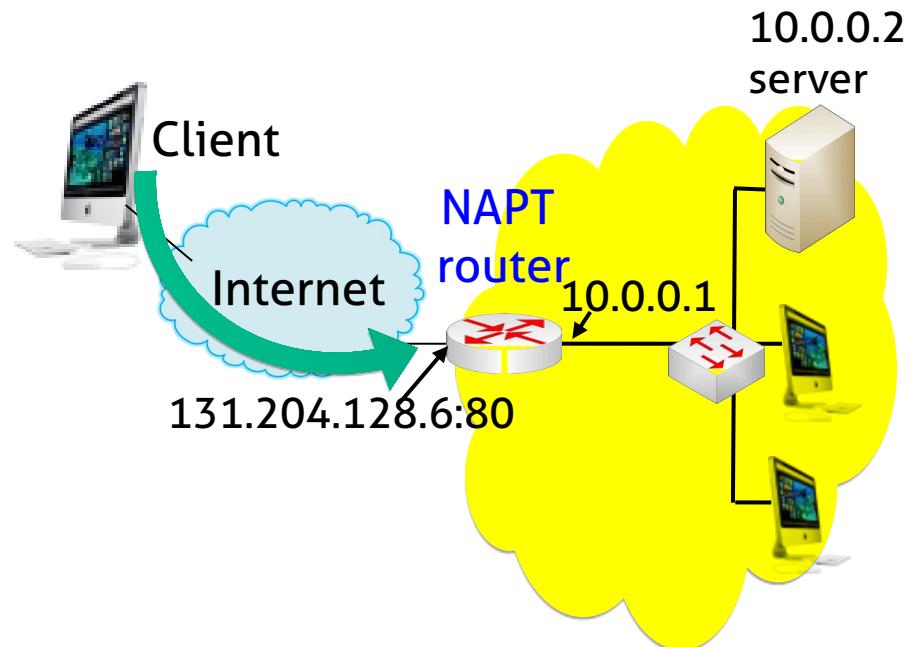
Network Address Translation (NAT)

- Translate the address (f.e.: between incompatible IP addressing)
- Informally speaking, connecting to the Internet a LAN using un-routable in-house LAN addresses
- NAT in a routed firewall:
 - Can filter requests from hosts on WAN side to hosts on LAN side
 - Allows host requests from the LAN side to reach the WAN side
 - Does not expose LAN hosts to external port scans



NAPT for Incoming Requests

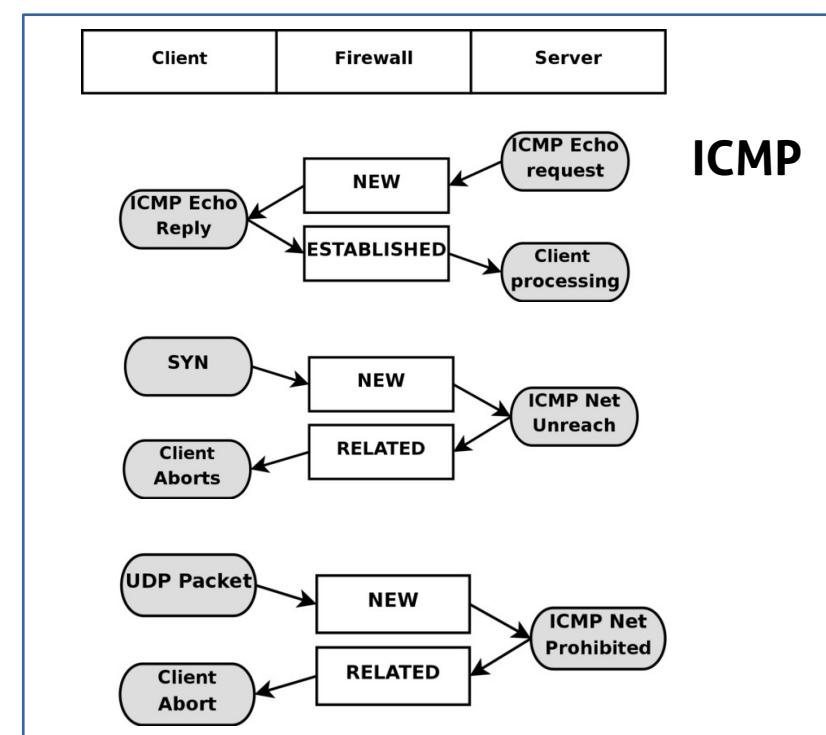
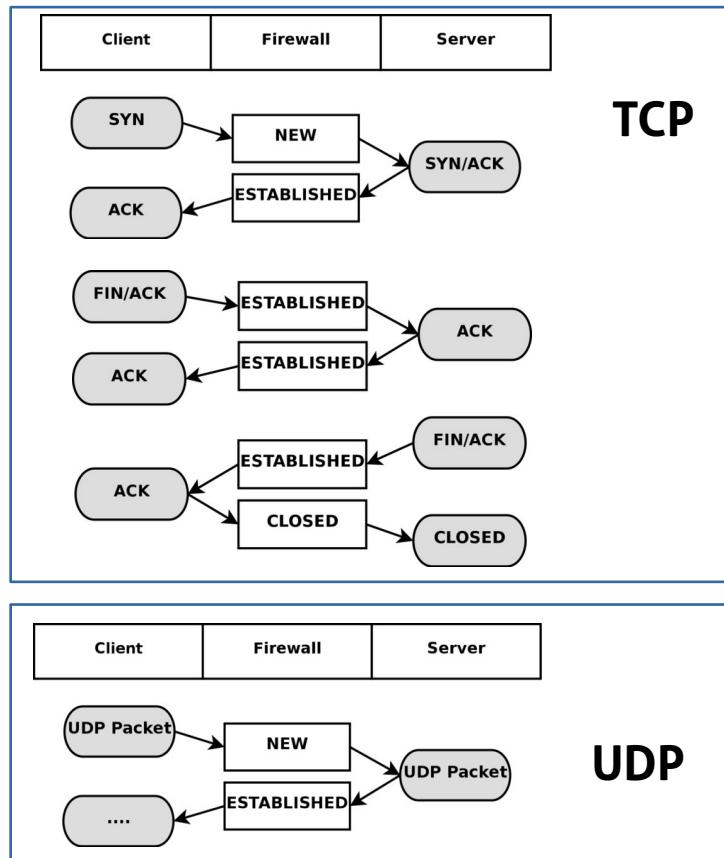
- NAPT router blocks all incoming ports by default
- Many applications have had problems with NAPT in the past in their handling of incoming requests
- Four major methods
 - Application Level Gateways (ALGs)
 - Static port forwarding
 - Universal Plug and Play (UPnP) Internet Gateway Device (IGD) protocol
 - Traversal Using Relays around NAT (TURN)





More on the conntrack module

- Clever use of logic to recognize connections, even with connection-less protocols (UDP, ICMP...)



More on this:

<https://www.frozenthux.net/iptables-tutorial/iptables-tutorial.html#STATEMACHINE>



iptables: four built-in tables

1.MANGLE: manipulate bits in TCP header

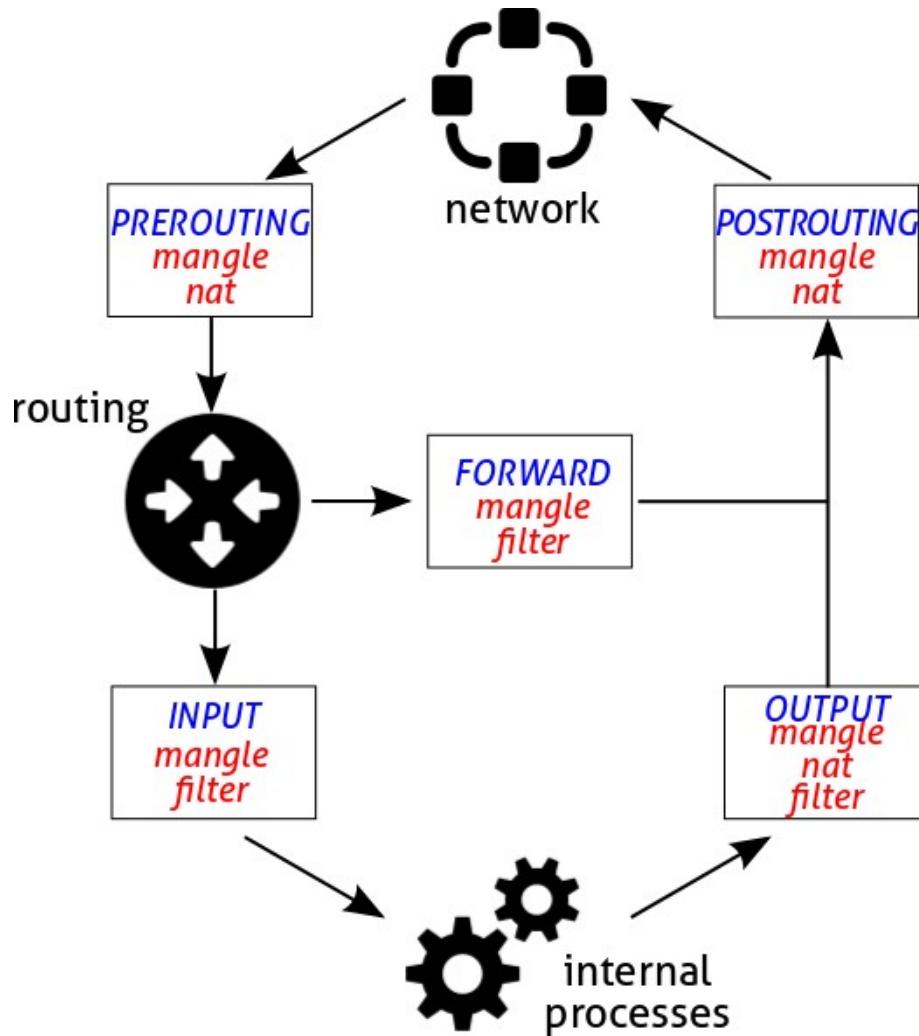
2.FILTER: packet filtering

3.NAT: network address translation

4.RAW: exceptions to connection tracking

- When present RAW table has the highest priority
- Used only for specific reasons
- Default: not loaded

Chain and table priorities



- MANGLE>NAT>FILTER
- RAW>MANGLE
 - Not shown in the picture
 - Only used during PREROUTING and OUTPUT



NAT table

- Used for NAT (Network Address Translation): to translate the packet's source field or destination field
 - Only the first packet in a stream will hit this table (the rest of the packets will automatically have the same action)
- Special targets (*packet fates/actions*):
 - DNAT: destination nat
 - SNAT: source nat
 - MASQUERADE: dynamic nat (when fw interface address is dynamically assigned)
 - REDIRECT: redirects the packet to the machine itself



NAT'ing targets

- DNAT: Destination address translation
 - Transform the destination IP of incoming packets
 - Used in PREROUTING chain
- SNAT: Source address translation
 - Transform the source IP of outgoing packets
 - Can be done one-to-one or many-to-one
 - Used in POSTROUTING chain
- MASQUERADE: like SNAT but the source IP is taken from the dynamically assigned address of the interface



iptables logging

- LOG as possible target
 - "non-terminating target", i.e. rule traversal continues at the next rule
 - to log dropped packets, use the same DROP rule, but with LOG target
- When this option is set for a rule, the Linux kernel will print some information on all matching packets (like most IP header fields) via the kernel log (where it can be read with dmesg or syslogd(8))
 - log-level level: specifies the type of log (emerg, alert, crit, err, warning, notice, info, debug)
 - log-prefix prefix: add further information to the front of all messages produced by the logging action



Log example

- Log forwarded packets
 - `iptables -A FORWARD -p tcp -j LOG \ --log-level info --log-prefix "Forward INFO"`
- Log and drop invalid packets
 - `iptables -A INPUT -m conntrack --ctstate \ INVALID -j LOG --log-prefix "Invalid packet"`
 - `iptables -A INPUT -m conntrack --ctstate \ INVALID -j DROP`



Lab activity



Main tasks

- Iptables and ip6tables
- Reference links:
 - Linux ipv6 configuration: ipv6 sysctl
 - <https://www.kernel.org/doc/Documentation/networking/ip-sysctl.txt>
 - Iptables reference manual
 - www.frozentux.net/iptables-tutorial/iptables-tutorial.html



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

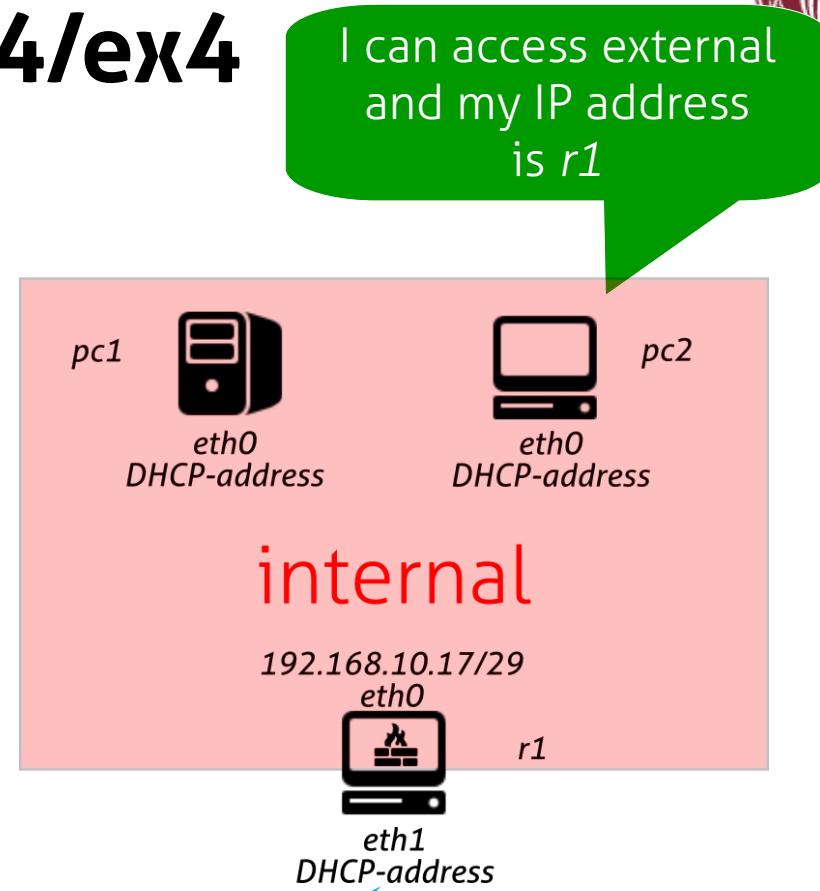
- It should work in both Virtualbox and VMware
- It should work in Linux, Windows and MacOS
- There are some alias (shortcuts) prepared for you
 - Check with `alias`
- All the exercises can be found in the git repository:
 - <https://github.com/vitome/pnd-labs.git>
- You can move in the directory and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



Lab activity: ex4

Exercise 1: pnd-labs/lab4/ex4

- Enable masquerade
- Setup r1 to perform NATting with iptables
 - Masquerade to exit
- internal is NOT exposed





Exercise 1: Policy to protect r1

- Accept ICMP echo replies destined to LAN
- Only accept ICMP echo request from eth1
- Respond with TCP RST or ICMP Unreachable for incoming requests for blocked ports

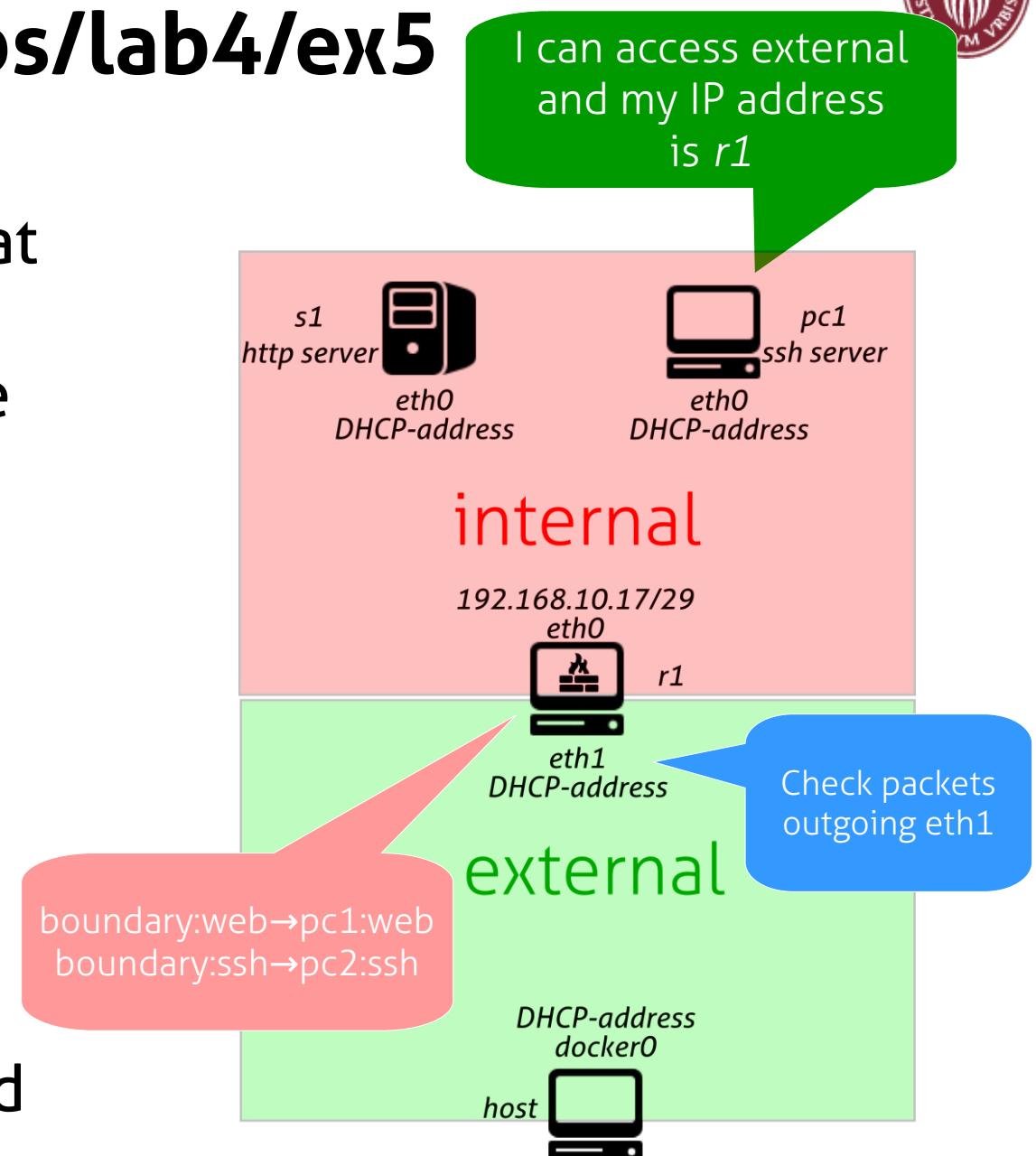


Lab activity: ex5



Exercise 2: pnd-labs/lab4/ex5

- Modify activity 1 so that internal servers are reachable from outside
 - http on s1
 - ssh on pc1
- Setup boundary to perform NATting with iptables
 - Destination NAT
- internal is NOT exposed



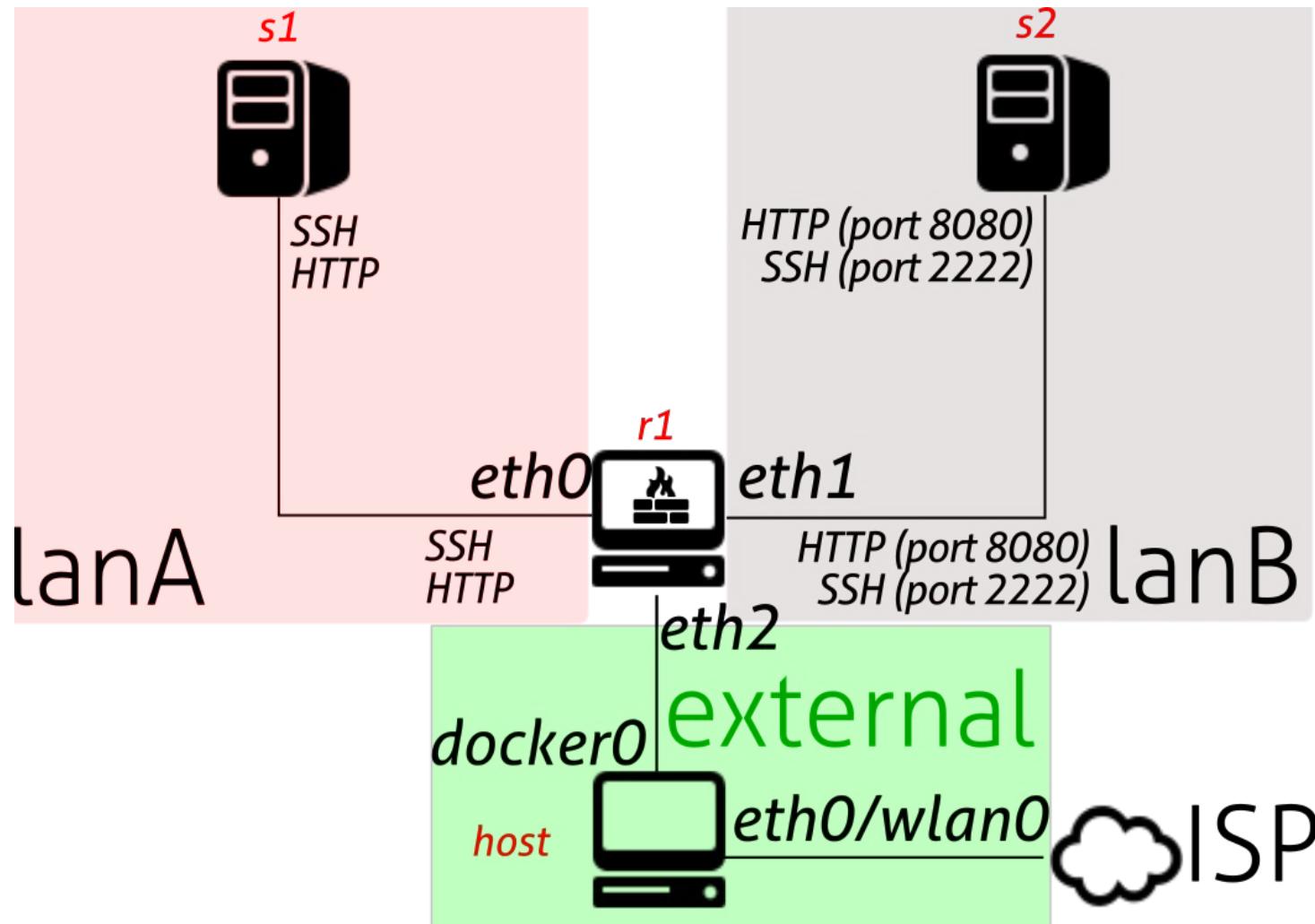


Lab activity: ex6



Exercise 3: pnd-labs/lab4/ex6

NAT with 2 networks and services





Exercise 3: policy to implement

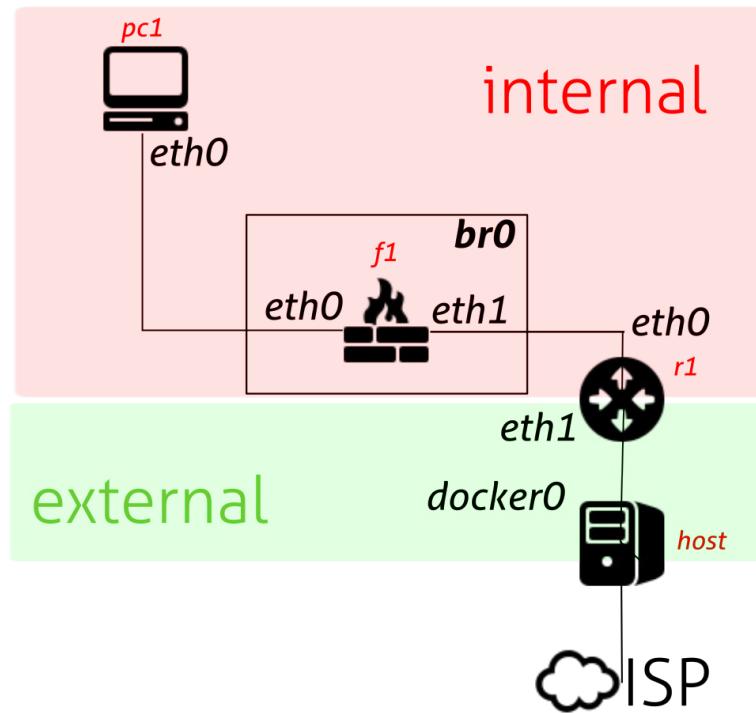
- Unrestricted internet access from all the machines in the lanA and lanB
- Use NAT to redirect incoming traffic from WAN to the all the services
 - SSH
 - HTTP and HTTPS
- Accept ICMP echo response also for both the lans
- Respond with TCP RST or ICMP Unreachable for incoming requests for blocked ports



Lab activity: ex7

Exercise 4: pnd-labs/lab4/ex7

Transparent firewall



- The lab is ready to have *f1* to act as a transparent firewall
- Try to configure it so that you can regulate the type of traffic *pc1* can use towards the ISP and the host



That's all for today

- **Questions?**
- See you tomorrow
- Resources:
 - “Building internet firewalls”, Elizabeth D. Zwicky, Simon Cooper, D. Brent Chapman, O'Reilly 2nd ed.
 - https://docstore.mik.ua/orelly/networking_2ndEd/fire/index.htm
 - “Firewalls and Internet security: repelling the wily hacker”, William R. Cheswick, Steven M. Bellovin, Aviel D. Rubin, Addison-Wesley 2nd ed.
 - www.frozentux.net/iptables-tutorial/iptables-tutorial.html

Practical Network Defense

Master's degree in Cybersecurity 2021-22

Link-local attacks

Angelo Spognardi
[*spognardi@di.uniroma1.it*](mailto:spognardi@di.uniroma1.it)

*Dipartimento di Informatica
Sapienza Università di Roma*



Agenda

- Network eavesdropping
- ARP poisoning
- IPv6 Neighbor Discovery threats
- IPv6 Rogue RA (or RA spoofing)
- Rogue DHCP

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 sniffer (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?

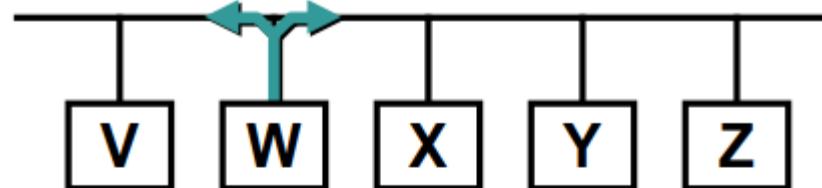
- It depends...
 - Switch starts flooding (attack success)
 - Switch freezes (denial of service)
 - Switch crash (denial of service)
- Today not really effective: port security in switches
 - Allows you to specify MAC addresses for each port, or to learn a certain number of MAC addresses per port
 - 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

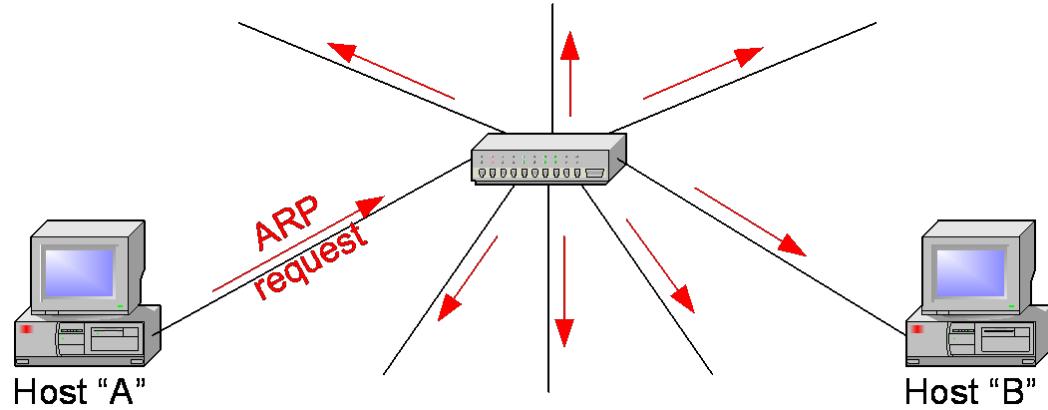
- An ARP request message should be placed in a frame and broadcast to all computers on the network
- Each computer receives the request and examines the IP address
- The computer mentioned in the request sends a response; all other computers process and discard the request without sending a response





ARP 1

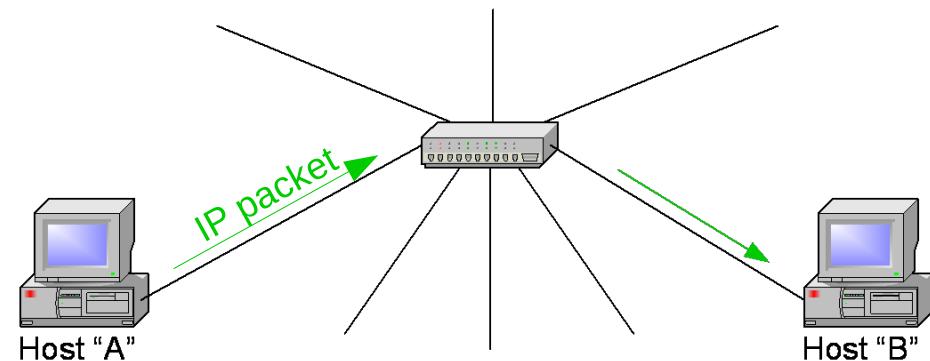
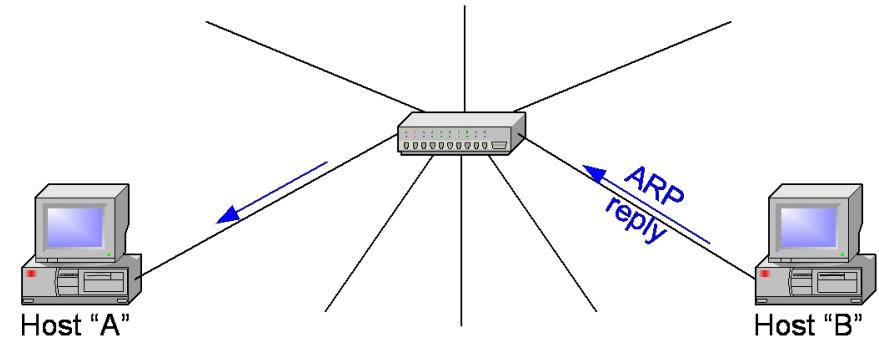
- Host A has the IP address of host B
- It knows it is in the same network → has to use Ethernet
- It needs to know the MAC address of host B
- It broadcasts an ARP request for IP of host B (MAC Dest = ff:ff:ff:ff:ff:ff)





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





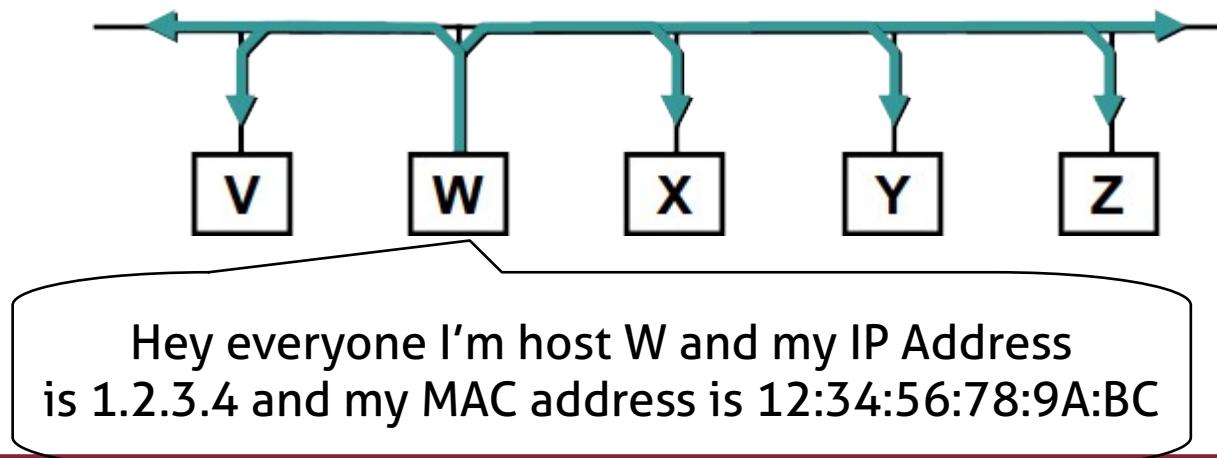
ARP table

- 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

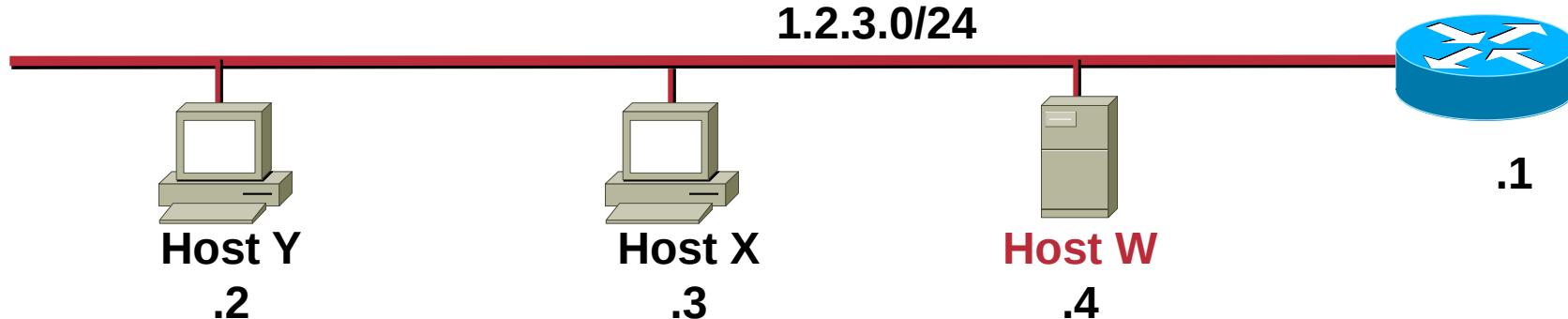
- Gratuitous ARP is used by hosts to “announce” their IP address to the local network and avoid duplicate IP addresses on the network
- Routers and other network hardware may use cache information gained from gratuitous ARP responses
- Gratuitous ARP is a broadcast packet (like an ARP-request)





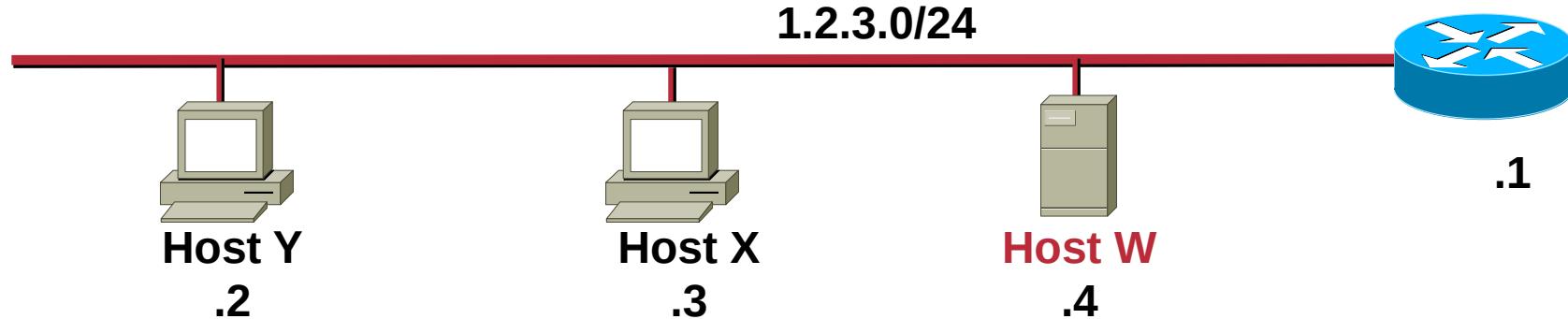
Misuse of Gratuitous ARP

- 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

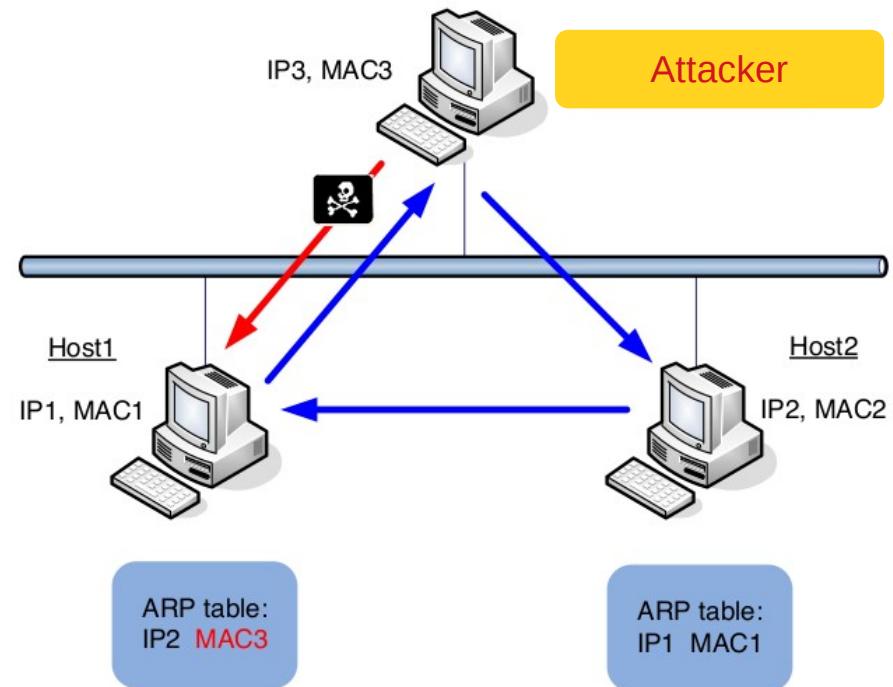
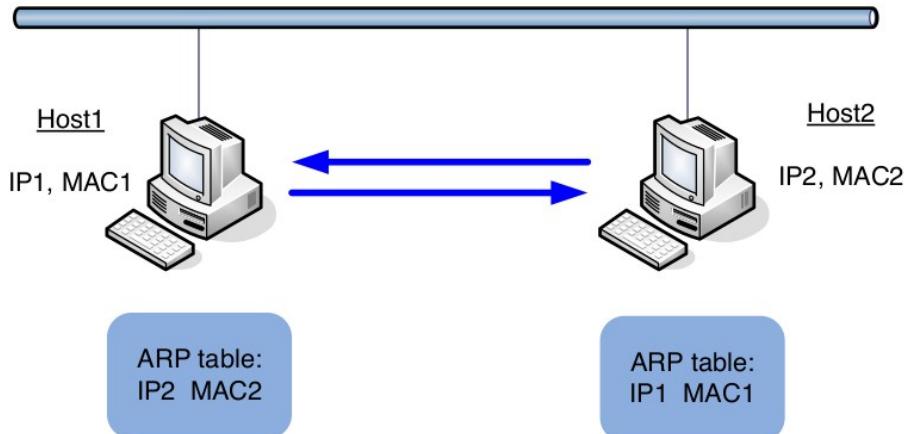


Hijacking in the local network

- 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?

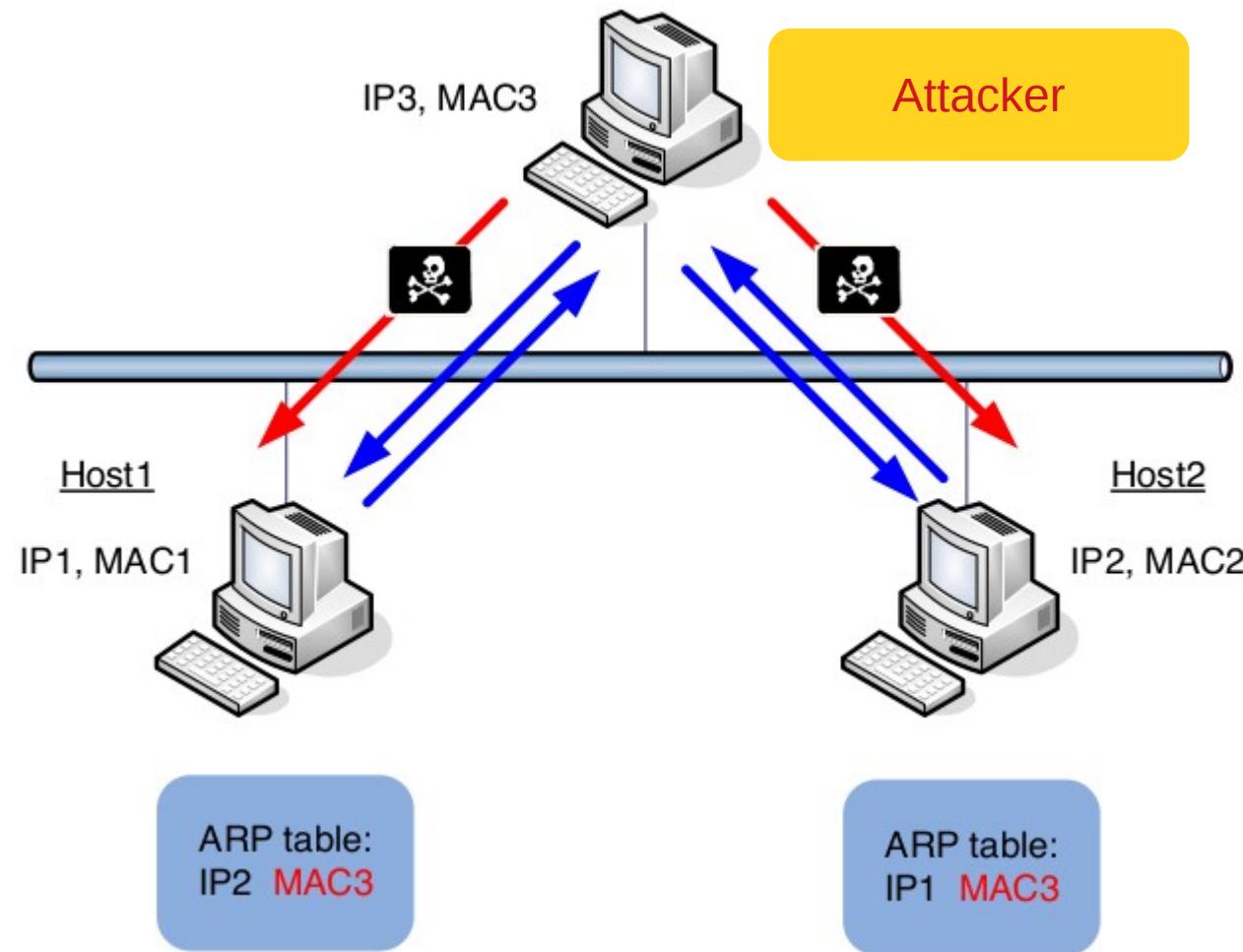


Man-in-the-middle 1





Man-in-the-middle 2



IPv6 Neighbor Discovery

Address Resolution: IPv4 and IPv6



Address Resolution: IPv4 and IPv6



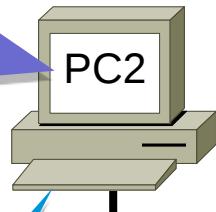
IPv4: ARP over Ethernet

ARP Request: Broadcast

Ethernet

ARP Request/Reply

My IPv4!
Here is the
MAC?



2

ARP Reply

ARP
Cache

Know IPv4,
what is the
MAC?

ARP Request

1

My IPv6!
Here is the
MAC?

2

Neighbor
Advertisement

Neighbor
Cache

Know IPv6,
what is the
MAC?

Neighbor
Solicitation

1

IPv6: ICMPv6 over IPv6 over Ethernet

NS: Multicast

NS: Solicited Node Multicast

Ethernet

IPv6 Header

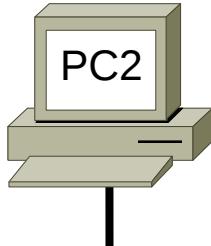
ICMPv6: Neighbor Solicitation/Advertisement

Neighbor Solicitation and Neighbor Advertisement



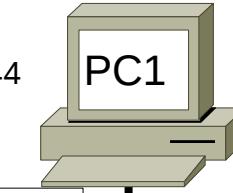
2001:DB8:CAFE:1::200/64

FF02::1:FF00:200 (Solicited Node Multicast)



MAC Address
00-1B-24-04-A2-1E

2001:DB8:CAFE:1::100/64



MAC Address
00-21-9B-D9-C6-44

1

PC1> ping 2001:DB8:CAFE:1::200

2 5



Neighbor
Solicitation

4

Neighbor
Advertisement

Neighbor Cache
<empty until step 5>

NS: Multicast

NS: Solicited Node Multicast

Ethernet

IPv6 Header

ICMPv6: Neighbor Solicitation/Advertisement

NA: Unicast

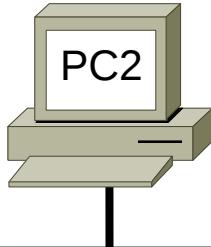
NA: Unicast

Neighbor Solicitation



2001:DB8:CAFE:1::200/64

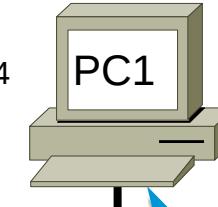
FF02::1:FF00:200 (Solicited Node Multicast)



MAC Address
00-1B-24-04-A2-1E

2001:DB8:CAFE:1::100/64

MAC Address
00-21-9B-D9-C6-44



Neighbor Cache

Neighbor
Solicitation

I know the
IPv6, but what
is the MAC?

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**

Mapped multicast address for PC2

Next header is an ICMPv6 header

Global unicast address of PC1

Solicited-node multicast address of PC2

Internet Control Message Protocol v6

Type: 135 (Neighbor solicitation)

Code: 0

Checksum: 0xbbbab [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

Neighbor Solicitation message

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

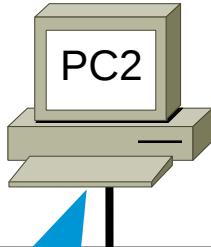
MAC address of the sender, PC1

Neighbor Advertisement



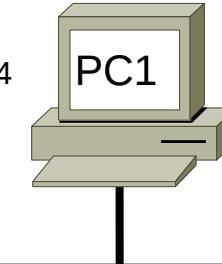
2001:DB8:CAFE:1::200/64

FF02::1:FF00:200 (Solicited Node Multicast)



MAC Address
00-1B-24-04-A2-1E

2001:DB8:CAFE:1::100/64



MAC Address
00-21-9B-D9-C6-44

It's my IPv6
and here is
my MAC?

Neighbor
Advertisement

Neighbor Cache

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)

Next header is an ICMPv6 header

Hop limit: 255

Source: 2001:db8:cafe:1::200

Global unicast address of PC2

Destination: 2001:db8:cafe:1::100

Global unicast address of PC1

Internet Control Message Protocol v6

Type: 136 (Neighbor advertisement)

Neighbor Advertisement message

Code: 0

Checksum: 0xb4d [correct]

Flags: 0x60000000

Target: 2001:db8:cafe:1::200

IPv6 address of the sender, PC2

ICMPv6 Option (Target link-layer address)

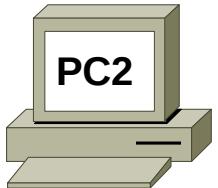
Type: Target link-layer address (2)

Length: 8

Link-layer address: 00:1b:24:04:a2:1e

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



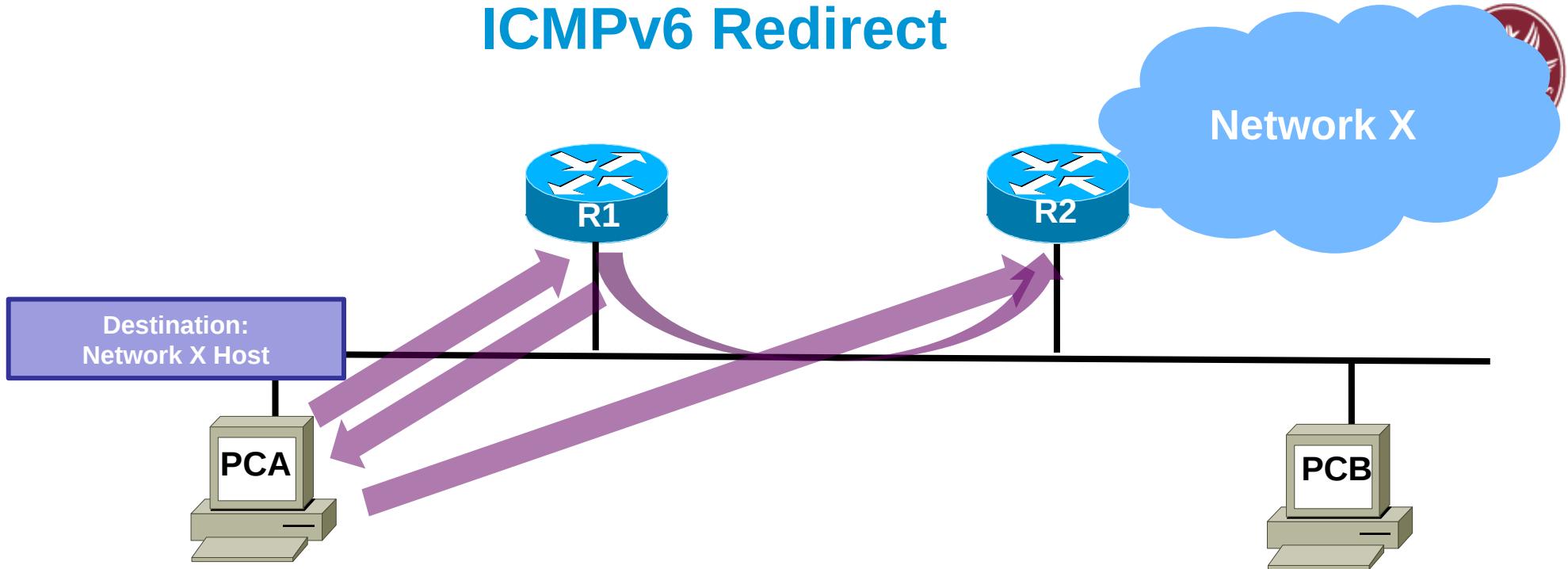
Neighbor Solicitation

See the process with:
R1# debug ipv6 nd

**Hopefully no
Neighbor Advertisement**

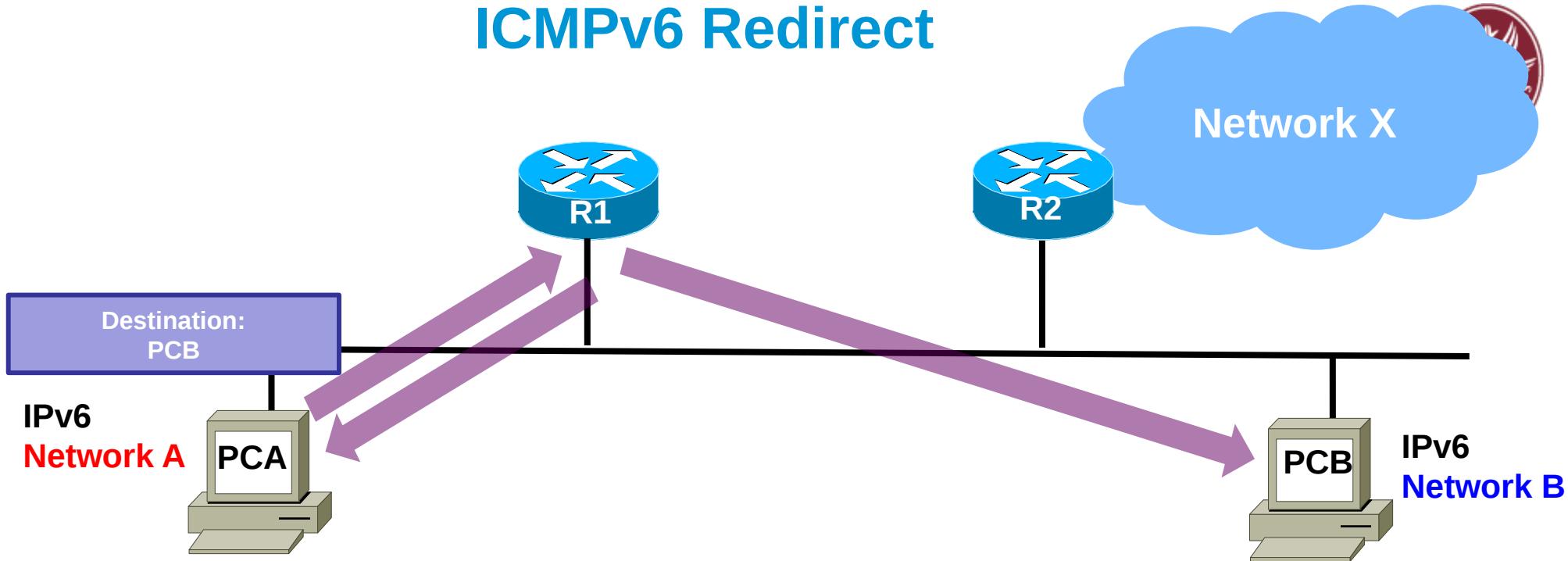
- Duplicate Address Detection (DAD) is used to guarantee that an IPv6 unicast address is unique on the link.
- A device will send a Neighbor Solicitation for its own unicast address (static or dynamic).
- After a period of time, if a NA is not received, then the address is deemed unique.
- Once required,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

ICMPv6 Redirect



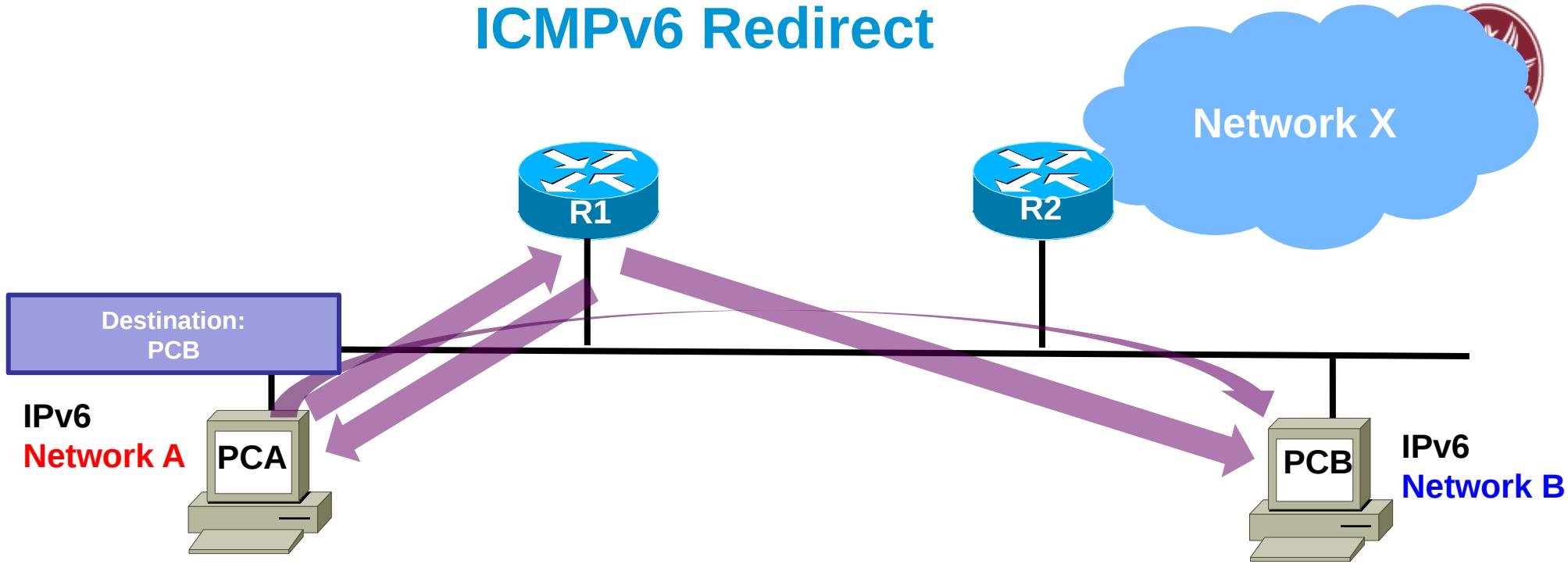
- Similar functionality as ICMPv4.
- Like IPv4, a router informs an originating host of the IP address of a router that is on the local link and is closer to the destination.

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.



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)



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 Advertisements
 - 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)
```

A problem has been detected and Windows has been shutdown to prevent damage to your computer.
DRIVER_INQL_NOT_LESS_OR_EQUAL
If this is the first time you've seen this stop error screen, restart your computer. If this screen appears again, follow these steps:
Check to make sure any new hardware or software is properly installed. If this is a new installation, ask your hardware or software manufacturer for any Windows updates you might need.
If problems continue, disable or remove any newly installed hardware or software. Disable BIOS memory options such as caching or shadowing. If you need to use Safe Mode to remove or disable components, restart your computer, press F8 to select Advanced Startup Options, and then select Safe Mode.

Technical information:

*** STOP: 0x00000001 (0x0000000C,0x00000002,0x00000000,0xF8685A89)
*** gv3.sys - Address F8685A89 base at F8685000, timestamp 3dd99194b

Beginning dump of physical memory

Physical memory dump complete.

Contact your system administrator or technical support group for further assistance.

:(
:(

Your PC ran into a problem and needs to restart. We're just collecting some error info, and then we'll restart for you. (0% complete)

Please feel free to provide us with any additional information for the error code. We're available 24/7.



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:
 - 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



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>



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_InternetProtocolVersion6IPv6IPNextGenerationPng.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)

Practical Network Defense

Master's degree in Cybersecurity 2021-22

Link-local ARP attacks: lab

Angelo Spognardi
[*spognardi@di.uniroma1.it*](mailto:spognardi@di.uniroma1.it)

*Dipartimento di Informatica
Sapienza Università di Roma*



Lab activity



Main tasks

- Network eavesdropping
- ARP poisoning
 - MITM
- Reference links:
 - <https://developers.redhat.com/blog/2018/10/22/introduction-to-linux-interfaces-for-virtual-networking/>
 - <https://www.fir3net.com/Networking/Terms-and-Concepts/virtual-networking-devices-tun-tap-and-veth-pairs-explained.html>
 - <https://www.ettercap-project.org/>
 - <https://pentestmag.com/ettercap-tutorial-for-windows/>



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>
- You can move in the directory and run `lstart`
 - **NOTE:** launch docker first or the first `lstart` attempt can (...will...) fail



Lab activity: ex1



Exercise 1: pnd-labs/lab3/ex1

- A lan with a host, a server and a router
- Join the network and eavesdrop the traffic in kathara
- The assignment is to create a virtual interface in the hosting machine and join the internal network A
- This can be done in several ways (using the bridge interface or creating a virtual interface to be joined to the actual bridge)
- Once done sniff the traffic
 - Create some traffic between the host and the server. Can you see the traffic from the hosting machine (the lubuntu box)?



Some hints about virtual interfaces

- add (or del) a virtual interface (pair veth0@veth1):
 - `ip link add dev veth0 type veth peer name veth1`
- connect one veth end to the virtual bridge:
 - `ip link set master br0 dev veth1`
- assign an IP address to the other end (not enslaved):
 - `ip addr add x.x.x.x/y dev veth0`
- enable both the ends of the virtual interface
 - `ip link set veth0 up`
 - `ip link set veth1 up`



Lab activity: ex2

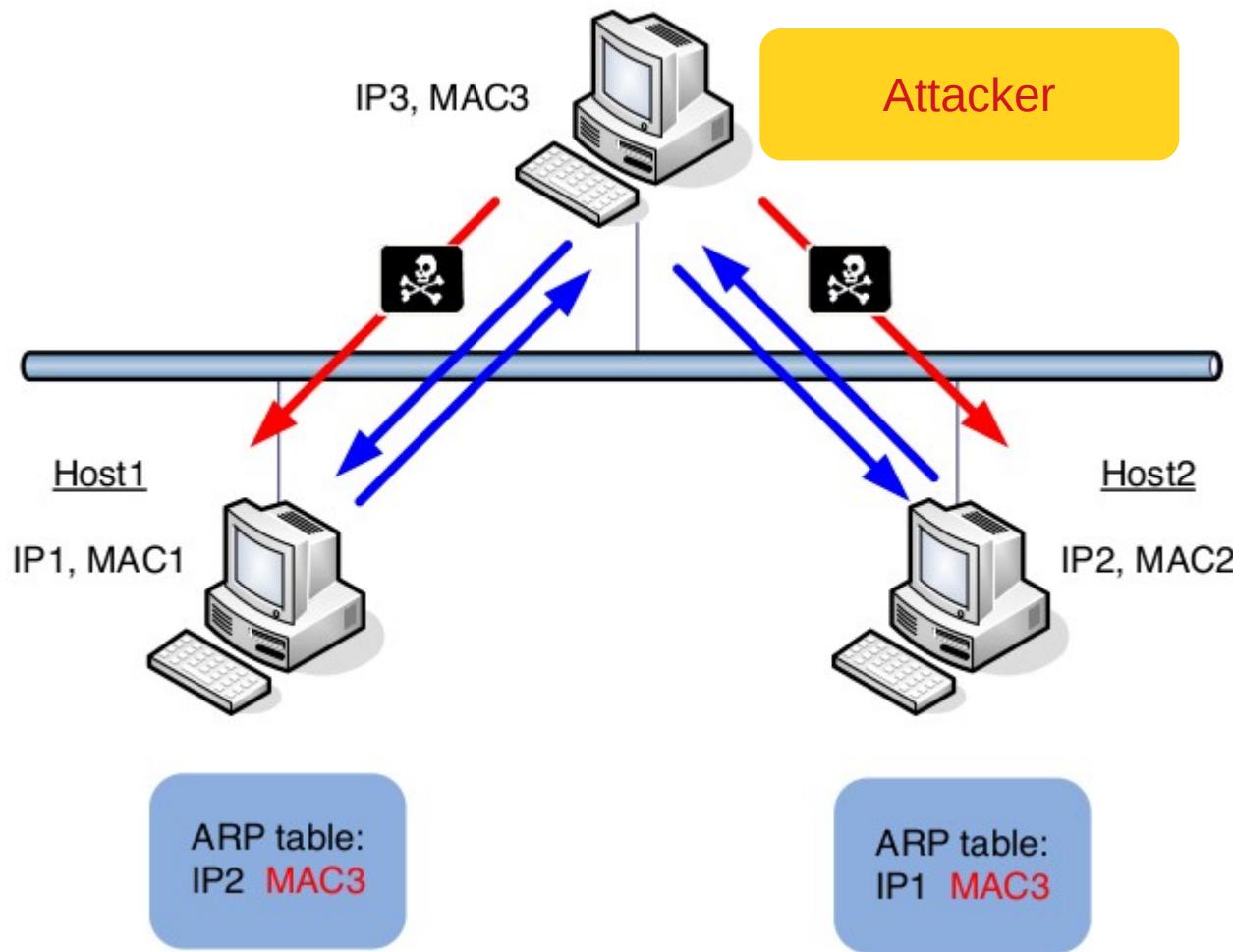


Exercise 2: pnd-labs/lab3/ex2

- 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-setu
p-caplets](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



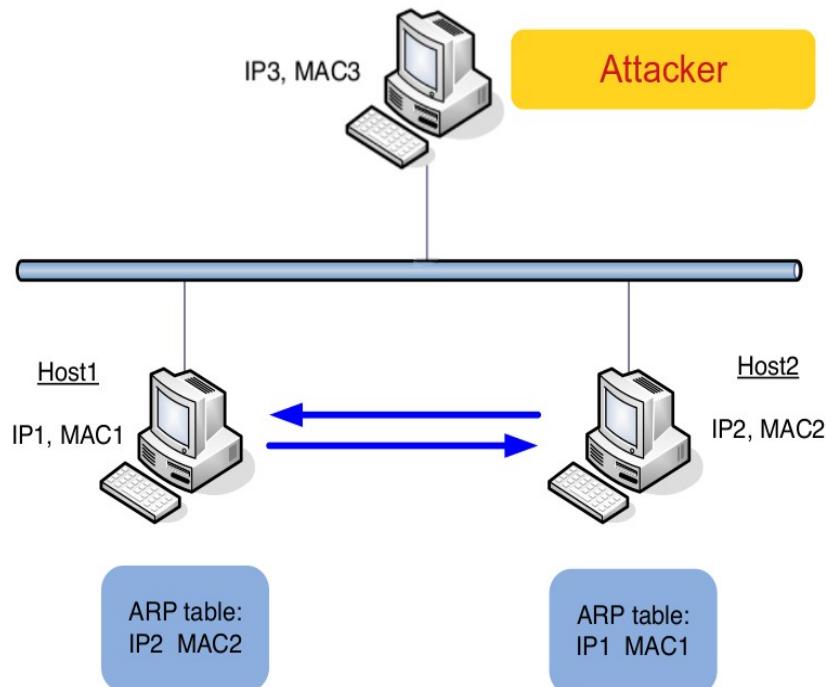


Lab activity: ex3



Exercise 3: pnd-labs/lab3/ex3

- A lan with a host, a server and a router
- Compile the thc toolkit and play with the tools
- If you join the network from the hosting box, you can also monitor the traffic with wireshark





That's all for today

- **Questions?**
- References:
- IPv6 security references:
<https://www.ripe.net/support/training/material/ipv6-security/ipv6security-references.pdf>
 - 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)

Practical Network Defense

Master's degree in Cybersecurity 2021-22

Link-local attacks: ICMP redirect lab

Angelo Spognardi
[*spognardi@di.uniroma1.it*](mailto:spognardi@di.uniroma1.it)

*Dipartimento di Informatica
Sapienza Università di Roma*



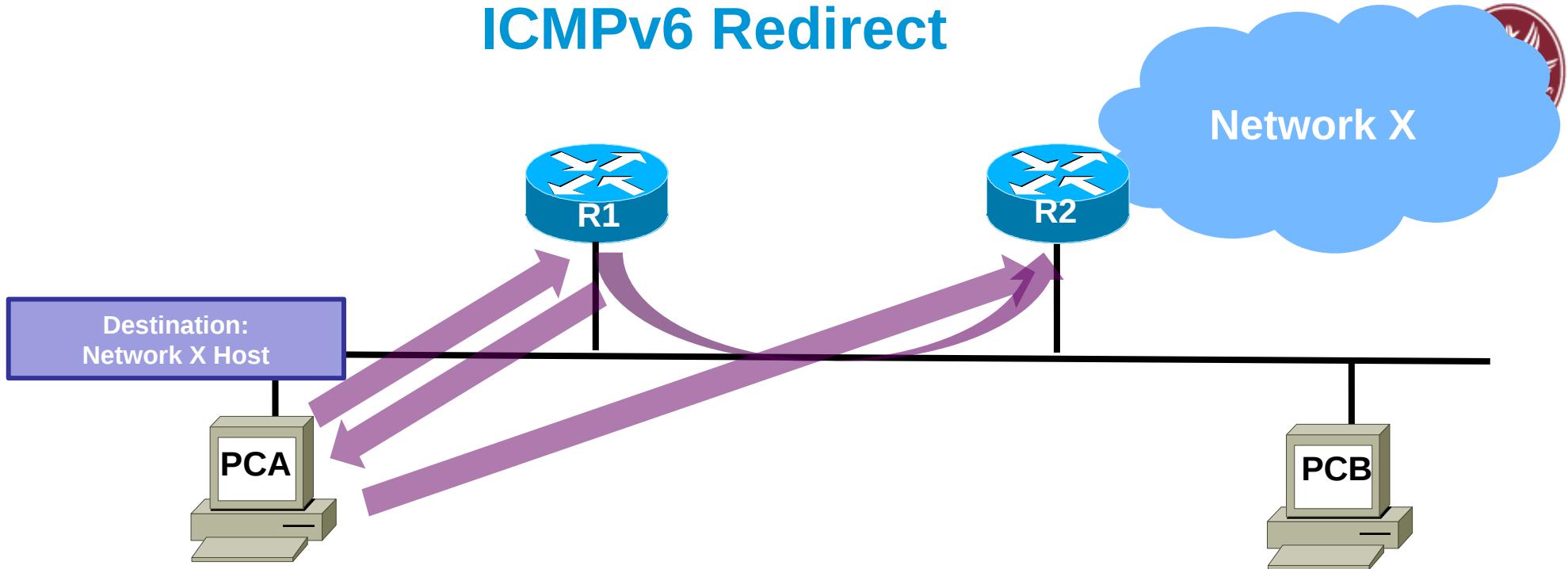
Lab activity



Main tasks

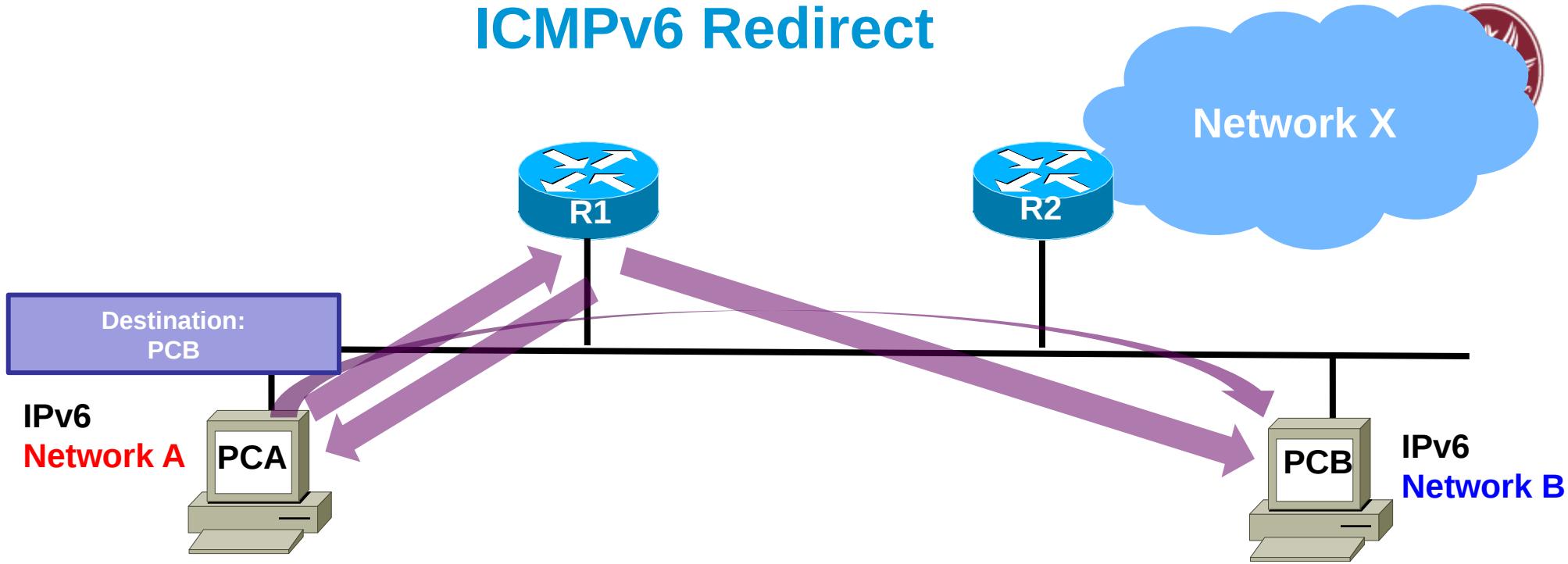
- Network eavesdropping
- ICMP redirect attack
 - MITM
- Reference links:
 - <https://developers.redhat.com/blog/2018/10/22/introduction-to-linux-interfaces-for-virtual-networking/>
 - <https://www.fir3net.com/Networking/Terms-and-Concepts/virtual-networking-devices-tun-tap-and-veth-pairs-explained.html>
 - <https://www.ettercap-project.org/>
 - <https://pentestmag.com/ettercap-tutorial-for-windows/>

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.



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>
- You can move in the directory and run `lstart`
 - **NOTE:** launch docker first or the first `lstart` attempt can (...will...) fail

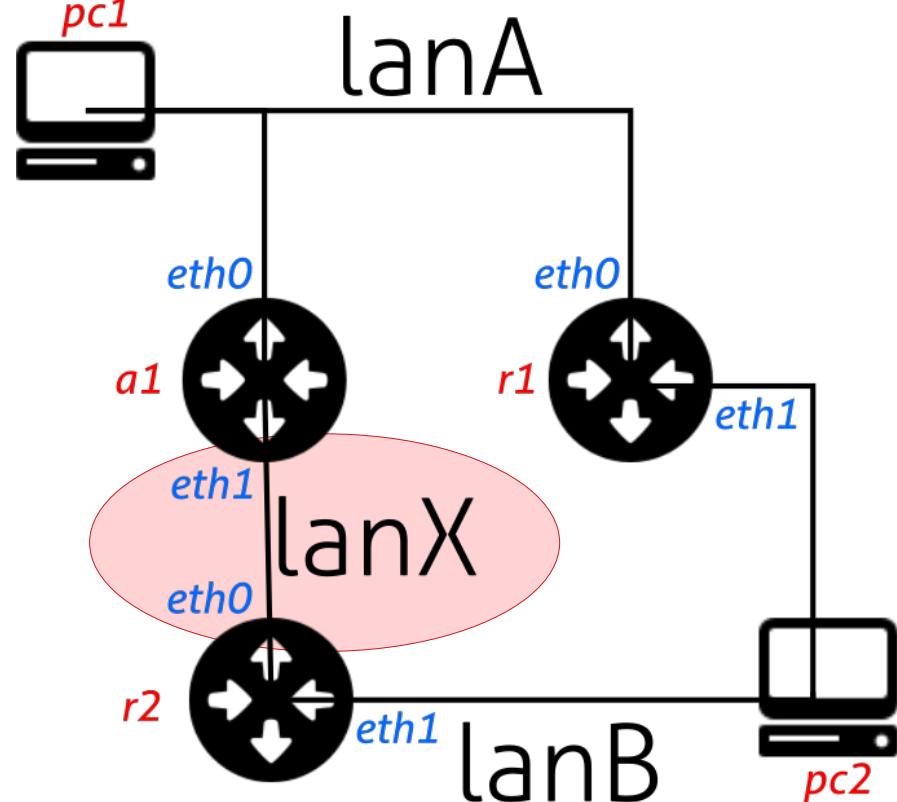


Lab activity: ex4



Exercise 1: 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`



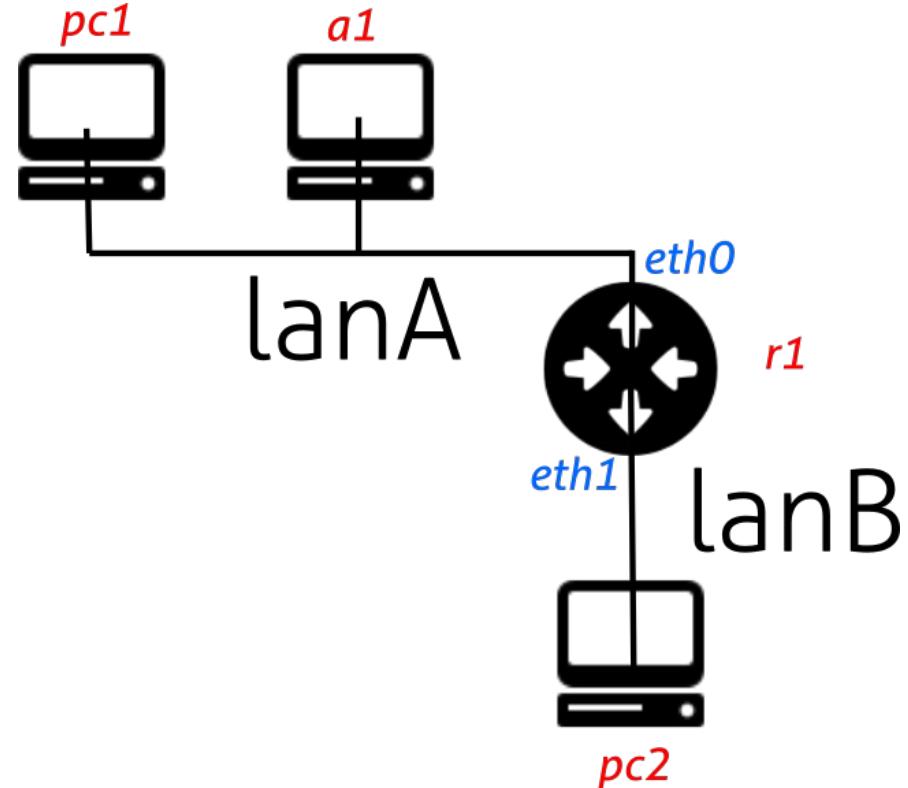


Lab activity: ex5



Exercise 2: 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?**
- References:
- IPv6 security references:
<https://www.ripe.net/support/training/material/ipv6-security/ipv6security-references.pdf>
 - 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)

Practical Network Defense

Master's degree in Cybersecurity 2021-22

Network hardening

Angelo Spognardi
[*spognardi@di.uniroma1.it*](mailto:spognardi@di.uniroma1.it)

*Dipartimento di Informatica
Sapienza Università di Roma*



Agenda

- Introduction
- Management plane protection
- Control plane protection
- Data plane protection



Network hardening, i.e. protecting network devices

- Computer networks are composed of different types of devices.
- If even one of them is breached, the entire infrastructure can be compromised.
- The use of a methodology to protect network devices makes it possible to reduce the risk of violations and to limit the impact of anomalous events, whether they are voluntary (attacks) or involuntary (human errors or failures).



Three scopes of action

- Management plane
 - The scope of network device management
 - It consists of an administrator's protocols and tools to configure, monitor, or access a network device (e.g., SSH, SNMP, NTP).
 - Breaches in this area can be caused by overly simple passwords or insecure protocols, resulting in unauthorized access or loss of access to the device.
- Control plane
 - The scope of support for the operation of network devices
 - It consists of the protocols and mechanisms devices use to perform their tasks (e.g., routing protocols).
 - Violations in this area are usually caused by unauthorized data exchange with the device, resulting in loss of performance (denial of service).
- Data plane
 - The scope of operation of network devices
 - It corresponds to the traffic forwarded by network devices (be they switches, routers, or firewalls) and the paths that appliances choose for individual packets.
 - Violations in this area are usually caused by external events (congestions), malicious interventions (spoofing, redirect, hijacking, etc.), and failures and can result in the alteration of packet paths and a block of network services.



Protection must occur at all scopes

- In all the three areas of action of the network devices it is possible to have violations
- The three areas are closely linked, as anomalies in one can be reflected in the others.
- It is important, therefore, to adopt best practices to protect devices in all three areas.

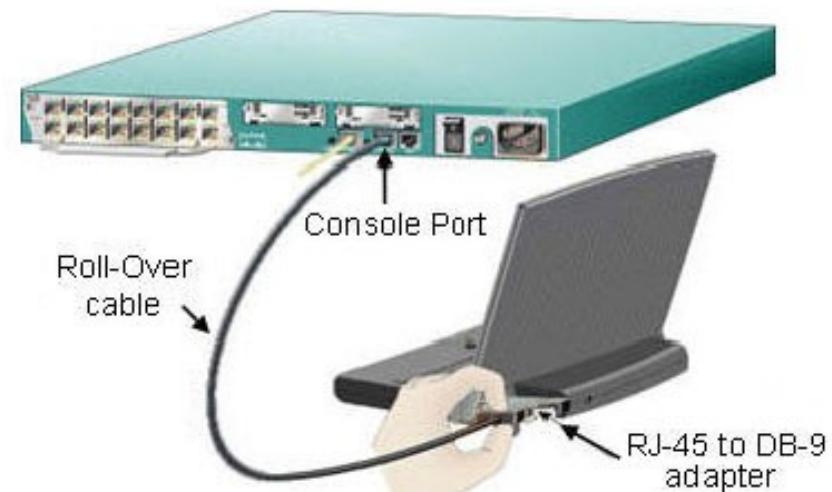


Management plane protection



Appliance access protection

- Access to a device is the first step in configuring its operation and determining its behavior.
- Suppose the access occurs in person (with a serial terminal or a laptop connected to the console port). The risk is reduced since the access is usually restricted physically (access to the room where the device is).
- Suppose access can also be done remotely (through the network itself, using a virtual terminal, such as SSH or telnet). In that case, the risk is higher since anyone who can send traffic that reaches the device can claim to be a network administrator.
- Access to a device also allows access to other network devices' monitoring and status management functions, usually through the SNMP protocol.





Password policy

- Passwords are the simplest and most widely used form of authentication
- It is good to use passwords that are difficult to guess, for example, by employing:
 - passwords that are at least eight characters long
 - Devices can often force you to use a minimum number of characters for a password
 - passwords that are combinations of alphanumeric, upper and lower case characters, punctuation marks, and spaces are often used to create passphrases, i.e., passwords consisting of multiple words, usually not logically related
 - passwords that are changed frequently
 - passwords that are not overly complex, perhaps complicated to remember, requiring transcription to be used



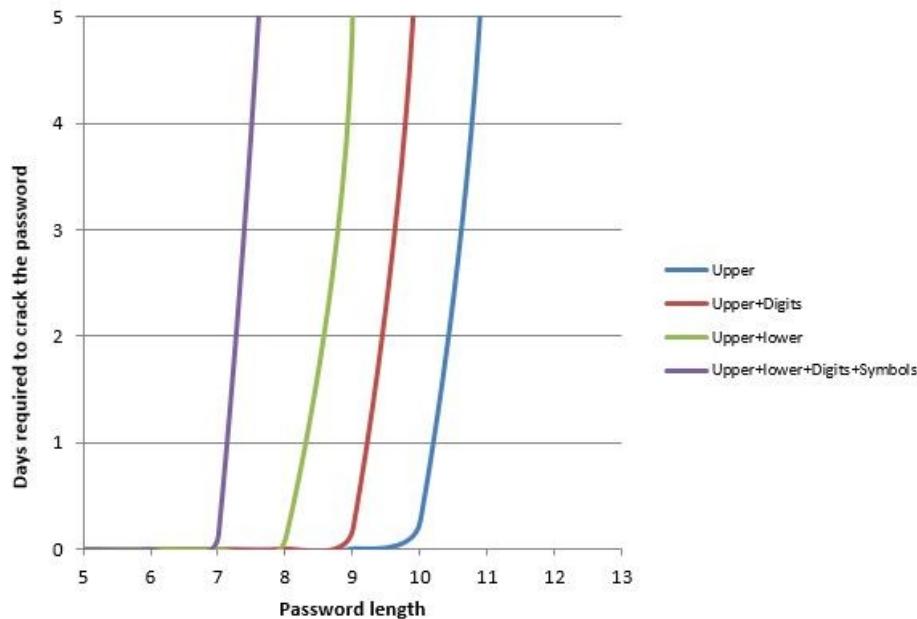
Brute force protection

- The use of different character types and a minimum length ensures that the password search space is enormous and such that a brute force attack is challenging to accomplish
- An attacker will guess a password with only two lower case letters in $2^{12}=441$ attempts.
- With an eight-character password of upper and lower case letters, the attacker will guess numbers and symbols in $(21+21+10+36)^8=3.6*10^{15}$ attempts, or more than 3 million billion attempts.
- It is a good idea to configure the devices so that they can
 - store passwords in an encrypted way
 - temporarily block accounts for which the password is wrong three times in a row, notifying the incident in the logs

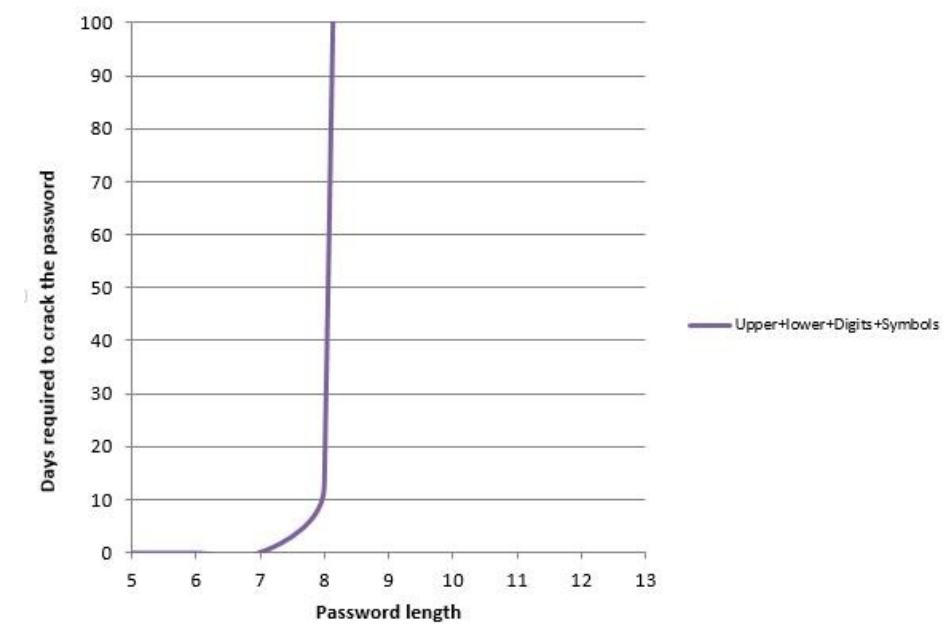


Password complexity

Example 2 - Modifying the Charset



Example 3 - Enforcing Password Length



<https://www.coresecurity.com/blog/the-exponential-nature-of-password-cracking-costs>



Use the AAA principle

- AAA is the principle of reducing unauthorized access to resources.
- It provides for identifying users and allowing them access only to those components for which they are authorized, keeping a record of their actions.
- **Authentication:** verifying the identity of a user
 - By user name and password, by token or similar
- **Authorization:** verify if a user is authorized to act on a system
 - The association between users and permissions can be done in different ways. The best known is RBAC, Role-Based Access Control, which considers the presence of different roles in the system, or groups of users with the same permissions (e.g., administrator, junior administrator, controller, and so on)
 - In RBAC, each user is associated with one or more roles, and each role has a set of permissions on system objects
- **Accounting/auditing:** store the details (e.g., time, duration, command used, and so on) of each action taken by each user in a permanent, unalterable log
 - Usually, network devices can send information to different destinations, such as a terminal, an SNMP server, or a Syslog server.



Use centralized solutions

- Often, it is possible to configure solutions that concentrate AAA functions in a single point in network devices, for example, an ACS, Access Control Server.
- This allows you to have a centralized point from which to manage all the devices on the network.
 - The network devices (switches or routers) connect to the server and interact with it every time they need to verify the identity of a user trying to access it, the user's permissions, and record the actions it performs.
- Usually, these functions use specific protocols for managing users, passwords, permissions, authentication and authorization checks, and so on.
 - Examples are RADIUS for user access and TACACS+ for administrator access.
- The alternative to centralized solutions is the local solution, where each device has its own set of users, passwords, and permissions.
 - This solution is often used to overcome the problems that would arise if the authentication server was temporarily unavailable.



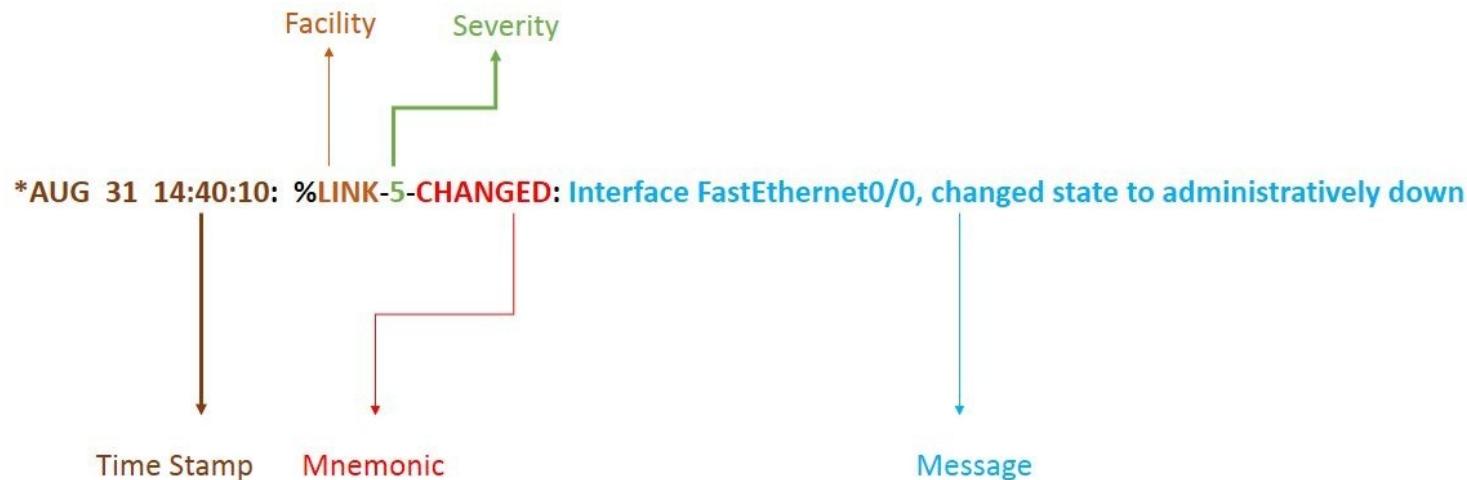
Use a reliable NTP server

- To have reliable Accounting and Auditing, all devices must know the current time correctly.
- For this purpose, the Network Time Protocol (NTP) is used.
- Usually, one of the devices on the network is designated as the reference NTP server, and all the other devices interact with it to synchronize with its time.
 - This device, in turn, can refer to another trusted NTP server, perhaps one that can be reached on the Internet, to receive the current time accurately.
- This ensures that all logs from all devices can be compared temporally.
- NTP version 3 is preferred since, unlike previous versions, it makes use of authentication and integrity verification mechanisms



Syslog

- syslog is a standard mechanism for generating log messages.
- Its structure allows you to efficiently separate the applications that generate logs from those that need to store them and those that need to consult them.
- The syslog server is critical to centrally collecting and managing logs
- One or more servers receive log messages from network devices representing noteworthy events





Correctly configure Syslog

- In Syslog, there are eight levels of criticality (severity) for individual log messages, ordered in such a way that levels with low values have higher criticality than those with high values:
 - 0 Emergencies, 1 Alert, 2 Critical, 3 Errors, 4 Warnings, 5 Notifications, 6 Informational, 7 Debugging
- Devices send log messages with a level less than or equal to the set criticality level.
 - This implies that a high level of lower criticality (such as 7, Debugging) will generate many more messages than a low level of higher criticality (such as 3, Errors).
- It is therefore essential:
 - to choose the appropriate level of criticality
 - For example, limit the use of Debugging only to the time required to solve a sudden anomaly
 - that the syslog server has an adequate log storage capacity, both in terms of space and computing capacity.
 - This is because the messages are transmitted and stored in plain text, with a timestamp (synchronized thanks to NTP) and with a text message



Remotely access using encryption

- Remotely accessing network devices must be done securely, i.e., using encryption protocols.
- For devices that require command-line configuration (CLI), avoid telnet, which is an unencrypted protocol since it involves sending a username and password without any protection.
 - Any intermediate node that separates the administrator and the remote device can potentially intercept the traffic and learn the credentials to access the device's configuration.
 - Administrators should only use telnet on an isolated channel (not intercepted) or within an encrypted tunnel (such as a VPN).
 - SSH should always be preferred, as it offers the same functionality as telnet but encrypts every packet exchanged between the device and the administrator. SSH version 2 is more secure than version 1.
- For devices that provide configuration through a graphical interface (GUI), similarly, should always be avoided the use of the HTTP protocol (clear) and prefer the use of HTTPS (encrypted).



Control plane protection



Protect control data

- It is essential to protect network devices to prevent them from using non-genuine information.
- This is done both to avoid unauthorized changes to the way traffic moves through the network and avoid overloading devices (denial of service attacks).
- For this purpose, we use
 - Control Plane Policing and Control Plane Protection, namely a series of measures that can limit the packets addressed directly to the devices and that require the use of their CPUs
 - routing protocols with authentication reduce the risk of using information for non-genuine traffic routing.



DoS protection

- The primary purpose of routers is to forward packets, so routers are very optimized.
 - Packet forwarding, based on routing table information, is almost always done from the cache, with zero impact on the CPU.
- Process packets directed to a router, instead, involve the CPU, possibly considerably.
 - E.g., routing table updates, management traffic (telnet, ssh, SNMP), service traffic (IGMP, DHCP), IP options that require processing by routers.
- Therefore, flooding routers with packets for processing can impact their performance and cause a DoS attack.
- The specific protections for the control plane limit this kind of traffic, setting up thresholds for the reception.
 - For example, x packets per second for protocol y, z traffic only from interface k, ignore protocol w, etc.



Manage dangerous ICMP packets

- The ICMP protocol can also have adverse effects on the CPUs of network devices.
 - For example, when there are particular topologies that, according to ICMP specifications, cause the systematic generation of such packets.
 - E.g., ICMP redirects in multi-point ethernet networks (with multiple routers connected) or in point-to-point ethernet networks.
- Considering that the ICMP packets are informative, some types of packets can be disabled in the devices without altering the functionality of the network.
- To limit the impact on the CPU of network devices, it is good practice to use the ICMP packet filtering mechanism to block the following ICMP packets:
 - **ICMP redirects**, which suggest an alternate route if the destination can be reached through another router in the same network
 - They can be used to indicate a different gateway and, thus, exposed to a man-in-the-middle.
 - **ICMP unreachable**, which informs the sender of a packet that the final destination is unavailable (e.g., not responding to ARPs or no route to that destination)
 - The unreachable ICMPs, besides overloading the CPU of the routers, can be used to know the internal topology of a network.



Only use authenticated routing protocols

- Routing information is critical to forwarding packets according to the correct routes.
- It is essential to ensure that packets with routing information from other routers are authentic.
- For this reason, it is best to use the authenticated version of the routing protocols wherever possible.
 - This ensures that unauthorized routers cannot distribute false information.



Data plane protection



Data plane protection

- The purpose of network devices is to move packets through the network according to the security policies established by governance.
- Without proper protections, attacks can be made to alter packet forwarding rules, potentially causing security policy violations.
- In addition, because network devices operate with virtually no administrator intervention once configured, it is challenging to observe security policy violations without proper monitoring.
- Therefore, data plane protection must be as thorough as possible.



Operate at all protocol stack layers

- To protect the data plan, it is necessary to intervene in several protocol stack layers.
- At level 2, you have to protect devices from possible MAC-IP association changes.
- At layer 3, you must protect devices from IP packets with dangerous configurations that attempt to map the network.
- At layer 4, you must protect devices from ICMP packets that may alter the normal flow of packets within the network.



Level 2 protection (switch configuration)

- Disable gratuitous ARP packets.
 - To avoid ARP spoofing attacks (MAC address theft), typical of man-in-the-middle attacks.
- Enable dynamic ARP inspection (DAI) mechanism
 - To protect against ARP spoofing and ARP poisoning, typical of man-in-the-middle attacks
- Disable ARP Proxy IP if not needed
 - IP proxy ARP allows ARP packets to traverse routers when a logical network is physically divided by routers and not just switches. This can be exploited for man-in-the-middle attacks.
- Enable port security mechanism
 - Allows only a limited number of MAC addresses to be exchanged, preventing attacks such as CAM table overflow, DHCP depletion, or misuse of a switch port.
- Enable DHCP snooping mechanism
 - This allows DHCP response packets to be forwarded only from authorized ports connected to a trusted DHCP server.
- Enabling the IP source guard mechanism
 - To allow blocking all traffic with an abnormal IP-MAC address association, typical of IP spoofing.



Level 3 and 4 protection

- The main tools to protect networks at layers 3 and 4 are Access Control Lists (ACLs) used for packet filtering functions.
- ACLs are the rules that routers follow to identify the type of traffic of the packets they forward.
- ACLs are the rules that routers follow to identify the type of packet traffic that they are forwarding.
- They can identify packets by considering only IP addresses (standard ACL) or IP addresses and Layer 4 header information (extended ACL).
- Depending on the specified policy, routers can transform packets (e.g., blocked, subjected to NAT, forwarded in a VPN, etc.).
- For the protection of networks, the primary use is to filter packets (packet filtering) according to the network's security policy.
 - They are also used for NAT, Quality of service (QoS), VPN traffic selection, policy-based routing, or routing information.



Use rules (ACL in routers) for traffic filtering

- Block packets with IP address spoofing
 - Through ACLs, it is possible to block all packets used as source IP addresses IPs that are not consistent with the network topology.
 - For example, a host in a 10.0.0.0/8 network that uses IP 11.0.0.1
- Block packets that can lead to network mapping (scanning)
 - For example, taking care not to block functional features in the network, you can use ACLs to block UDP or ICMP packets from outside the web that may reveal information about the internal structure.
- It's a good idea to allow only packets that correspond to the traffic expected on the network.
 - According to the principle of least permission, blocking everything that is not explicitly expected to be exchanged in the network would be good.



Use rules (ACL in routers) to authorize only trusted sources

- Routers can use ACLs to limit the type of traffic of supporting protocols
- Examples
 - an ACL allowing NTP traffic from the trusted server only
 - an ACL allowing SSH access only from administrator hosts.
 - an ACL allowing ICMP or SNMP diagnostic packets from the administrator's hosts only.



Lab activity



Agenda

- Introduction of Opnsense
- Management plane protection
- Control plane protection
- Data plane protection



Introduction of Opnsense



OPNsense

- OPNsense is an open-source router-firewall based on a particularly robust version of BSD
 - BSD is considered one of the most security-conscious Unix distributions
- OPNsense is very popular because it is also easy to use and install, besides having a free license
- It installs like a regular operating system
- Default behavior: DENY all



Some OPNsense characteristics

- Stateful inspection firewall
- A modern and intuitive graphical interface
- Built-in intrusion detection & prevention system (Suricata)
- High reliability (High availability & hardware failover)
- Availability of the most common network services (DNS, DHCP, traffic shaper, captive portal, proxy)
- Management of different types of VPN (Virtual private network)
- Management of backups & backup recovery
- Possibility of expansion through plugins
- Built-in reporting and traffic monitoring tools



OPNsense usage

- OPNsense is configured through a web browser (remote access)
- Only initial installation and emergency access should be through the console (in-person access)
- The navigation system is very intuitive and based on graphical menus

The screenshot shows the OPNsense web interface at <https://192.168.1.75/index.php>. The title bar indicates the session is root@mainfw.pndeflab. The left sidebar menu includes: Cruscotto, Licenza, Password, Esci, Reporting, Sistema, Interfacce, Firewall, VPN, Servizi, Power, and Aiuto. The main content area is titled "Lobby: Cruscotto" and displays "System Information". Key details include:

Nome	mainfw.pndeflab.edu
Versioni	OPNsense 19.1-amd64 FreeBSD 11.2-RELEASE-p8-HBSD OpenSSL 1.0.2q 20 Nov 2018
Aggiornamenti	Clicca per controllare se ci sono aggiornamenti.
Tipo di CPU	Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz (1 cores)
Uso CPU	100% (0/0)
Carico medio	0,76 0,40 0,30
Uptime	02:01:06
Data e ora correnti	Thu Aug 12 11:34:41 CEST 2021
Ultima modifica alla configurazione	Thu Aug 12 1:55:59 CEST 2021
Dimensioni tabella di stato	0 % (0/201000)
Uso MBUF	0 % (1016/125460)
Uso Memoria	9 % (198/2011 MB)
Uso Disco	11% ([ufs] 1,4G/15G)

The right side of the interface lists "Servizi" and "Gateway" configurations. The "Servizi" section includes configd, login, ntpd, openvpn, pf, strongswan, suricata, syslog, and unbound. The "Gateway" section includes INTERNET_DHCP (IP 192.168.1.1) and HostonlyNet (IP 100.64.200.1). The bottom of the page credits OPNsense (c) 2014-2019 Deciso B.V.



First access

- To access the OPNsense control panel, just enter the IP address in the web browser
 - ex: 100.100.4.1
- In the login window, enter **root** as user name and **opnsense** as password



Management plane protection

- Use strong passwords
 - Change the default password
- Use encrypted communication
 - Make sure to log in only using HTTPS
- Configure NTP service on internal interfaces
- Send logs to a syslog server
 - Configure opnsense to send logs to a centralized server



Opnsense management plane

- Change admin user password
 - Lobby: Password
 - Enter a password other than opnsense
 - Save the new password → Save
- Make sure to log in only using HTTPS
 - System: Settings: Administration → Protocol HTTPS
 - Apply the new settings → Save
 - Caution: it may be necessary to reload the page and authenticate in the firewall again for switching to encrypted communication



Management plane in Opnsense

- Configure NTP service on internal interfaces
 - Services: Network Time: General → select INTERNAL, DMZ, EXTERNAL
 - Save the new settings → 
- Send logs to a syslog server
 - System: Settings: Logging / targets → add 
 - Add as target the machine logserver.acme-XX.test, listening on port 514 via UDP.
 - To limit the number of logs, do NOT send messages for debug and info levels.
 - Save new settings → 
 - Apply changes → 



Control plane protection

- Protect against too many ICMP messages
 - Limit incoming ICMP traffic
- Block potentially malicious ICMP messages
 - Filter redirect and unreachable ICMP messages from external sources



Opnsense control plane: limit incoming ICMP traffic

- Firewall: Shaper: Settings
 - In Pipes tab, add a limitation to 10 Kbit/s without selecting mask
 - In Rules tab, a rule for WAN about ICMP packets from each source and for each destination, using the limitation defined in Pipes tab as target
- Apply changes → 

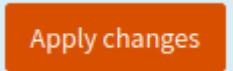


Opnsense control plane: filter ICMP redirect messages

- Firewall: Rules: Floating → 
 - Add a Block rule in the first position, for ICMP redirect packets from any source and for any destination on the DMZ and EXTERNAL interfaces.
 - The rule must be of type "in" since packets enter the interface.
 - If necessary, to change the position of a rule, check the box next to the rule and click on the arrow icon () at the position where you want to move the rule



Opnsense control plane: filter outgoing *ICMP unreachable* messages

- Firewall: Rules: WAN → 
 - Add a Block rule in the first position, for ICMP packets of type Destination Unreachable with outbound direction from each source and to each destination
 - To change the position of a rule, check the box next to the rule and click on the arrow icon () at the position you want to move the rule to
 - The rule must be of type "out" since the packets we want to block go out of the interface to reach the network connected to it
- Apply the new rules → 



Data plane protection

- Block packets with spoofed IP address
 - Filter packets with IP addresses not coming from local networks
- Allow only packets that match the traffic expected on the network
 - Allow access to the DMZ only to packets that require the services provided



Opnsense data plane protection

- Like many other firewalls, the data plane protects itself by inserting traffic filtering rules.
- In Opnsense, rules are either interface-specific or floating.
- Floating rules are considered before any other interface rules.
 - A floating rule of type "pass" source any destination any overrides any other rule, making firewall rules quite useless.
- The rules for an interface are either "in" or "out":
 - "in", when packets are generated by the network to which the interface is connected, and they want to enter the interface (so "in" input) to reach a different network
 - "out", when another network generates packets, reach an interface, and want to exit the interface (thus "out") to enter the network to which the interface is connected



Opnsense data plane protection: block packets with spoofed address

- Filter packets with IP addresses **not** coming from local networks
 - Firewall: Rules: DMZ →
- Add a pass rule for packets originating from the DMZ net, destined for any host and with any protocol.
 - The rules must be of type "in" since the packets enter the interface from the DMZ net network
- Create similar rules with the necessary modifications for the other internal interfaces, i.e. INTERNAL, EXTERNAL_CLIENT
- Apply the new rules →



Opnsense data plane protection: accept packets for running services

- Allow access in the DMZ only to packets that require the services provided (web and proxy)
- Firewall: Rules: DMZ
 - To disable the rule that allows all IP packets coming from any source to reach any destination (it goes from a default pass to deny)
 - To disable a "pass" type rule you can click on the green icon
- Firewall: Rules: WAN →  Add
 - Add a "pass" rule for packets destined for the host with the webserver on port 80, coming from any source.
 - Add a "pass" rule for packets destined via tcp to the host with proxyservice on port 3128, coming from any source.
 - The rules must be of type "in" since the packets enter the interface from the WAN1 network.
- Create similar rules with the needed modifications for the other networks
- Apply the new rules →  Apply changes



That's all for today

- **Questions?**
- References:
 - **CCNA Security 640-554 (Official Cert Guide)**



Main tasks

- CCNA Security 640-554
- Official Cert Guide

Practical Network Defense

Master's degree in Cybersecurity 2021-22

VPN, SSL/TLS and IPSec

Angelo Spognardi
[*spognardi@di.uniroma1.it*](mailto:spognardi@di.uniroma1.it)

*Dipartimento di Informatica
Sapienza Università di Roma*



Today's agenda

- VPN principles
- VPN device placement
- SSL Tunneling
- IPsec



VPN principles



Virtual Private Networks

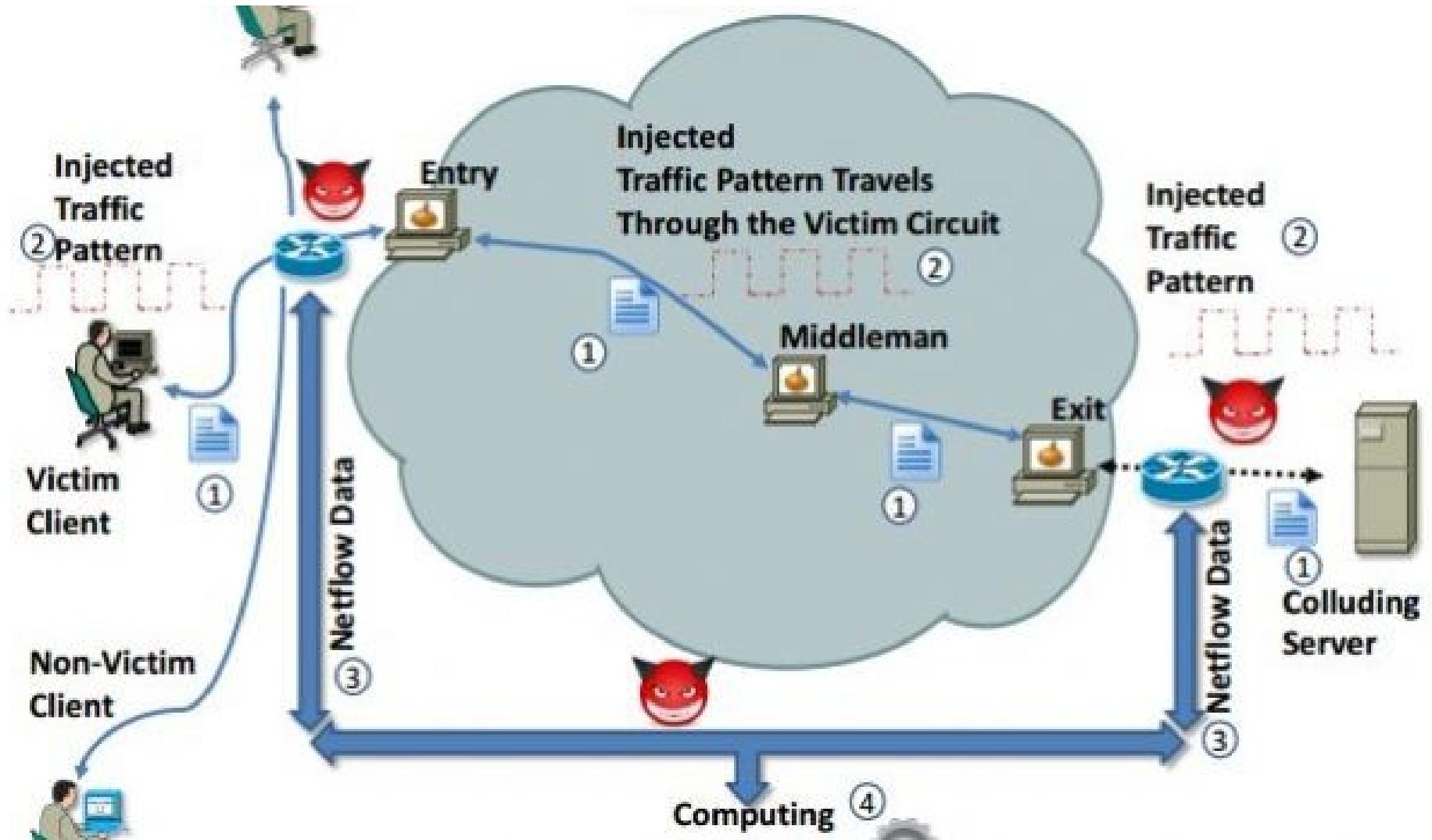
- Definition (NIST SP800-113): A virtual network, built on top of an existing network infrastructure, which can provide a secure communications mechanism for data and other information transferred between two endpoints
- Typically based on the use of encryption, but several possible choices for:
 - How and where to perform the encryption
 - Which parts of communication should be encrypted
- Important subsidiary goal: usability
 - If a solution is too difficult to use, it will not be used → poor usability leads to no security



Security Goals for a VPN

- Traditional
 - Confidentiality of data
 - Integrity of data
 - Peer Authentication
- Extended
 - Replay Protection
 - Access Control
 - Traffic Analysis Protection

Traffic analysis

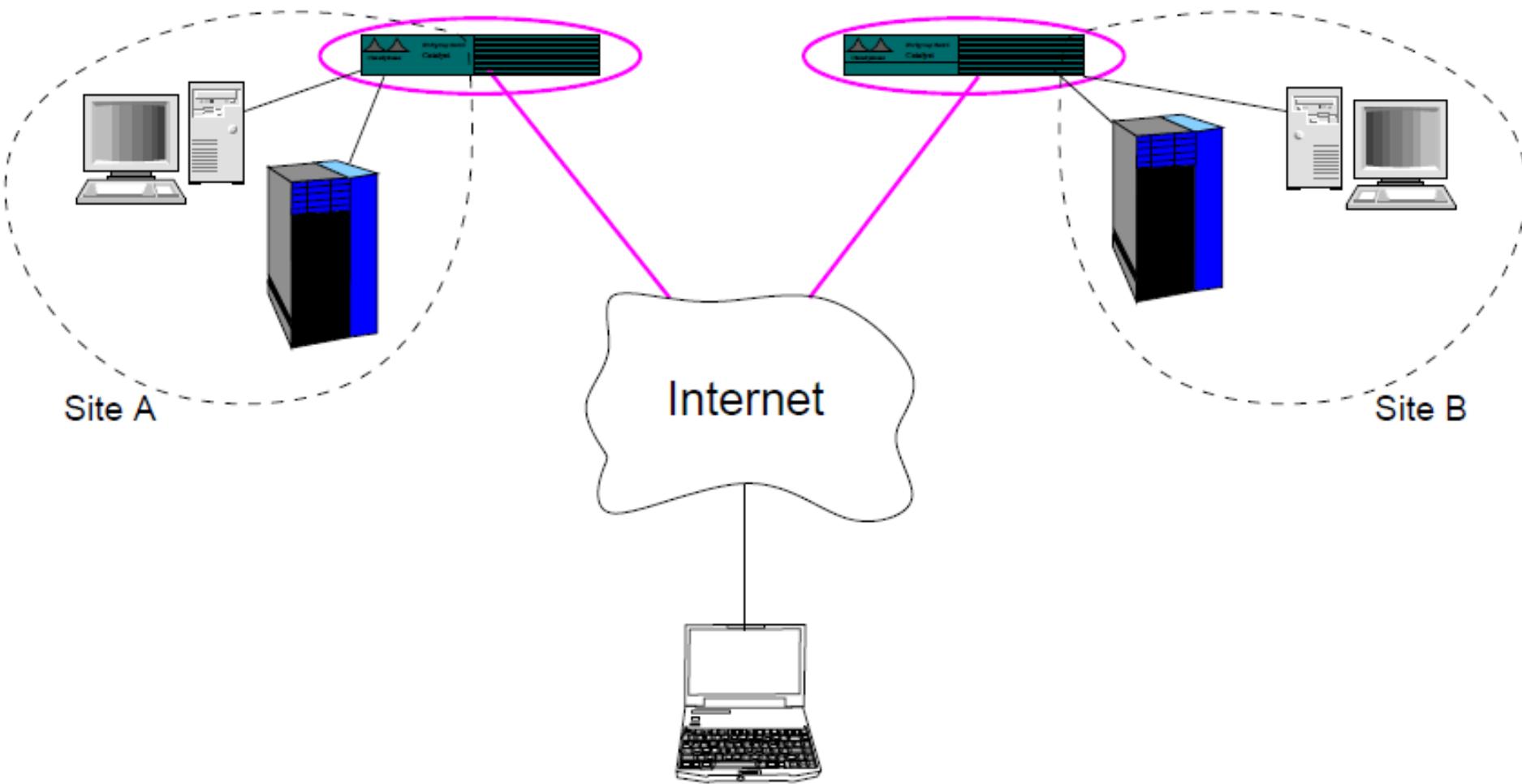




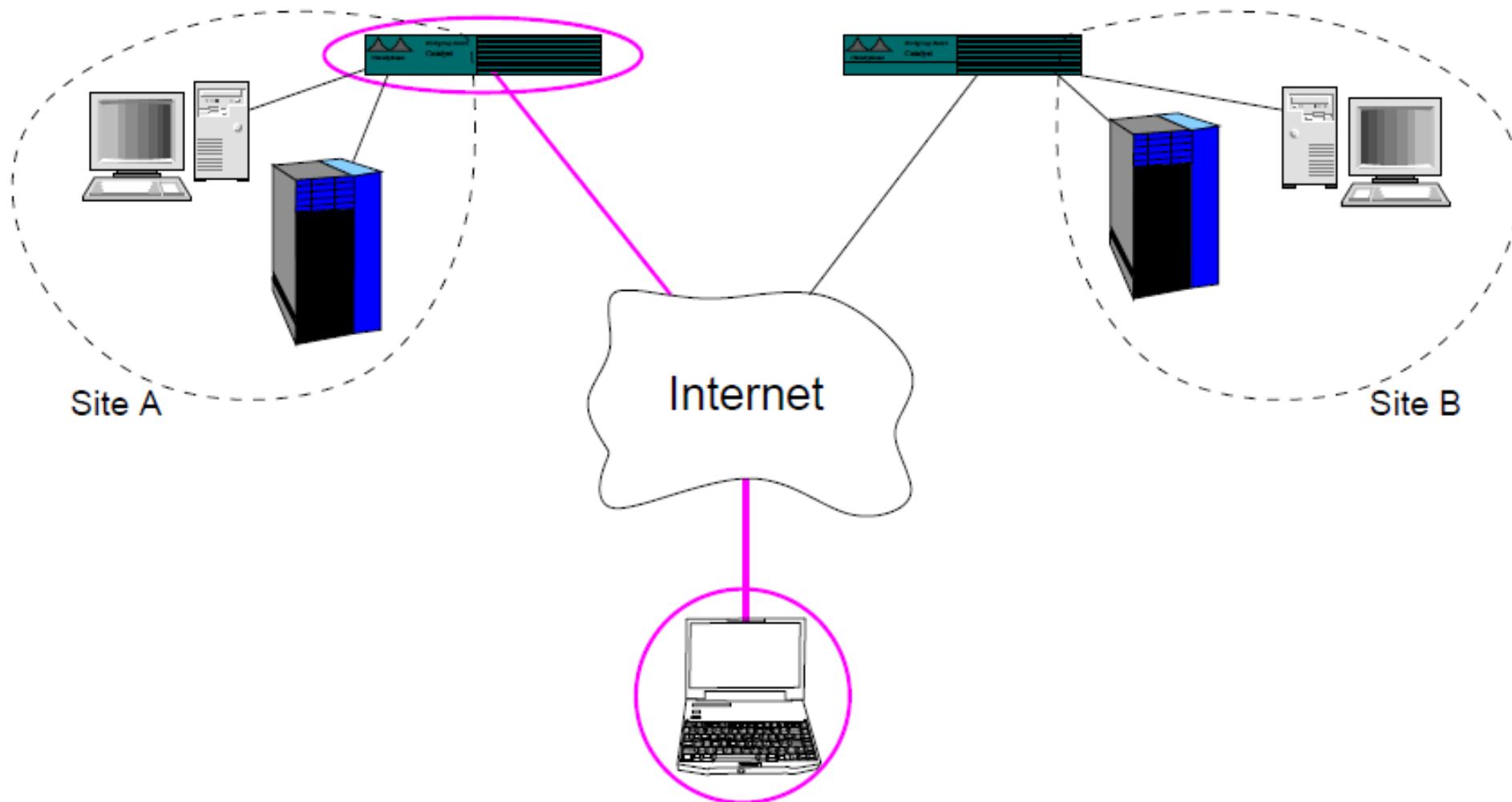
Usability goals

- Transparency
 - VPN should be invisible to users, software, hardware.
- Flexibility
 - VPN can be used between users, applications, hosts, sites.
- Simplicity
 - VPN can be actually used

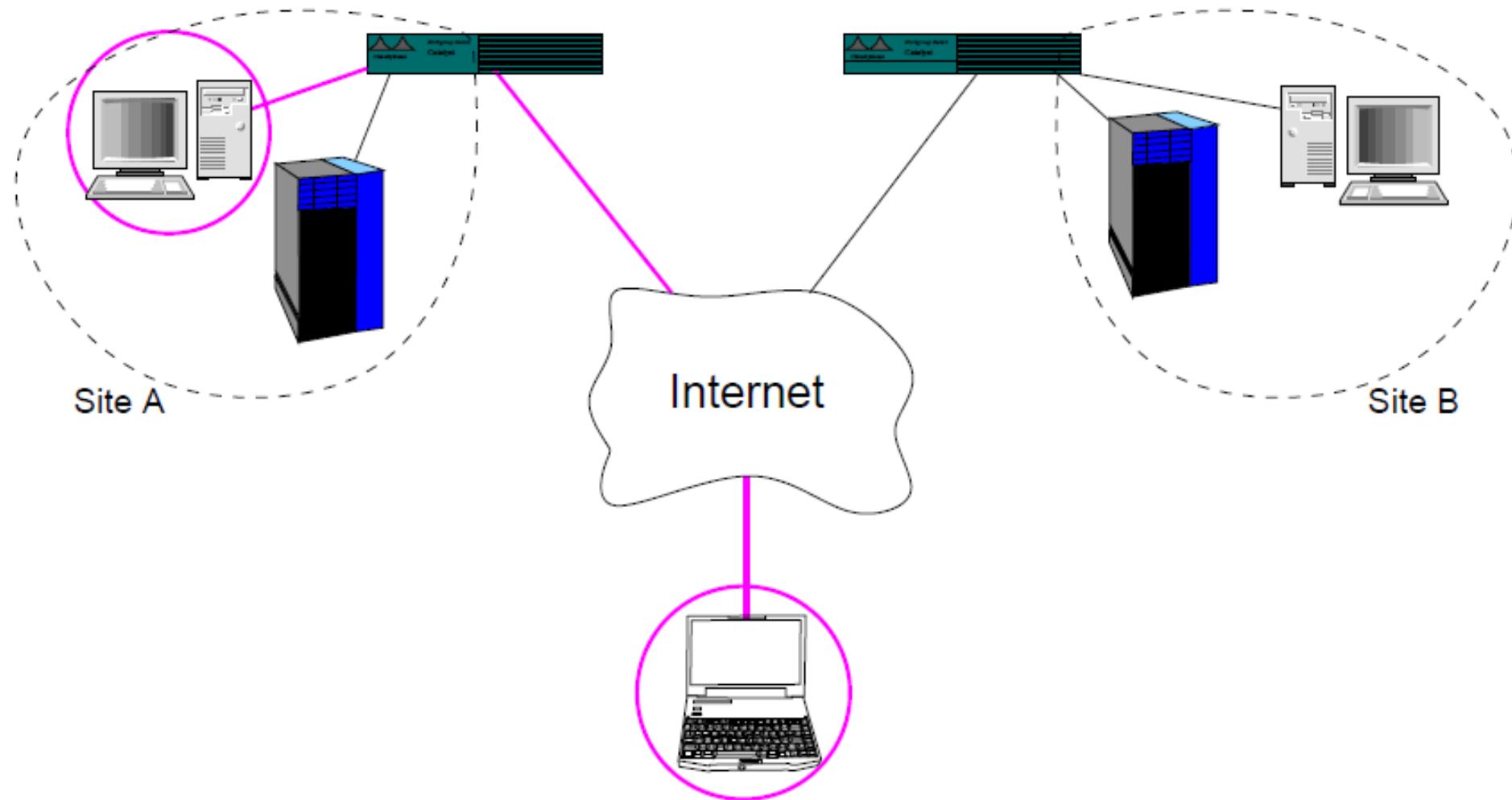
Site-to-site security



Host-to-site security

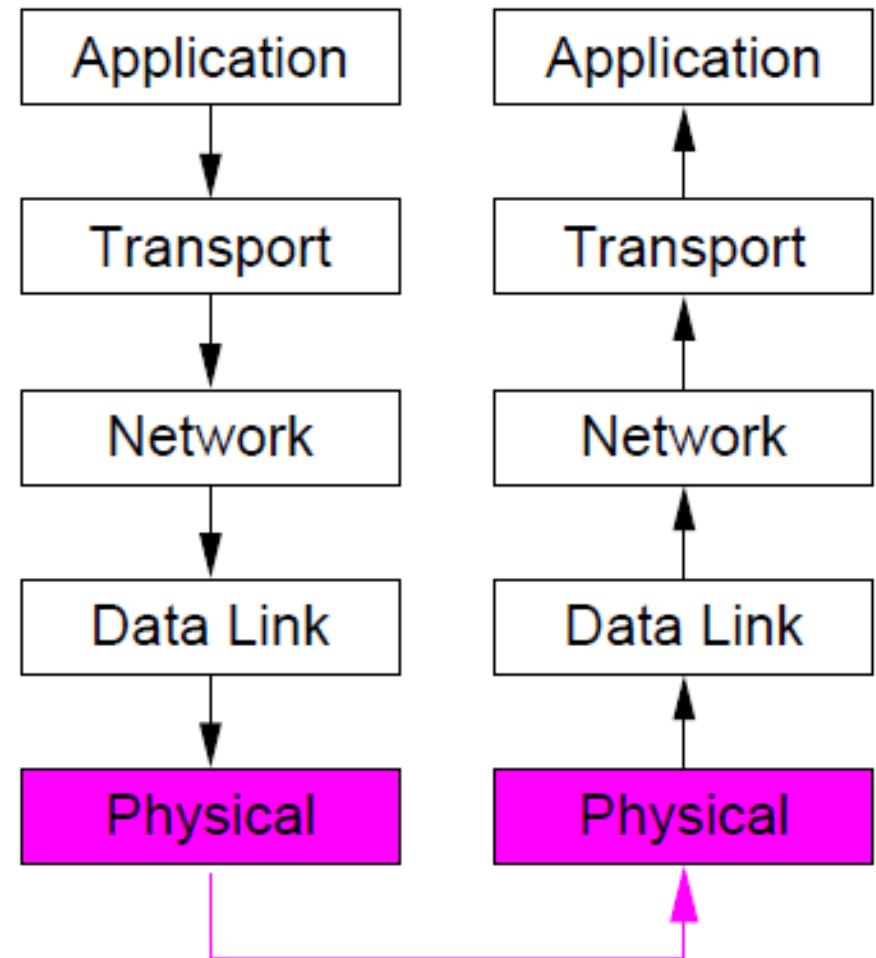


Host-to-host security

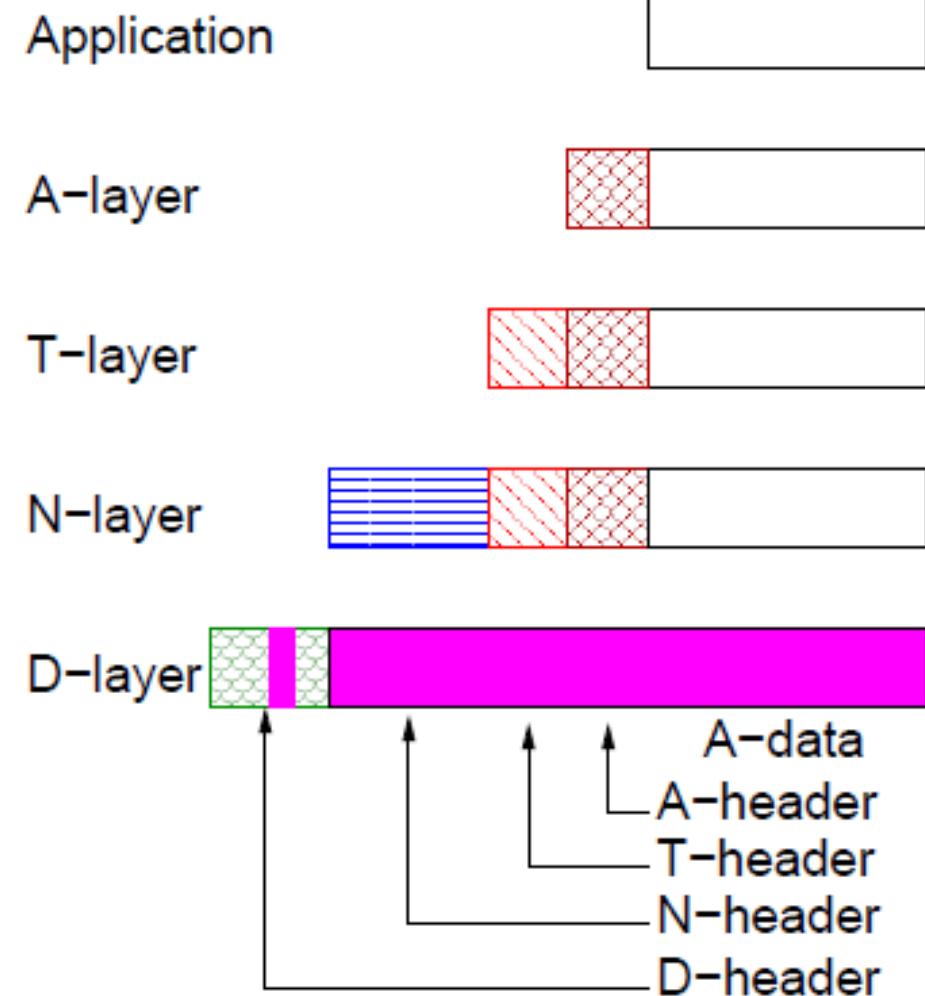
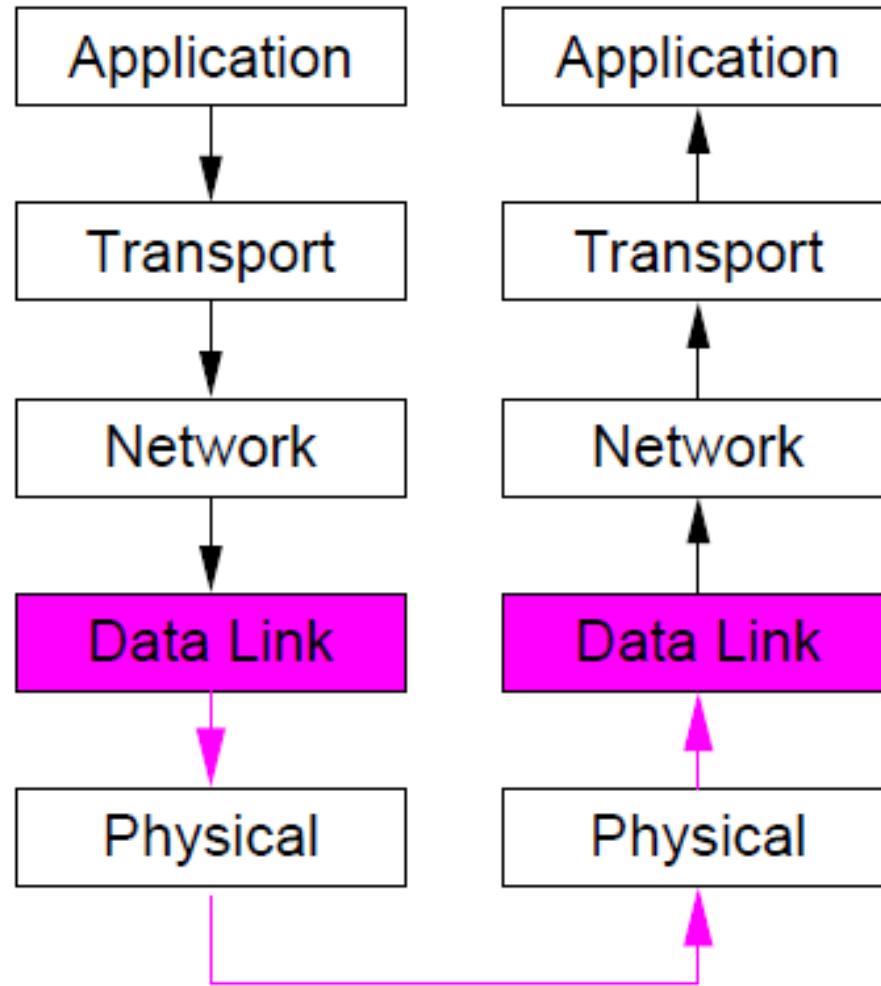


Physical layer

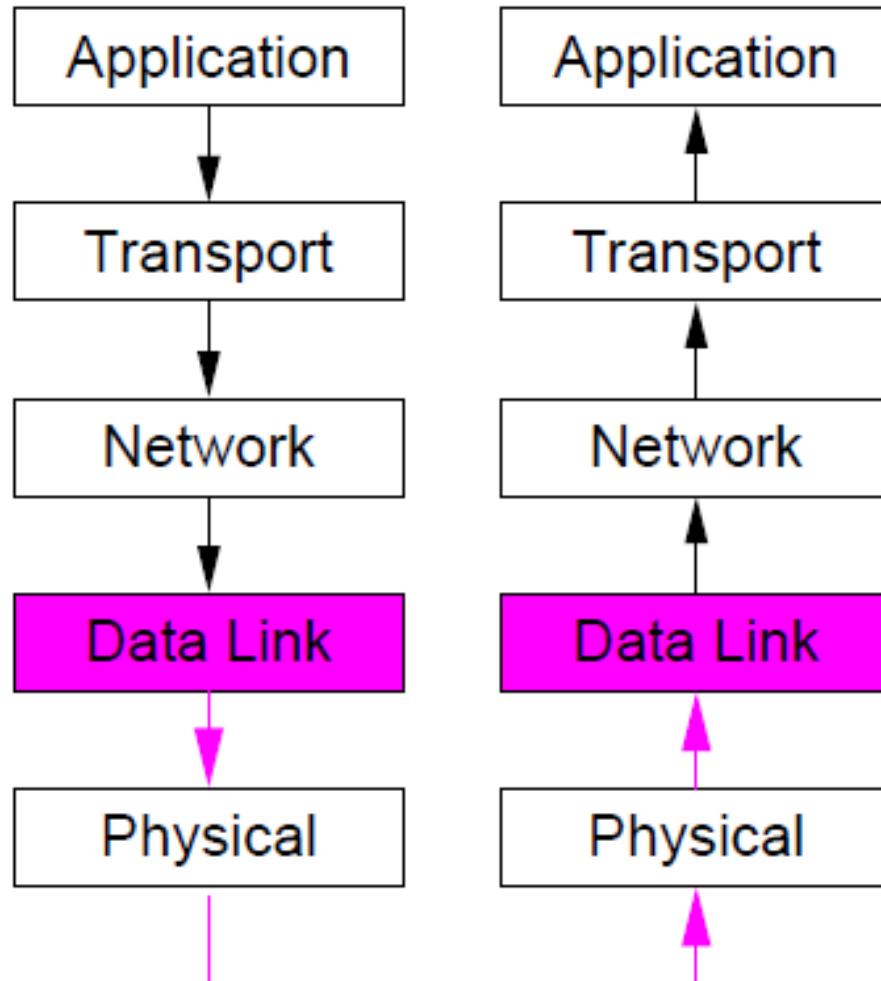
- Confidentiality: on cable
- Integrity: on cable
- Authentication: none
- Replay protection: none
- Traffic analysis protection: on cable
- Access control: physical access
- Transparency: full transparency
- Flexibility: can be hard to add new sites
- Simplicity: excellent!



Datalink layer: protect a single link

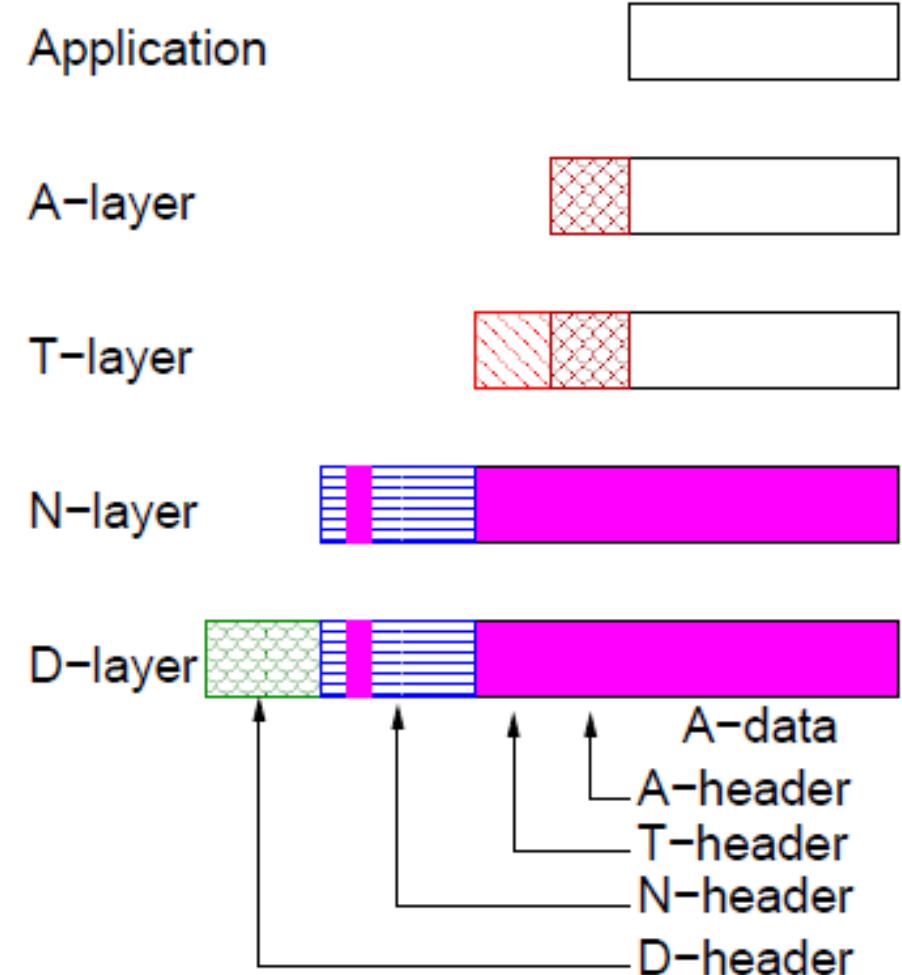
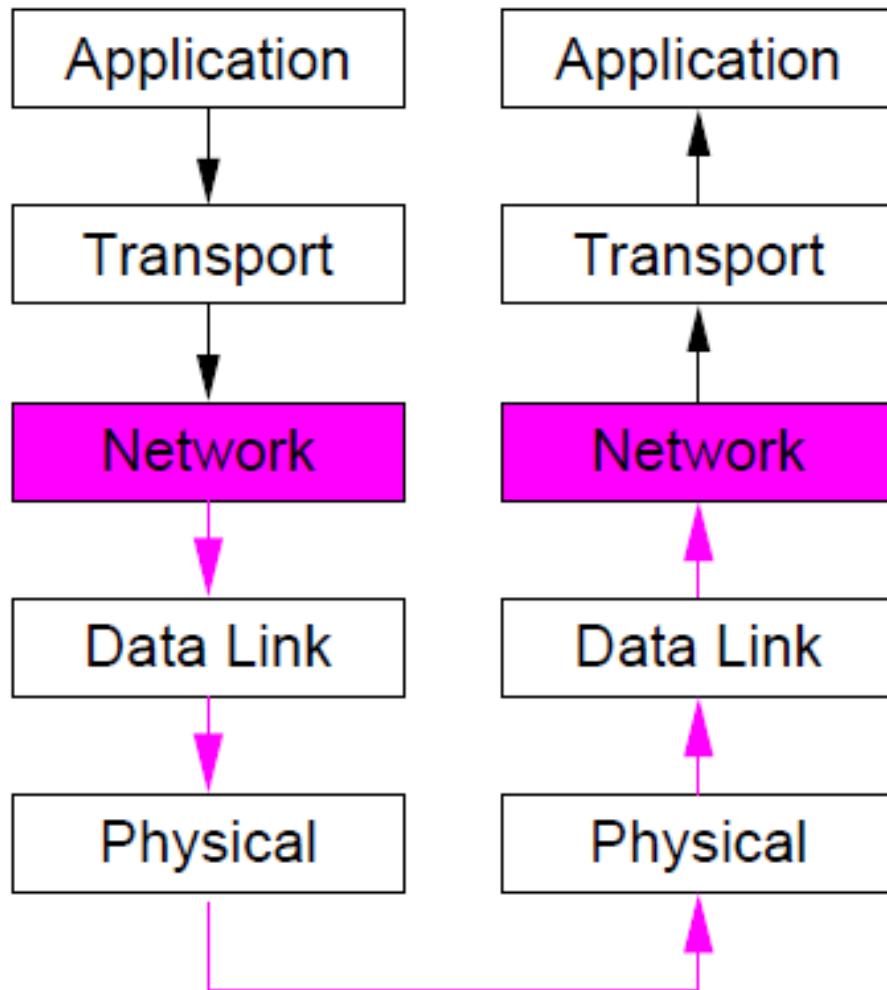


Datalink layer: protect a single link

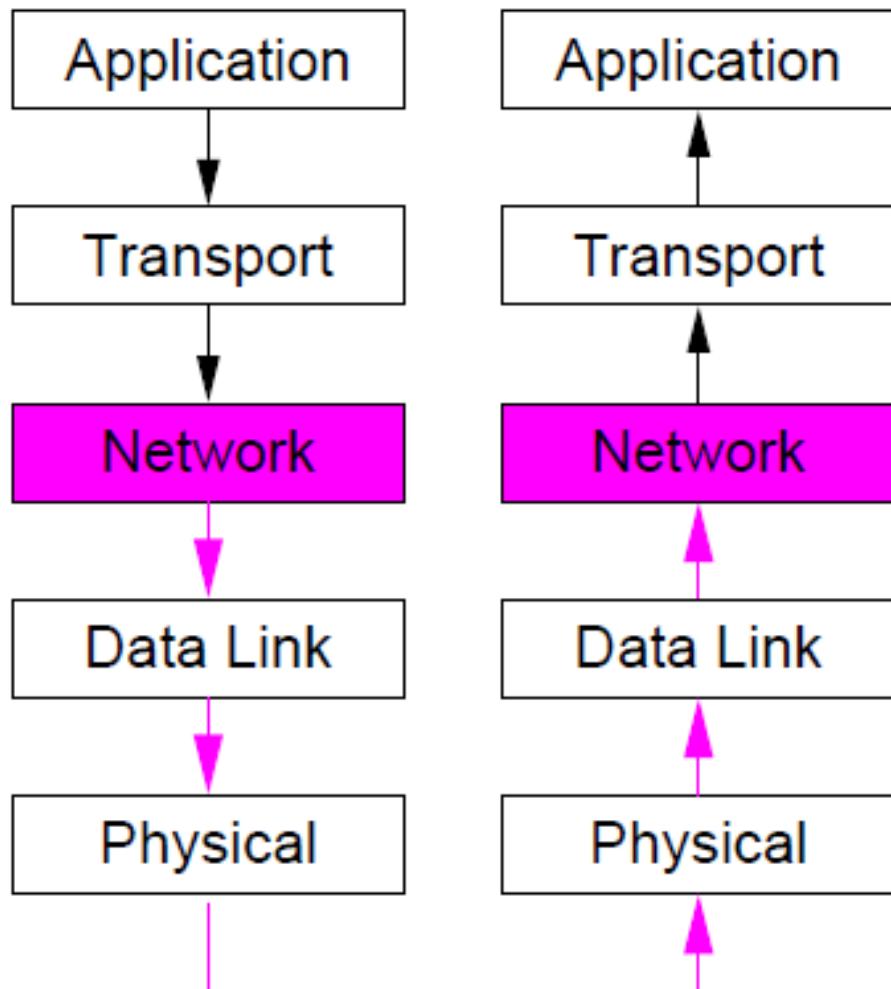


- Confidentiality: on link ("virtual cable")
- Integrity: on link
- Authentication: none
- Replay protection: none
- Traffic analysis protection: on link
- Access control: physical access
- Transparency: full transparency
- Flexibility: can be hard to add new sites
- Simplicity: excellent!

Network layer: protect end-to-end between systems



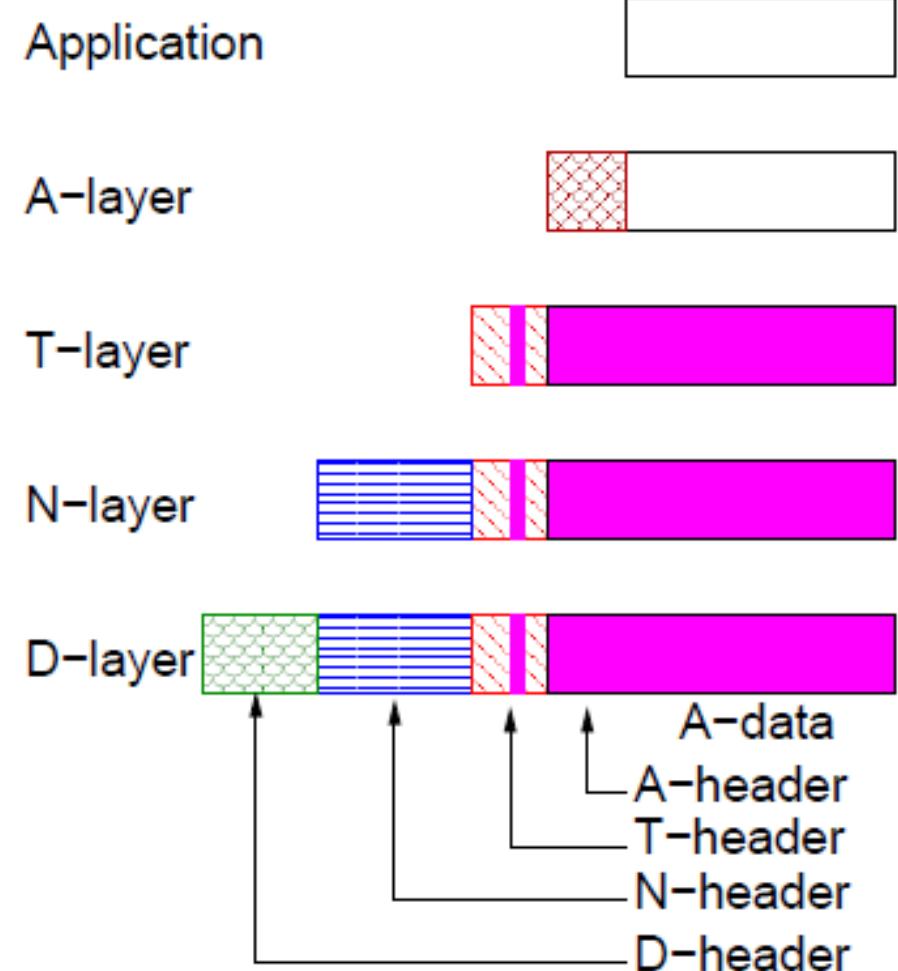
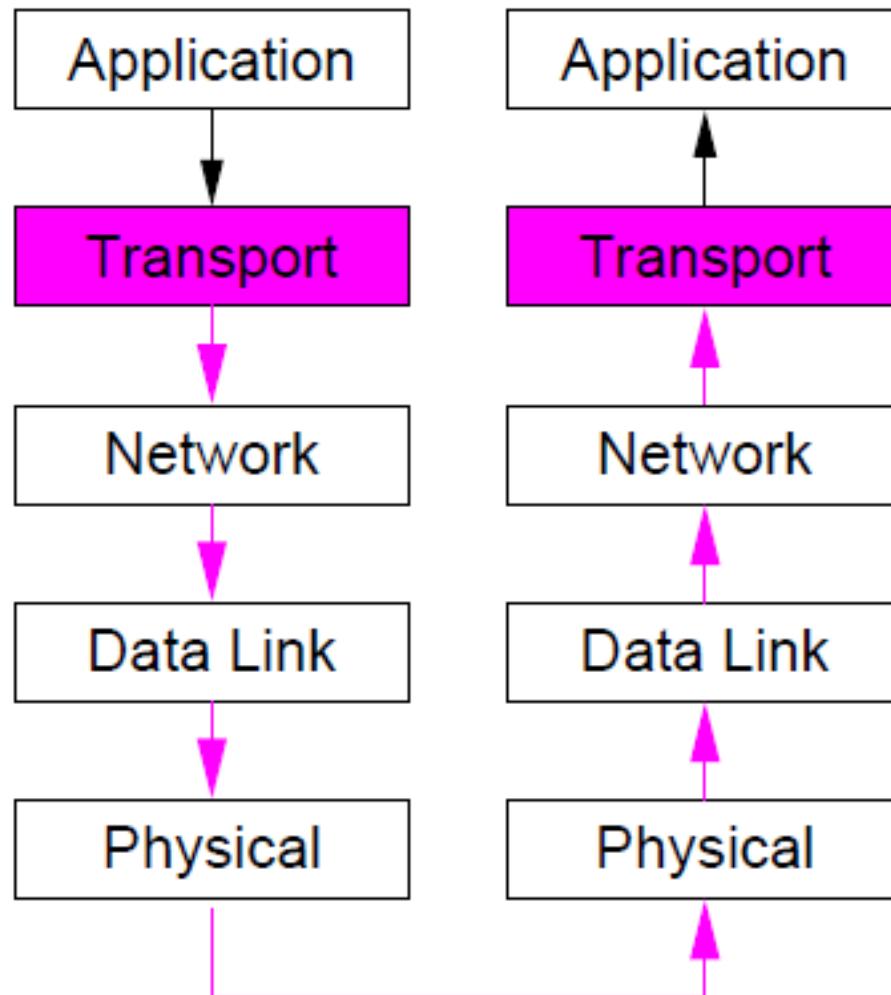
Network layer: protect end-to-end between systems



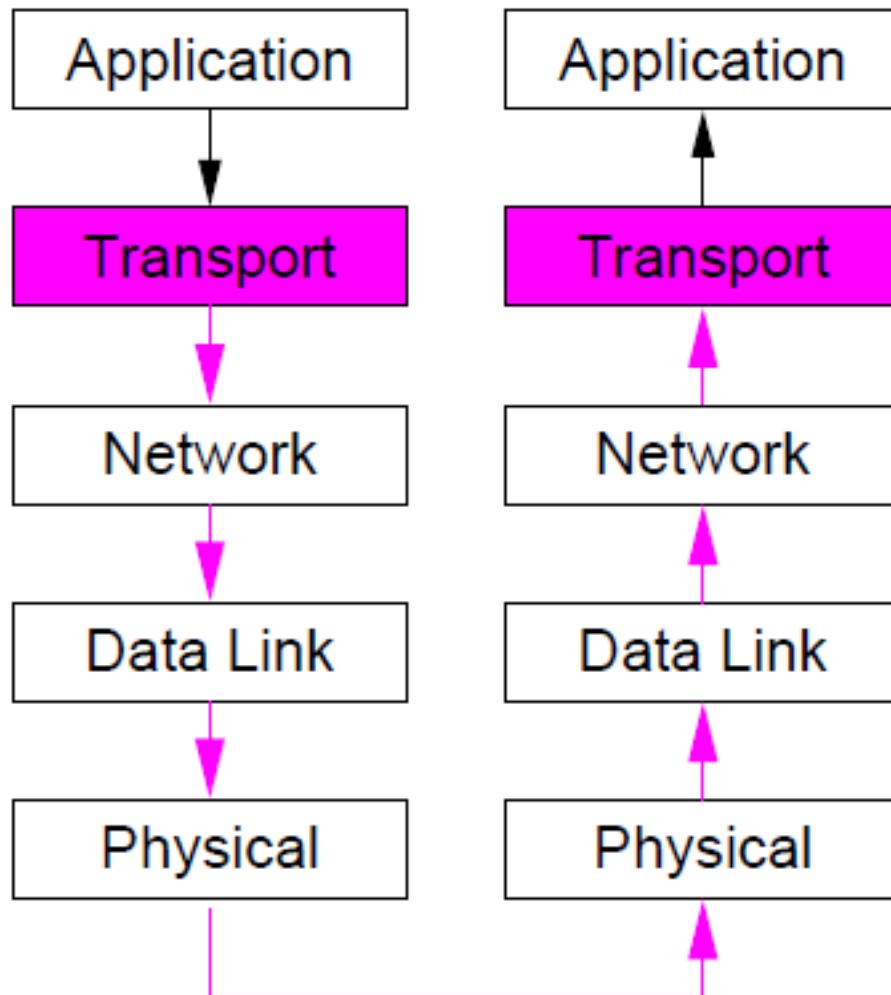
- Confidentiality: between hosts/sites
- Integrity: between hosts/sites
- Authentication: for host or site
- Replay protection: between hosts/sites
- Traffic analysis protection: host/site information exposed
- Access control: to host/site
- Transparency user and SW transparency possible
- Flexibility: may need HW or SW modifications
- Simplicity: good for site-to-site, not good for host-to-host



Transport layer: Protection end-to-end between processes

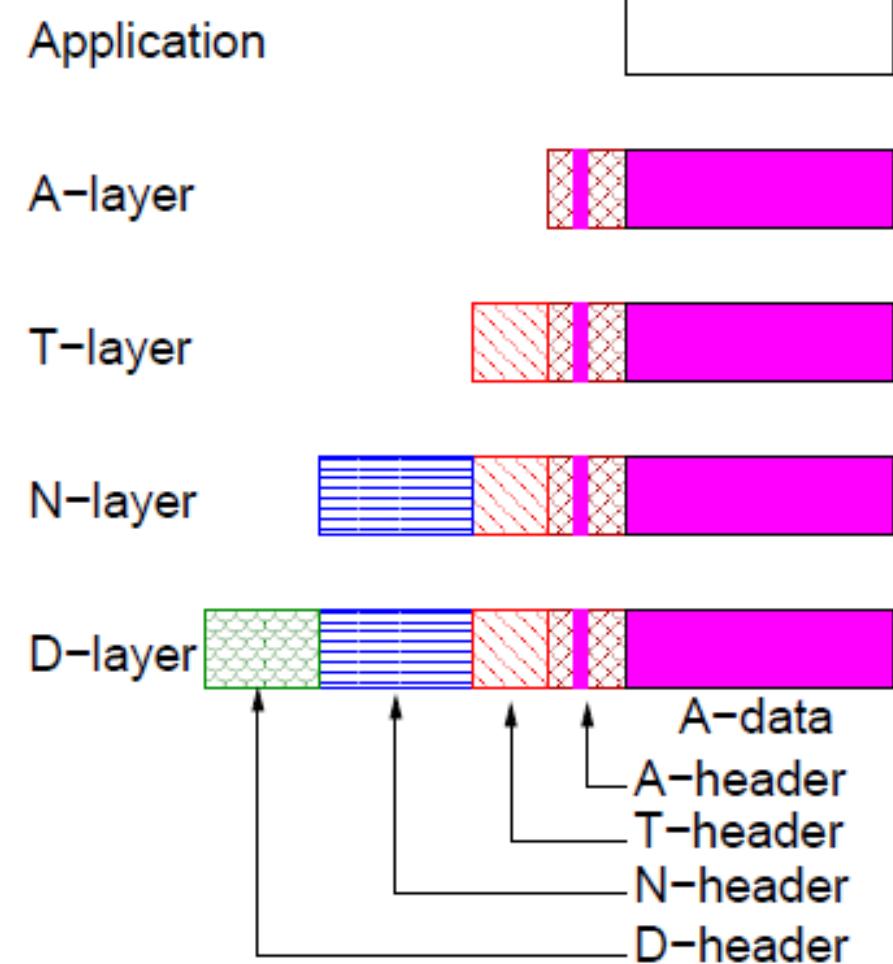
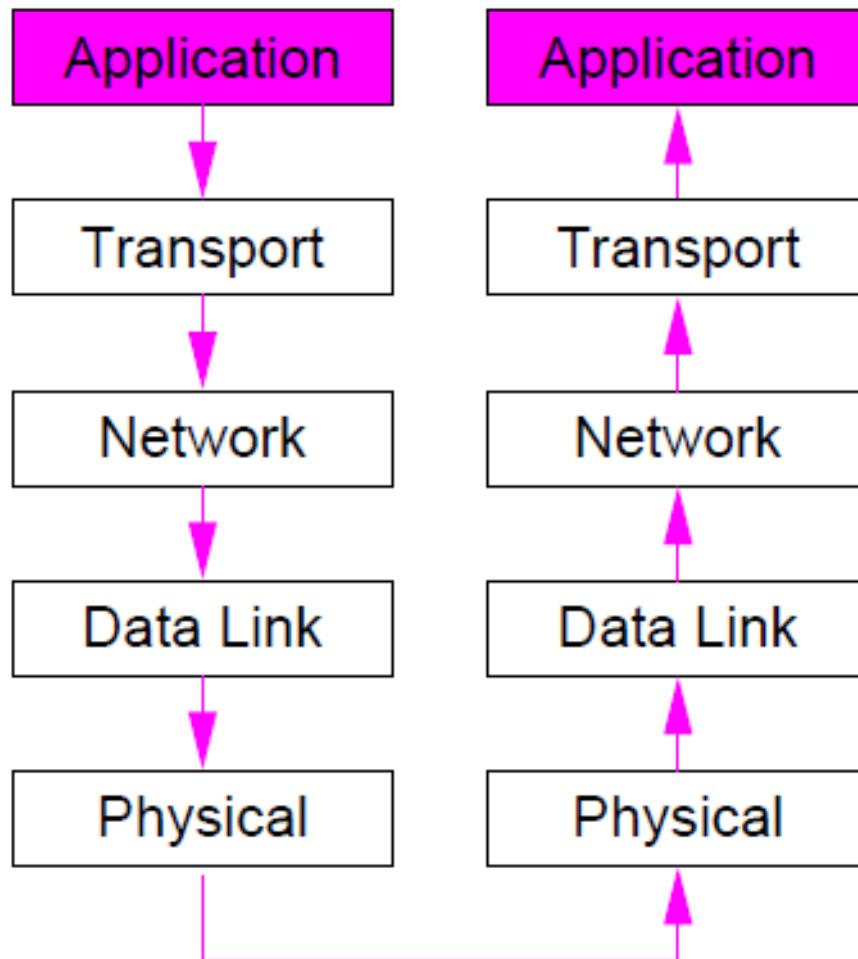


Transport layer: Protection end-to-end between processes

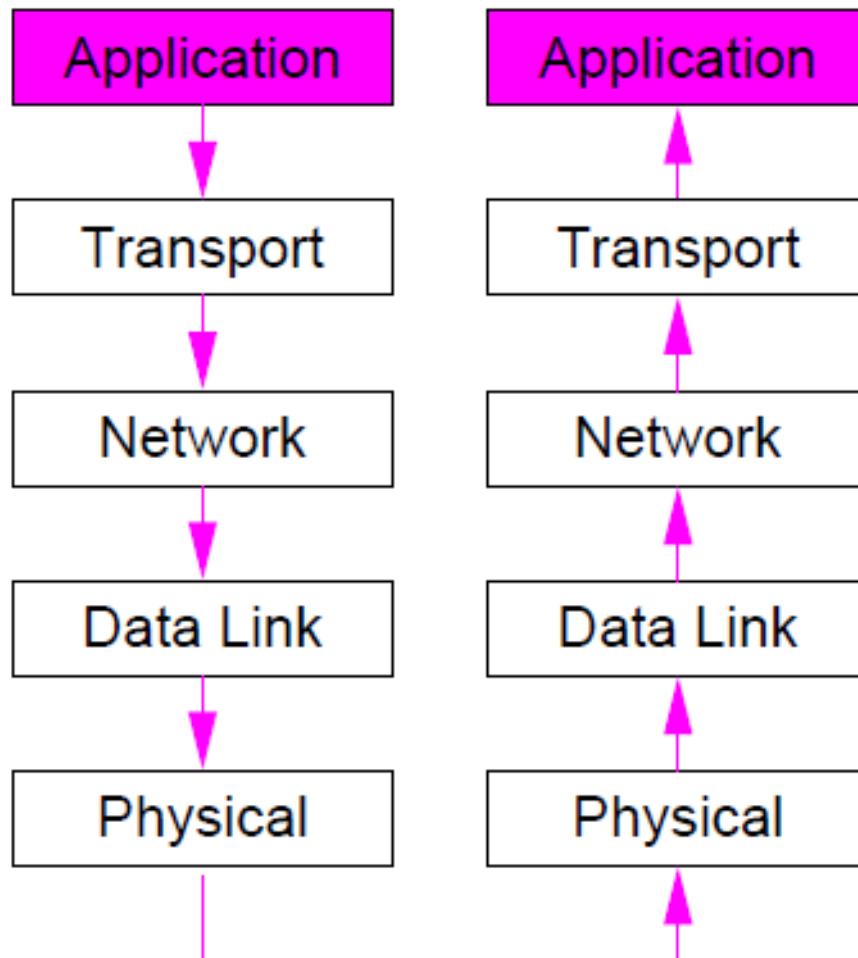


- Confidentiality: between apps/hosts/sites
- Integrity: between apps/hosts/sites
- Authentication: for user, host, site
- Replay protection: between apps/hosts/sites
- Traffic analysis protection: protocol/host/site info. exposed
- Access control: user/host/site
- Transparency user and SW transparency possible
- Flexibility: HW or SW modifications
- Simplicity: good for site-to-site, not good for host-to-host

Application layer: Security for a single application



Application layer: Security for a single application



- Confidentiality: between users/apps
- Integrity: between users/apps
- Authentication: user
- Replay protection: between apps
- Traffic analysis protection: all but data exposed
- Access control: only data access secured
- Transparency: only user transparency
- Flexibility: SW modifications
- Simplicity depends on application



VPN: then?

- It looks best to introduce security in the
 - Transport layer
 - Network layer
- These are the most popular choices for VPNs
- Other options:
 - Secure Application layer protocols: only protect a single application, but are often used for specialized purposes, e.g. S/MIME or PGP for secure e-mail
 - Secure Data Link layer protocols: are mostly used with PPP or other modem-based communication. e.g. PPTP, L2TP, LTF



VPN device placement



SSL VPN Device placement

- Device placement is a challenge because it affects:
 - Security
 - Functionality
 - Performance
- Main options for placement:
 - VPN functionality in firewall
 - VPN device in internal network
 - Single-interface VPN device in DMZ
 - Dual-interface VPN device in DMZ
- Remember: Cryptographic protection only extends from VPN client systems to the SSL VPN device.



SSL: the Client's Knowledge of the Server

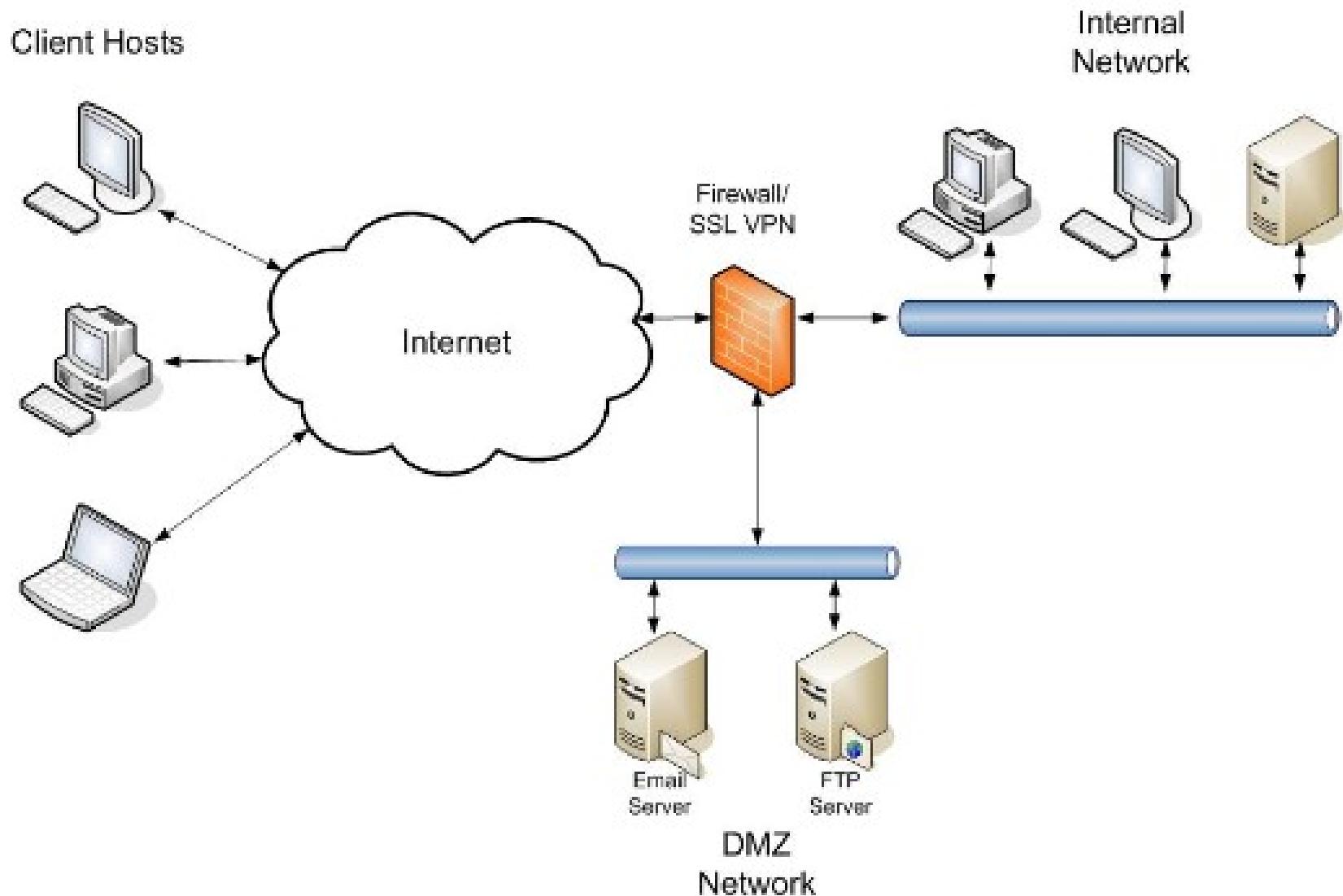
- The client receives the server's certificate
- Does it help?
- A certificate means that someone attests to binding some name to a public key.
- Who has done the certification? Is it the correct name?
- Every browser has a list of built-in certificate authorities
 - Hundreds of certificate authorities! Do you trust them all to be honest and competent? Do you even know them all?
- It's all a matter of trust...



Conclusions on SSL

- The cryptography itself seems correct
 - The human factors are dubious
 - Most users don't know what a certificate is, or how to verify one
- Even when they do know, it's hard to know what it should say in any given situation
- There is no rational basis for deciding whether or not to trust a given CA

Firewall with an SSL VPN

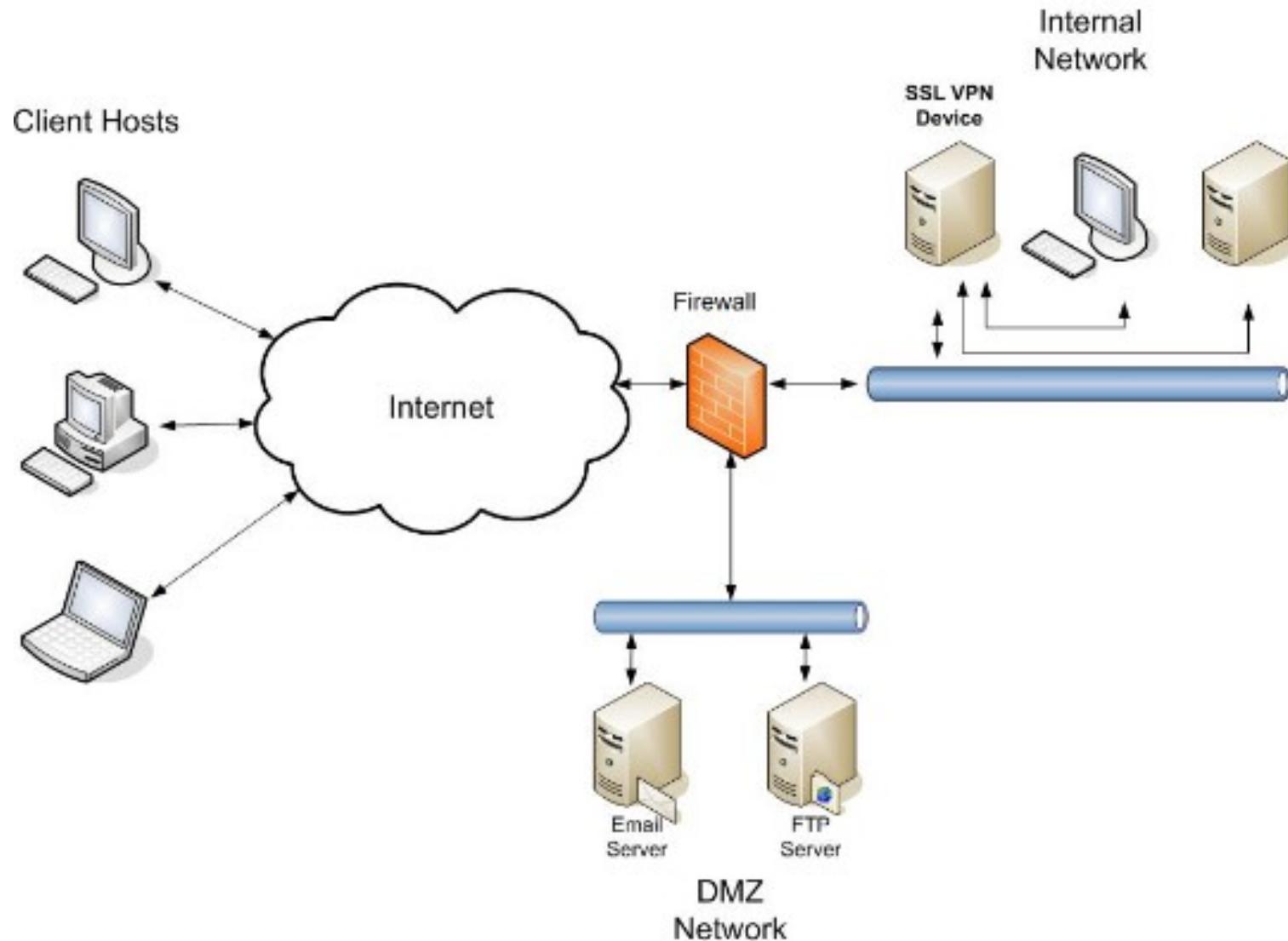




VPN-enabled firewall

- The VPN device communicates directly with internal hosts
- Advantages
 - No holes in FW between external VPN device and internal network.
 - Traffic between device and internal network must go through FW.
 - Simple network administration since only one “box” to administer.
- Disadvantages
 - Limited to VPN functionality offered by FW vendor.
 - FW directly accessible to external users via port 443.
 - Adding VPN functionality to FW can introduce vulnerabilities.
- Note: TCP port 443 (standard) must be open on external FW interface, so clients can initiate connections.

SSL VPN in internal network



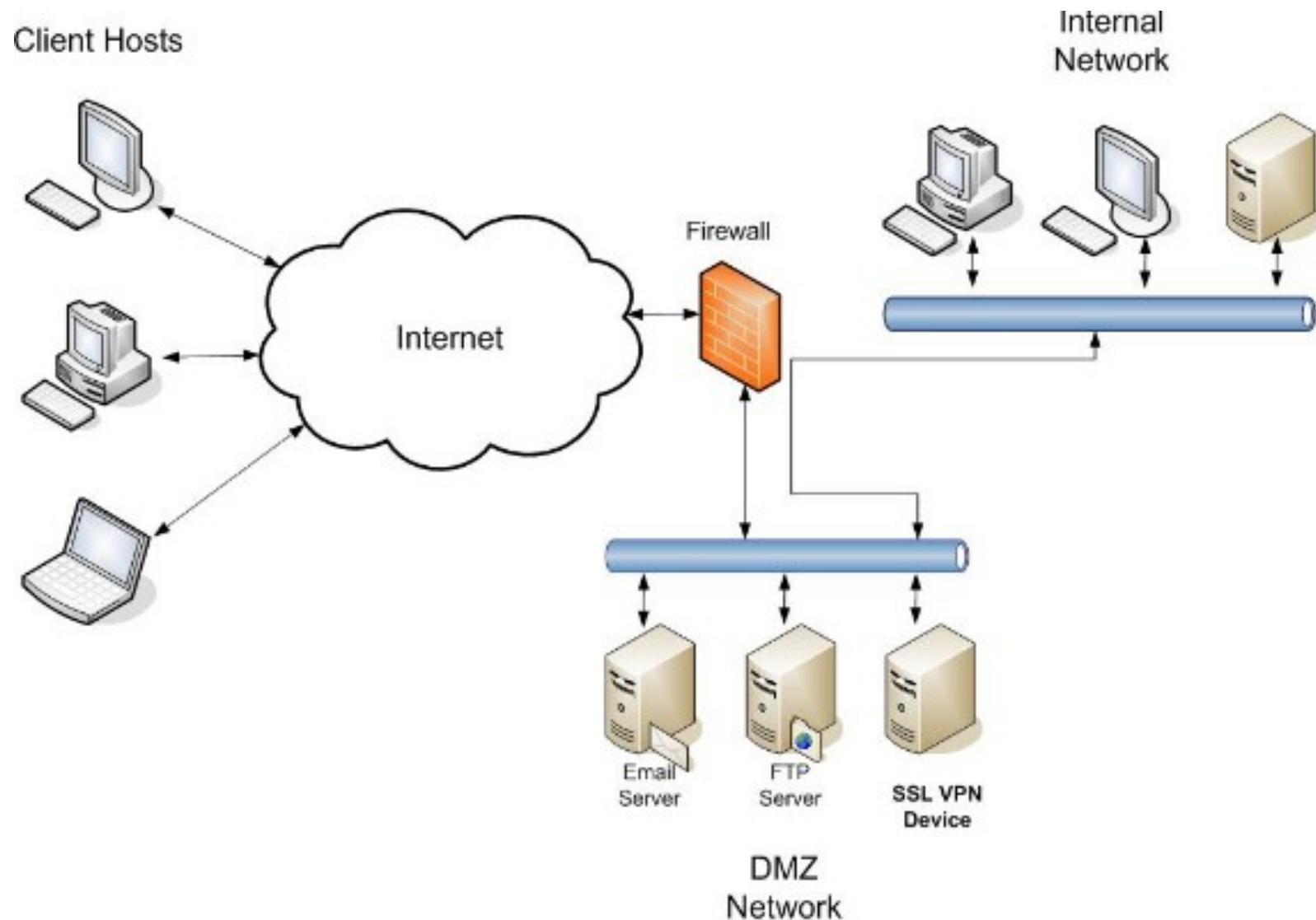


VPN internal

- Advantages
 - Only single rule for single address to be added to FW.
 - No “holes” needed in FW between VPN device and internal network.
 - VPN traffic is behind FW, so protected from attacks by machines in DMZ.
- Disadvantages
 - VPN traffic passes through FW on tunnel, so it is not analyzed.
 - Unsolicited traffic can be sent into internal network from outside to internal VPN device.
 - Internal network is compromised if VPN device is compromised.
- Note: TCP port 443 (standard) opened on FW for the address of the device.



SSL VPN In DMZ

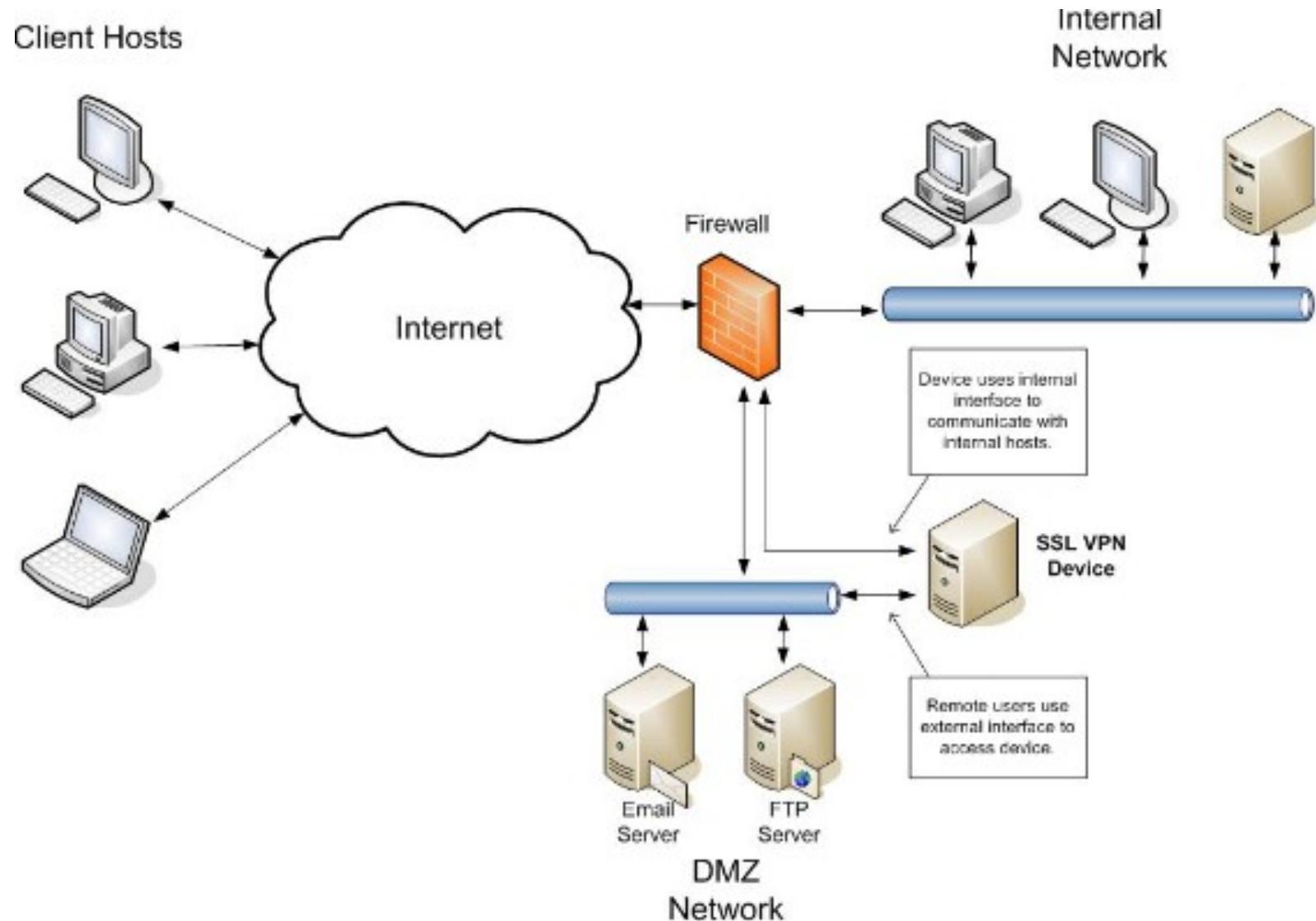




DMZ with VPN

- Advantages
 - Internal network protected against compromised VPN device.
 - Traffic between device and internal network must go through FW.
 - IDS in DMZ can analyze traffic destined for internal network.
- Disadvantages
 - Numerous ports open in FW between device and internal hosts.
 - Decrypted traffic from device to internal network must be sent through DMZ.
 - FW bypassed when user traffic is destined for hosts in DMZ.
- Note: TCP port 443 (standard) opened on FW for the address of the device

Dual interfaces VPN device in DMZ





VPN with two interfaces in DMZ

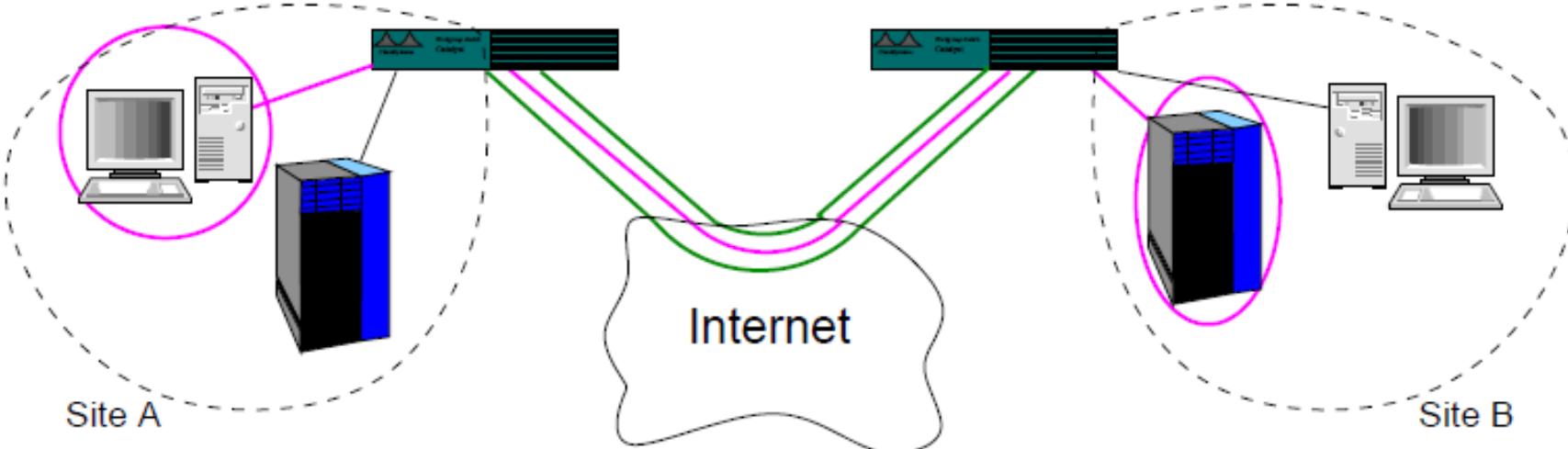
- Clients connect to external device interface, internal traffic uses internal interface.
- Advantages
 - All advantages of placing VPN device DMZ.
 - Unencrypted traffic to internal hosts is protected from other hosts in DMZ.
 - Only FW interface connected to device's internal interface needs to permit traffic from VPN device.
- Disadvantages
 - Numerous ports open in FW between device and internal hosts.
 - May introduce additional routing complexity.
 - FW bypassed if split tunneling is not used and user traffic is destined for hosts in DMZ



SSL Tunneling

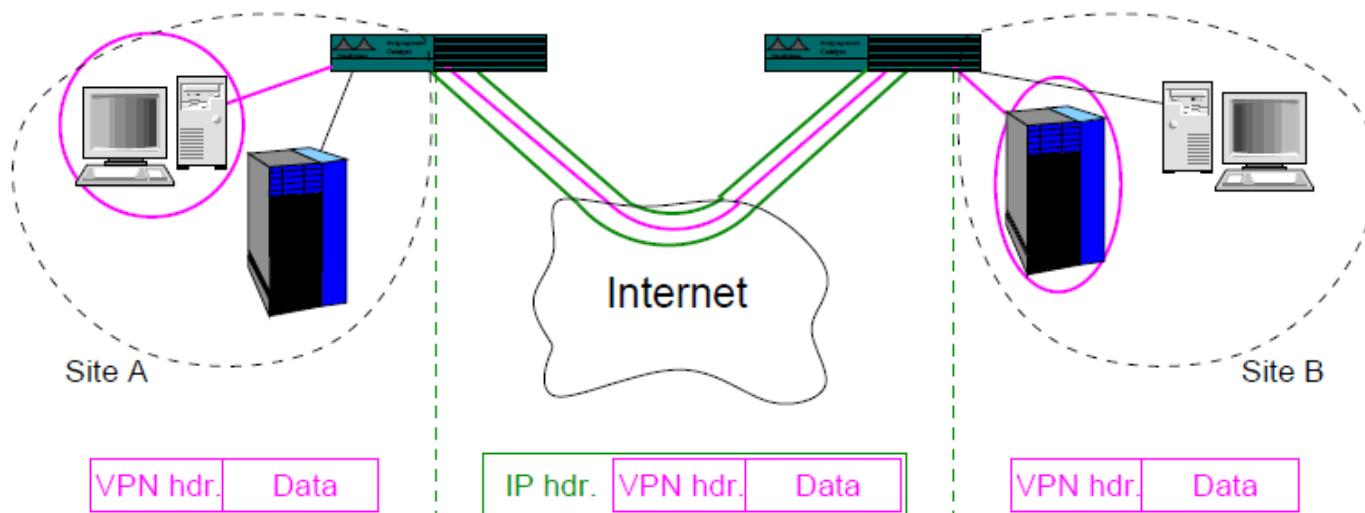
Tunneling

- Operation of a network connection on top of another network connection
- It allows two hosts or sites to communicate through another network that they do not want to use directly

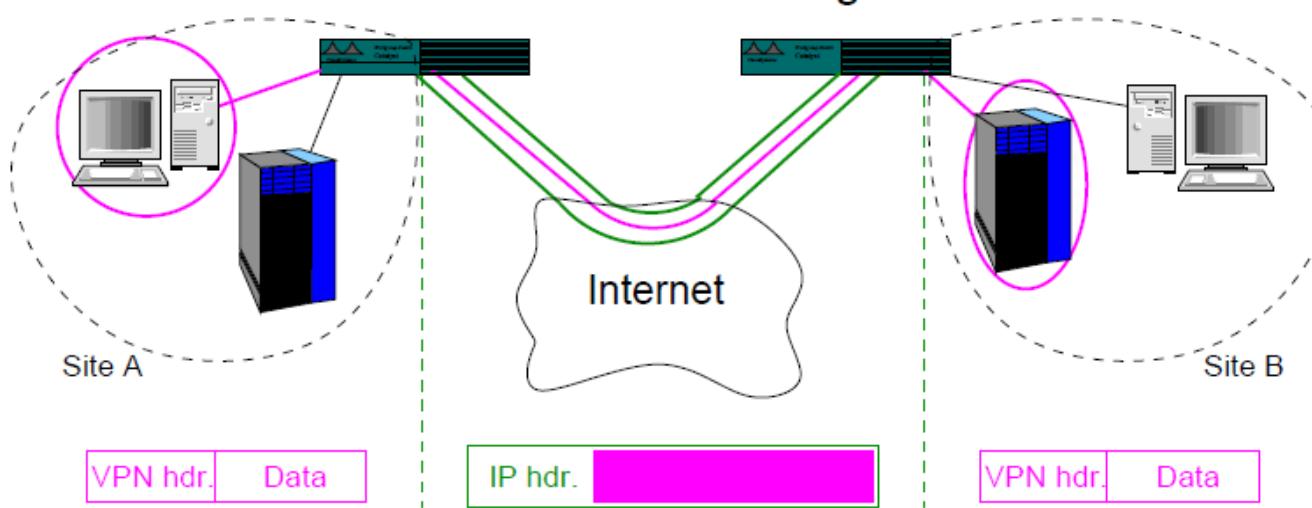


Site-to-site tunneling

- Enables a PDU to be transported from one site to another without its contents being processed by hosts on the route.
- Idea: Encapsulate the whole PDU in another PDU sent out on the network connecting the two sites.
 - Encapsulation takes place in edge router on src. site.
 - Decapsulation takes place in edge router on dst. site.
- Note that the host-to-host communication does not need to use IP



Secure tunneling



- Enables a PDU to be transported from one site to another without its contents being seen or changed by hosts on the route.
- Idea: Encrypt the PDU, and then encapsulate it in another PDU sent out on the network connecting the two sites.
 - Encryption can take place in edge router on src. site.
 - Decryption can take place in edge router on dst. site.
- Note: dst. address in IP header is for dst. edge router.



Tunneling for VPNs

- Tunneling offers the basic method for providing a VPN.
- Where in the network architecture to initiate and terminate the tunnel:
 - Router/firewall?
 - Special box?
 - Host?
 - Application?
- Which layer to do the tunneling in:
 - Transport layer?
 - Network layer?
- Other possibilities (see previous discussion)
- And of course: Is tunneling the only possible technique?



Secure Socket Layer

- SSL 3.0 has become TLS standard (RFC 2246) with small changes
- Applies security in the Transport layer.
- Originally designed (by Netscape) to offer security for client-server sessions.
- If implemented on boundary routers (or proxies), can provide a tunnel between two sites – typically LANs.
- Placed on top of TCP, so no need to change TCP/IP stack or OS.
- Provides secure channel (byte stream)
 - Any TCP-based protocol
 - <https://> URLs, port 443
 - NNTP, SIP, SMTP...
- Optional server authentication with public key certificates
 - Common on commercial sites

How HTTPS (HTTP on top of TLS) works

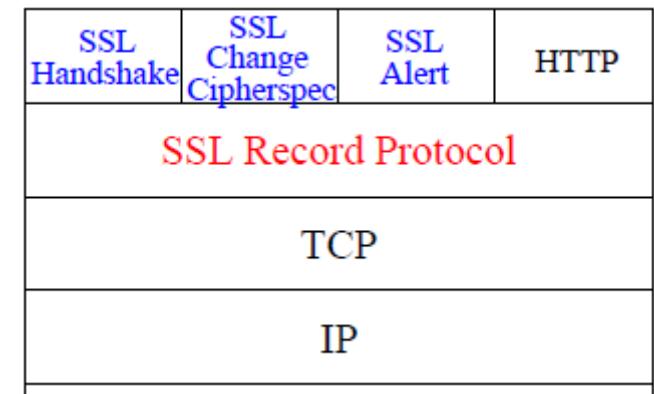
Asymmetric cryptography





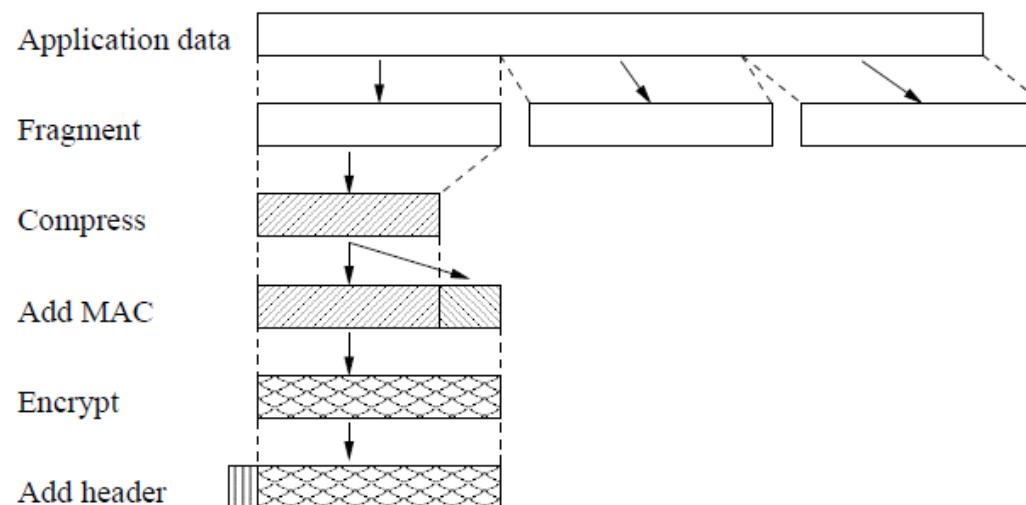
SSL protocol Architecture

- Adds extra layer between T- and A-layers, and extra elements to A-layer
- Record Protocol: Protocol offering basic encryption and integrity services to applications
- Application Protocols: control operation of the record protocol
 - Handshake: Used to authenticate server (and optionally client) and to agree on encryption keys and algorithms.
 - Change cipher spec: Selects agreed keys and encryption algorithm until further notice.
 - Alert: Transfers information about failures.

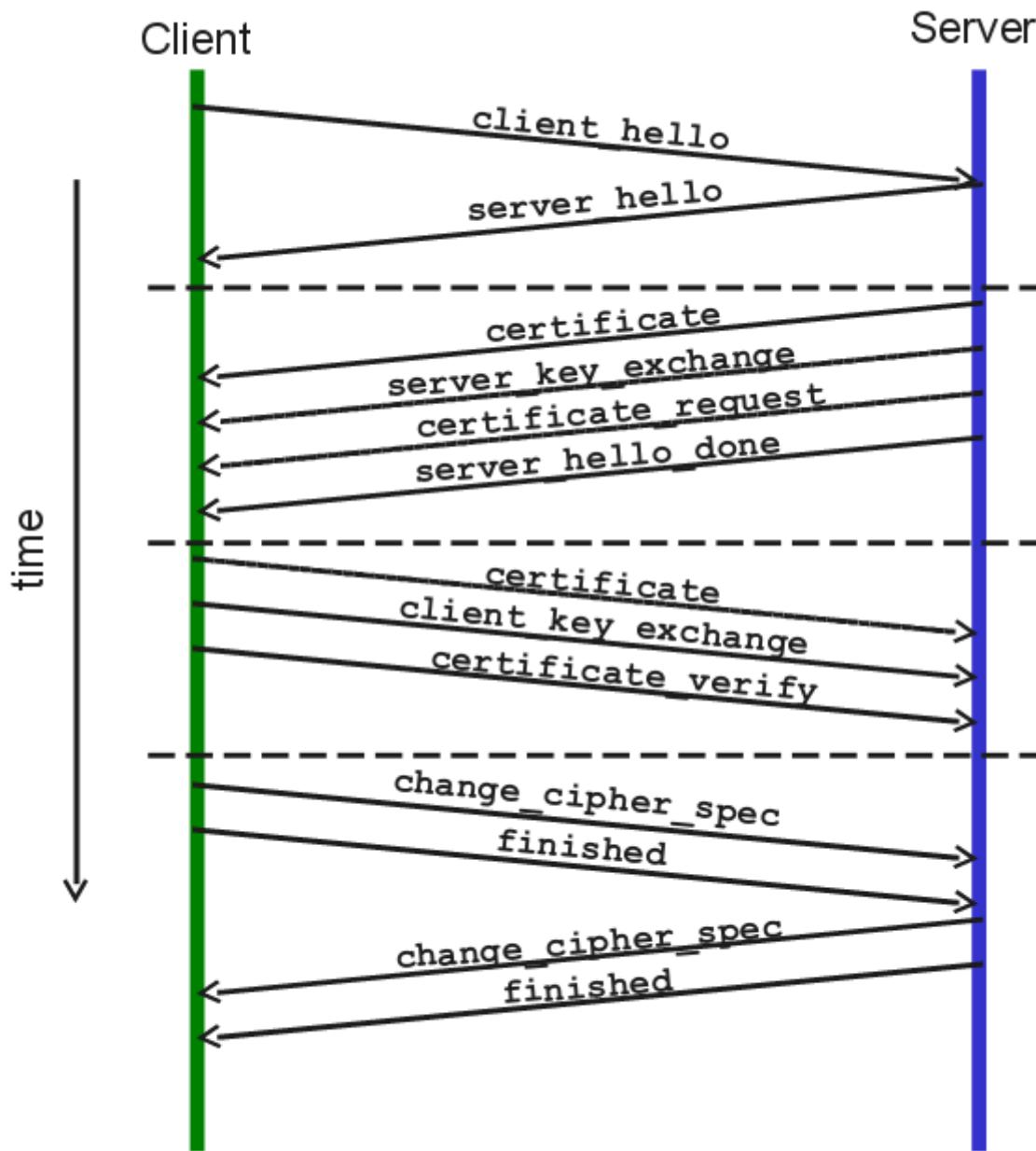


SSL/TLS Record Protocol

- Offers to apply the following steps to PDUs:
 - 1 Fragmentation into blocks of $\leq 2^{14}$ bytes.
 - 2 (optional) Lossless compression.
 - 3 Addition of a keyed MAC, using a shared secret MAC key.
 - 4 Encryption, using a shared secret encryption key.
 - 5 Addition of header indicating Application protocol in use.



SSL Handshake



1) Hello phase

2) Server authentication

3) Client authentication

4) Finish



SSL/TLS Handshake Protocol

4-phase “Client/Server” protocol to establish parameters of the secure connection (“Client” is the initiator):

- 1) Hello:** Establishment of security capabilities: Client sends list of possibilities, in order of preference. Server selects one, and informs Client of its choice. Parties also exchange random noise for use in key generation.
- 2) Server authentication and key exchange:** Server executes selected key exchange protocol (if needed). Server sends authentication info. (e.g. X.509 cert.) to Client.
- 3) Client authentication and key exchange:** Client executes selected key exchange protocol (mandatory). Client sends authentication info. to Server (optional).
- 4) Finish:** Shared secret key is derived from pre-secrets exch. in 2, 3. Change Cipher Spec. protocol is activated. Summaries of progress of Handshake Protocol are exchanged and checked by both parties.



SSL/TLS Security Capabilities

- Conventionally expressed by a descriptive string, specifying:
 - Version of SSL/TLS
 - Key exchange algorithm
 - Grade of encryption (previous to TLSv1.1)
 - Encryption algorithm
 - Mode of block encryption (if block cipher used)
 - Cryptographic checksum algorithm
- Example: TLS_RSA_WITH_AES_128_CBC_SHA
 - TLS → (Latest version of) TLS
 - RSA → RSA key exchange
 - WITH → (merely filler...)
 - AES_128 → 128-bit AES encryption
 - CBC → Cipher Block Chaining
 - SHA → Use HMAC-SHA digest



Key exchange and authentication

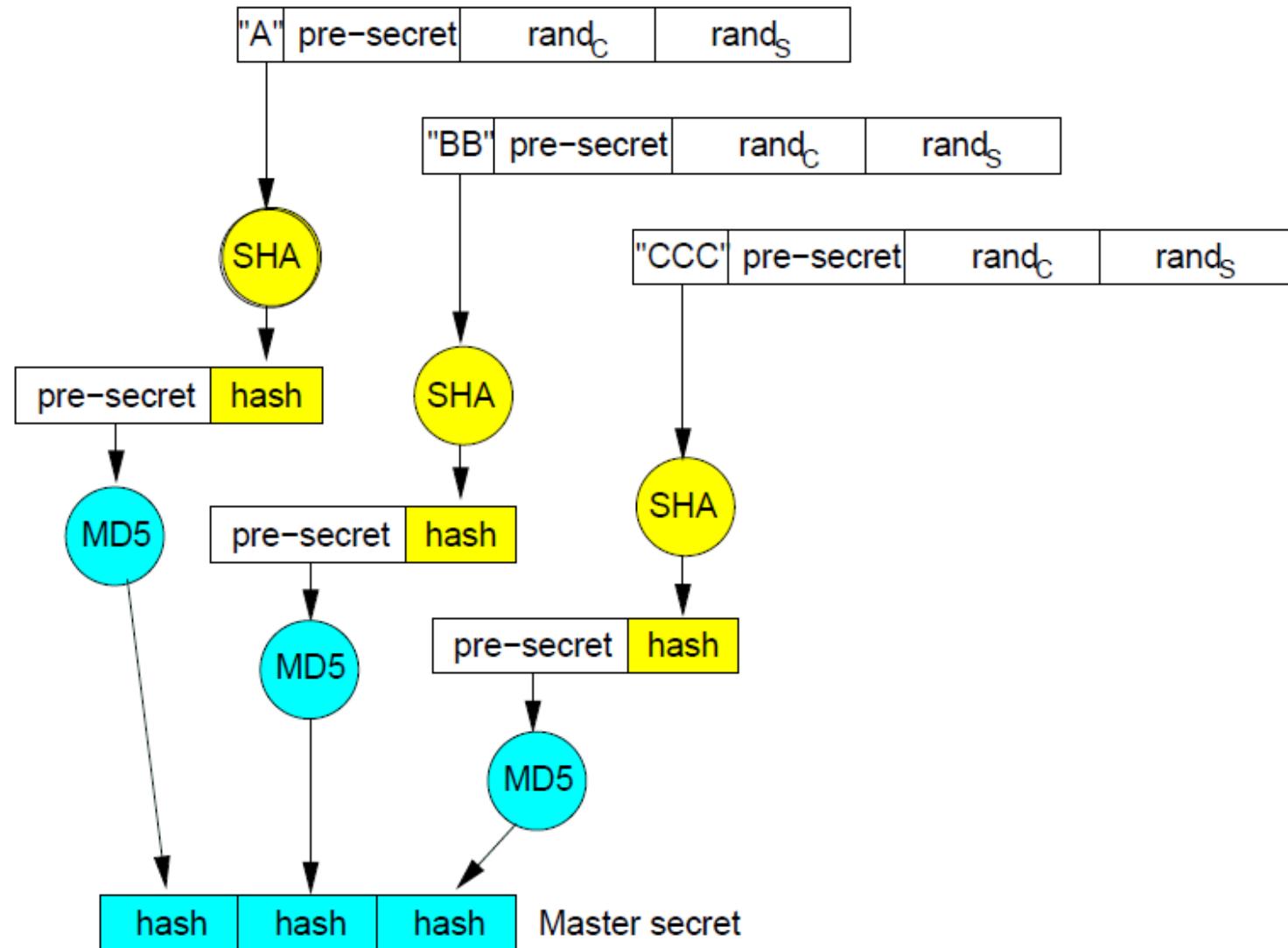
Possible ways of agreeing on secrets in TLS are:

- RSA: RSA key exch. (secret encrypted with recipient's publ. key)
- DHE RSA: Ephemeral Diffie-Hellman with RSA signatures
- DHE DSS: Ephemeral Diffie-Hellman with DSS signatures
- DH DSS: Diffie-Hellman with DSS certificates
- DH RSA: Diffie-Hellman with RSA certificates
- DH anon: Anonymous Diffie-Hellman (no authentication)
- NULL No key exch.

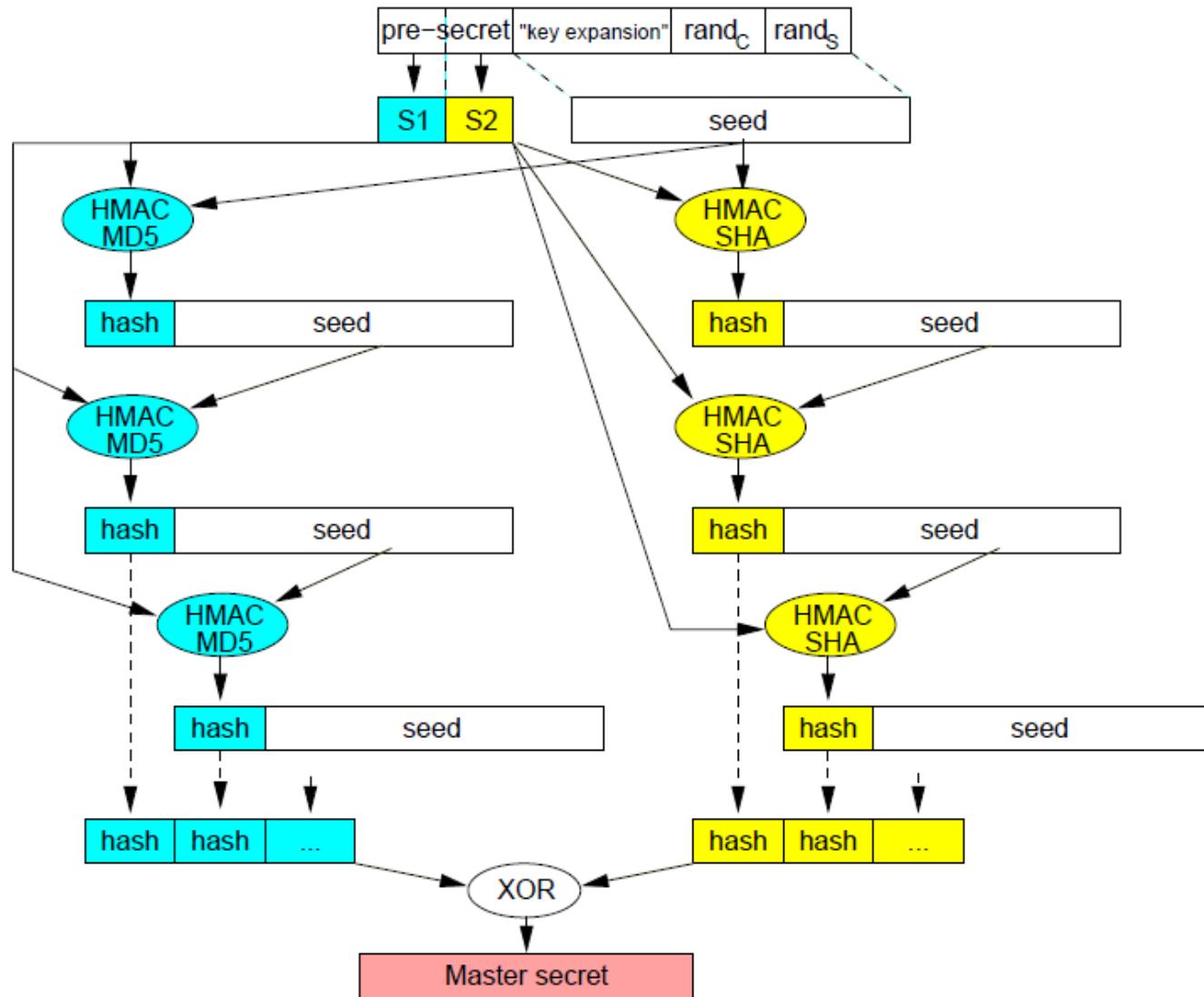
Variant: If followed by "EXPORT_", weak encryption is used. (This option only available prior to TLSv1.1)

- **Note:** "Key exchange" only establishes a pre-secret! From this, a master secret is derived by a pseudo-random function (PRF). Shared secret encryption key is derived by expansion of master secret with another PRF. (In TLS several keys are derived for different purposes.)

SSL Master Secret



TLS Master Secret





SSL/TLS Heartbeat

- It is an extension (RFC 6520) that allows to keep an established session alive
 - That is, as soon as the data exchange between two endpoints terminates, the session will also terminate
- To avoid the re-negotiation of the security parameters for establishing a secure session, we can keep using the same parameters even if there is no exchange of data
- It introduces two messages: **HeartbeatRequest** and **HeartbeatResponse**



Heartbeat exchange

- When one endpoint sends a HeartbeatRequest message to the other endpoints, the former also starts what is known as the **retransmit timer**
 - During the time interval of the retransmit timer, the sending endpoint will not send another HeartbeatRequest message.
- An SSL/TLS session is considered to have terminated in the absence of a HeartbeatResponse packet within a time interval



Heartbeat payload

- As a protection against a replay attack, HeartbeatRequest packets include a payload that must be returned without change by the receiver in its HeartbeatResponse packet
- The Heartbeat message is defined as

```
struct {  
    HeartbeatMessageType type;  
    uint16 payload_length;  
    opaque payload[HeartbeatMessage.payload_length];  
    opaque padding[padding_length];  
} HeartbeatMessage;
```



Heartbleed bug

- Bug in OpenSSL library (4/4/2014)
- The receiver of request did not check that the **size of the payload in the packet** actually equaled the **value given by the sender to the payload length field** in the request packet
 - The attacker sends little data but sets the size to **max**
 - The receiver allocates that amount of memory for the response and copied **max bytes** from the mem location where the request packet was received
 - Then, the actual payload returned could potentially include objects in the memory that **had nothing to do** with the received payload
 - Objects could be private keys, passwords, and such...



SSL VPN Architecture

- Two primary models:
- 1 SSL Portal VPN
 - Allow remote users to:
 - Connect to VPN gateway from a Web browser
 - Access services from Web site provided on gateway
- 2 SSL Tunnel VPN
 - Allow remote users to:
 - Access network protected by VPN gateway from
 - Web browser allowing active content.
 - More capabilities than portal VPNs, as easier to provide more services.



SSL VPN functionalities

Most SSL VPNs offer one or more core functionalities:

- **Proxying:** Intermediate device appears as true server to client. E.g. Web proxy.
- **Application Translation:** Conversion of information from one protocol to another.
 - e.g. Portal VPN offers translation for applications which are not Web-enabled, so users can use Web browser to access applications with no Web interface.
- **Network Extension:** Provision of partial or complete network access to remote users, typically via Tunnel VPN.
 - Two variants:
 - Full tunneling: All network traffic goes through tunnel.
 - Split tunneling: Organisation's traffic goes through tunnel, other traffic uses remote user's default gateway.



SSL VPN Security Services

Typical services include:

- **Authentication** Via strong authentication methods, such as two-factor authent., X.509 certificates, smartcards, security tokens etc. May be integrated in VPN device or external authent. server.
- **Encryption and integrity protection:** Via the use of the SSL/TLS protocol.
- **Access control:** May be per-user, per-group or per-resource.
- **Endpoint security controls:** Validate the security compliance of clients attempting to use the VPN.
 - e.g. presence of antivirus system, updated patches etc.
- **Intrusion prevention:** Evaluates decrypted data for malicious attacks, malware etc.



SSL again

- Crypto is insufficient for Web security
- One issue: linkage between crypto layer and applications
- Trust: what does the server really know about the client?
 - Unless client-side certificates are used, absolutely nothing
 - SSL provides a secure pipe. “Someone” is at the other end; you don’t know who
 - Usually there is no user authentication in SSL, but in the application layer!



IPSec



IPsec

- A Network Layer protocol suite for providing security over IP.
- Part of IPv6; an add-on for IPv4.
- Can handle all three possible security architectures:

Feature	Gateway-to-Gateway	Host-to-Gateway	Host-to-Host
Protection between client and local gateway	No	N/A (client is VPN endpoint)	N/A (client is VPN endpoint)
Protection between VPN endpoints	Yes	Yes	Yes
Protection between remote gateway and remote server (behind gateway)	No	No	N/A (client is VPN endpoint)
Transparency to users	Yes	No	No
Transparency to users' systems	Yes	No	No
Transparency to servers	Yes	Yes	No



IPsec services

- Basic functions, provided by separate (sub-)protocols:
 - Authentication Header (AH): Support for data integrity and authentication of IP packets.
 - Encapsulated Security Payload (ESP): Support for encryption and (optionally) authentication.
 - Internet Key Exchange (IKE): Support for key management etc.

Service	AH	ESP (encrypt only)	ESP(encrypt+authent.)
Access Control	+	+	+
Connectionless integrity	+		+
Protection between VPN endpoints	+		+
Data origin authentication	+		+
Reject replayed packets		+	+
Payload confidentiality		+	+
Metadata confidentiality		partial	partial
Traffic flow confidentiality		(*)	(*)



IPsec Security Associations

- Think of it as an IPsec connection: all of the parameters needed, like crypto algorithms (AES, SHA1, etc.), modes of operation (CBC, HMAC, etc.), key lengths, traffic to be protected, etc.
- Both sides must agree on the SA for secure communications to work
- For a two-way communication, two SAs must be defined.
- SA parameters must be negotiated (using IKE) between sender and receiver before secure communication can start.
- Each SA is identified by:
 - Security Parameters Index (SPI): 32-bit integer chosen by sender. Enables receiving system to select the required SA.
 - Destination Address: Only unicast IP addresses allowed!
 - Security Protocol Identifier: AH or ESP.



IPsec modes

- Transport Mode
 - Provides protection for a T-layer packet embedded as payload in an IP packet.
- Tunnel Mode
 - Provides protection for an IP packet embedded as payload in an IP packet.

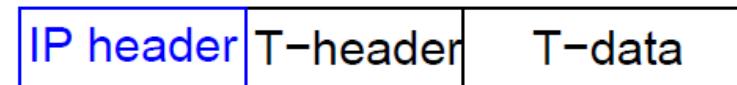
	Transport Mode SA	Tunnel Mode SA
AH	Authenticate IP payload and selected parts of IP header and IPv6 extension headers.	Authenticate entire inner IP packet and selected parts of outer IP header and outer IPv6 extension headers.
ESP	Encrypt IP payload + any IPv6 extension headers after ESP header.	Encrypt inner IP packet.
ESP + authent.	Encrypt IP payload + any IPv6 extension headers after ESP header. Authenticate IP payload.	Encrypt and authenticate inner IP packet.



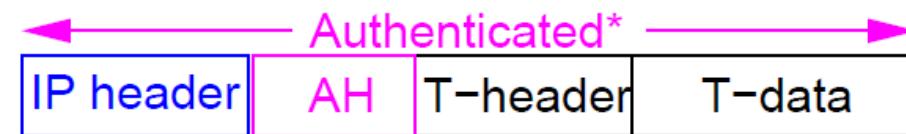
Authentication with IPv4

- AH header inserted after the outermost IP header – depending on whether Transport or Tunnel mode is used.
 - Do not forget that integrity check (and thus authentication) does not cover any mutable, unpredictable header fields.

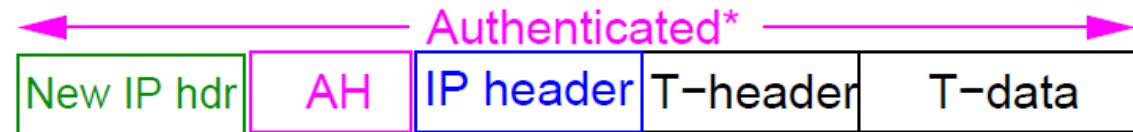
Without IPsec



AH Transport mode



AH Tunnel mode





Authentication with IPv6

- AH header inserted after the outermost IP header – depending on whether Transport or Tunnel mode is used.
 - Do not forget that integrity check (and thus authentication) does not cover any mutable, unpredictable header fields.

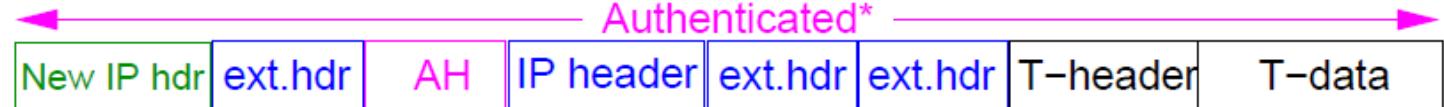
Without IPsec



AH Transport mode



AH Tunnel mode





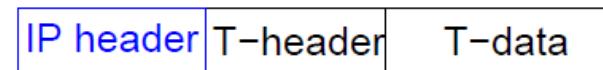
Authentication Header

- One of the (many possible) IP header fields. Contains:
 - Next Header: Type of following header field.
 - Payload Length: (Length - 2), in 32-bit words, of AH.
 - SPI: Identifies SA in use.
 - Sequence Number: Monotonically increasing packet counter value.
 - Authentication Data (AD): (variable length) HMAC based on MD5 or SHA-1 cryptographic hashing algorithm, or AES-CBC, evaluated over:
 - Immutable or predictable IP header fields. (Other fields assumed zero when MAC is calculated.)
 - Rest of AH header apart from AD field.
 - All embedded payload (from T-layer or embedded IP packet), assumed immutable.
- Immutable fields do not change as the packet traverses the network.
 - Example: Source address.
- Mutable but predictable fields may change, but can be predicted.
 - Example: Destination address.
- Mutable, unpredictable fields include Time-to-live, Header checksum.

ESP with IPv4

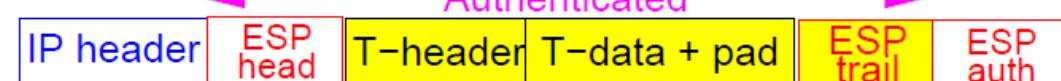
- ESP header inserted after the outermost IP header – depending on whether Transport or Tunnel mode is used:
- Padding is added to end of T-layer payload to give (a certain amount) of traffic analysis protection.
- ESP trailer and (optional) ESP authentication field added after the end of the padded T-layer payload.
- As usual, integrity check (and thus authentication) does not cover any mutable, unpredictable header fields.

Without IPsec



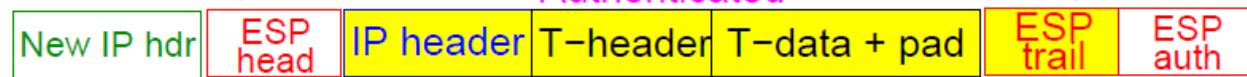
Encrypted
Authenticated*

ESP Transport mode



Encrypted
Authenticated*

ESP Tunnel mode



ESP with IPv6

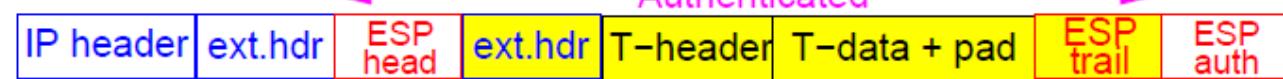
- ESP header inserted after the outermost IP header – depending on whether Transport or Tunnel mode is used:
- Padding is added to end of T-layer payload to give (a certain amount) of traffic analysis protection.
- ESP trailer and (optional) ESP authentication field added after the end of the padded T-layer payload.
- As usual, integrity check (and thus authentication) does not cover any mutable, unpredictable header fields.

Without IPsec



Encrypted
Authenticated*

ESP Transport mode



Encrypted
Authenticated*

ESP Tunnel mode





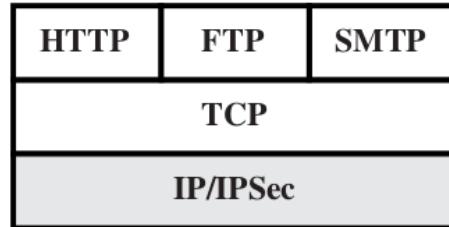
Encryption + Authentication

A common combination, can be achieved by:

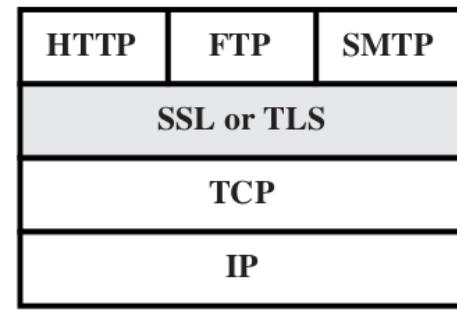
- 1) ESP with Authentication. First apply ESP to data, then add AH field.
Two subcases:
 - 1) Transport mode: E+A apply to IP payload, but IP header not protected.
 - 2) Tunnel mode: E+A apply to entire inner packet.
- 2) Transport Adjacency. Use bundled SAs, first ESP, then AH.
- 3) Encryption covers original IP payload. Authentication covers ESP + original IP header, including source and destination IP addresses
- 4) Transport-Tunnel bundle. Used to achieve authentication before encryption, for example via inner AH transport SA and outer ESP tunnel SA.
- 5) Authentication covers IP payload + IP immutable header. Encryption is applied to entire authenticated inner packet.

IPsec vs TLS

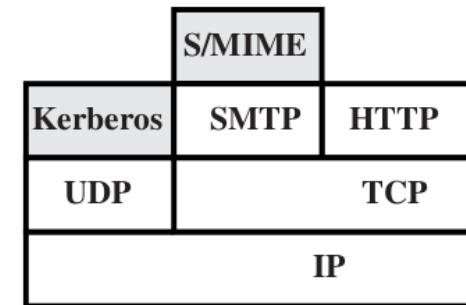
- TLS much more flexible because is in the upper levels
- TLS also provides application end-to-end security, best for web applications → HTTPS
- IPsec has to run in kernel space
- IPsec much more complex and complicated to manage with



(a) Network level



(b) Transport level



(c) Application level



That's all for today

- **Questions?**
- See you on Thursday (12:00)
- Resources:
 - Chapter 24 textbook
 - “Virtual private networking”, Gilbert Held, Wiley ed.
 - http://www.tcpipguide.com/free/t_IPSecurityIPSecProtocols.htm
 - “Guide to IPsec VPNs”, NIST800-77
 - “Guide to SSL VPNs”, NIST-SP800-113

Practical Network Defense

Master's degree in Cybersecurity 2021-22

OpenVPN activity

Angelo Spognardi
[*spognardi@di.uniroma1.it*](mailto:spognardi@di.uniroma1.it)

*Dipartimento di Informatica
Sapienza Università di Roma*



Aim of the lab

- 1) Highlight the tunneling concept
- 2) Use openvpn to make the boundary to be the VPN tunnel end-point
 - The kali/host will have the access to the **internal** subnet passing through an encrypted channel
 - Two scenarios
 - Static key configuration
 - Dynamic key configuration
 - Using certificates



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>
- You can move in the directory and run `lstart`
 - **NOTE:** launch docker first or the first `lstart` attempt can (...will...) fail

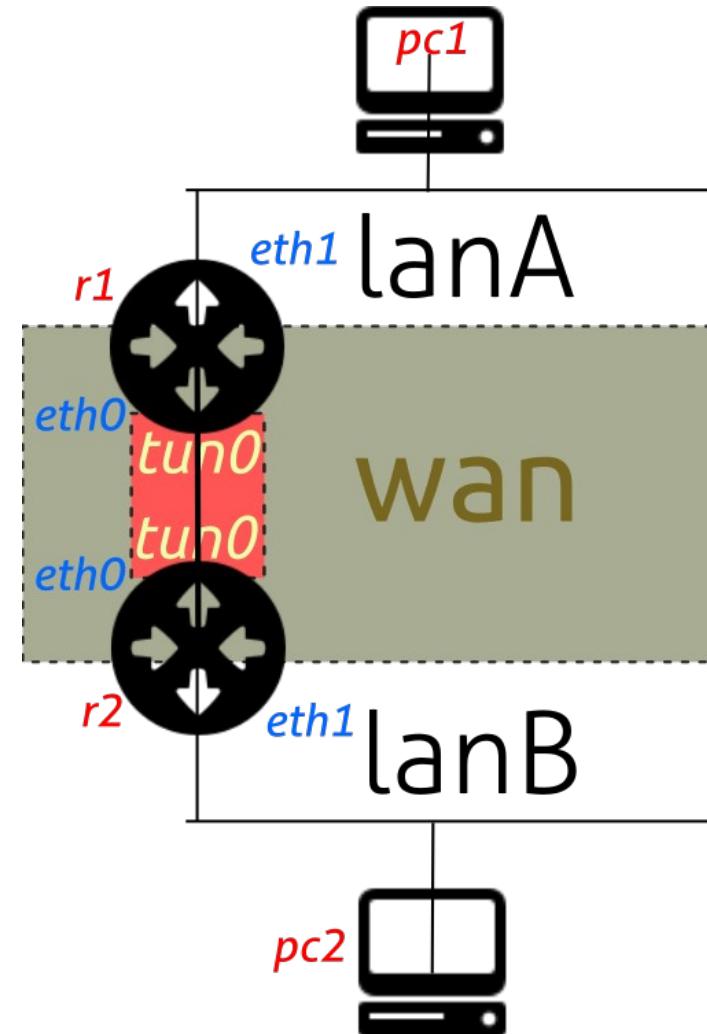


Lab activity: ex1



pnd-labs/lab5/ex1

- You have to setup the IPv4 addressing
 - Follow README instructions



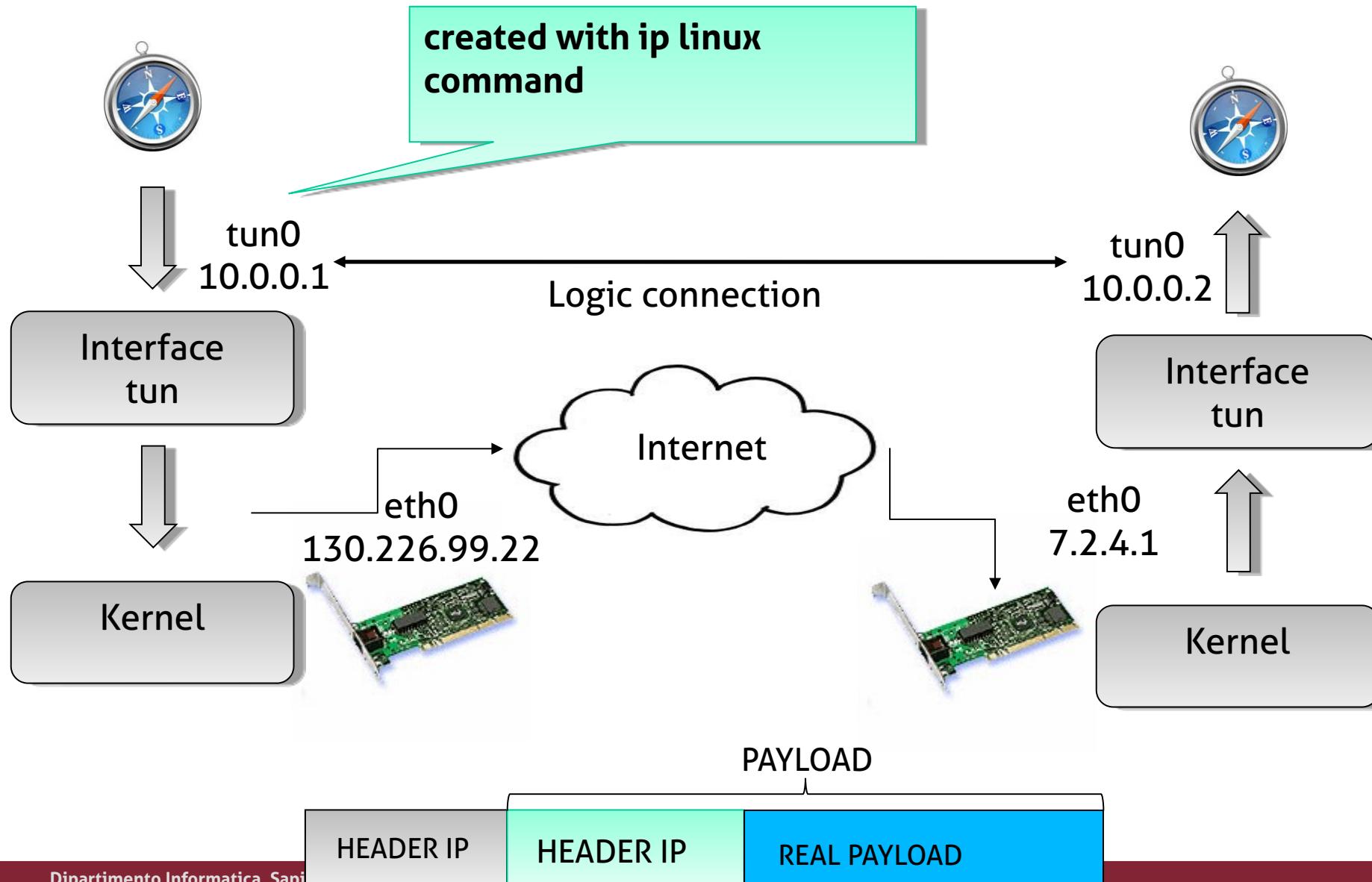


Fundamentals: simple tunneling

- Universal tun/tap drive
 - Creates a virtual interface that encapsulates network traffic
- Any application can use that interface without any need to change its code
- Usually identified with names tun* or tap*
- tun* encapsulate IP layer
- tap* encapsulate Ethernet layer
- Reference:
 - <http://man7.org/linux/man-pages/man8/ip-tunnel.8.html>
 - https://developers.redhat.com/blog/2019/05/17/an-introduction-to-linux-virtual-interfaces-tunnels#ipip_tunnel



Universal tun driver (L3)





Create a tunnel (r1-r2)

- Let's check our local interfaces

```
ip link
```

- Create a new tun virtual interface (use root user)

```
ip tunnel add tun0 mode ipip remote <ipaddressR> local <ipaddressL>
```

- ipaddressR is the IP address of the remote machine
- ipaddressL is the IP address of our local machine
 - alternatively use the `ip link add name tun0 type ipip ...` command
- Let's check again our local interfaces

```
ip link
```

- Let's activate the new interface

```
ip addr add 10.0.0.1/30 dev tun0  
ip link set tun0 up
```

- the IP address of the remote machine **MUST be different!!**



Analyze the traffic

- Open wireshark to sniff the traffic
 - Use tcpdump to save the traffic from different observation points
- Generate some traffic in the tunnel
 - `ping 10.0.0.2`
- What do you notice?
 - See the difference between tun0 and the eth0
- Can you see the basic principle of a VPN?



Use the tunnel to connect lanA and lanB

- You have to setup the routes
- Who will r1 forward the packets for lanB?
 - In order to use the tunnel?
- Properly set up the routing tables
- Check what happens when pc1 tries to reach pc2
 - Use wireshark in r1 or r2
 - You can join the wan network

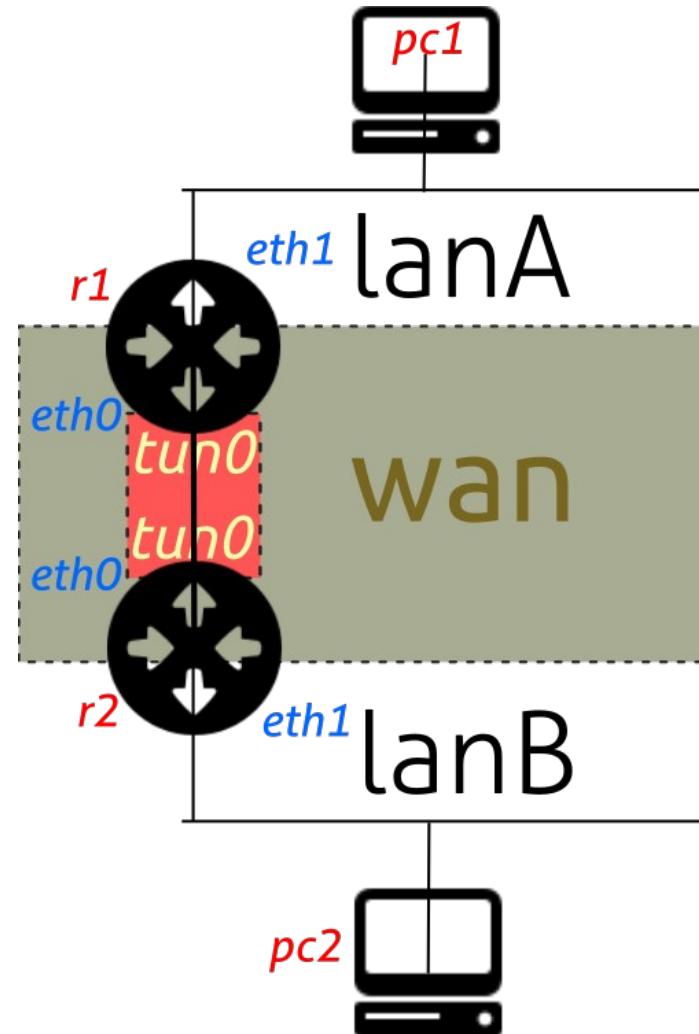


Lab activity: ex2



pnd-labs/lab5/ex2

- Same topology than ex1
- Must run with the option – **privileged**
 - Use `lstart.sh` script
- You have to setup the IPv4 addressing
 - Follow README instructions
- Check that openvpn is installed in both r1 and r2





OpenVPN

- Open-source software to realize VPN, namely encrypted tunnels
- Usually uses UDP with one single port
 - Can also use TCP
- Can be used also through firewalls or NAT
- OpenSSL based
- Multiple modes
 - Static: symmetric shared key
 - Dynamic: Public Key Infrastructure





OpenVPN static key

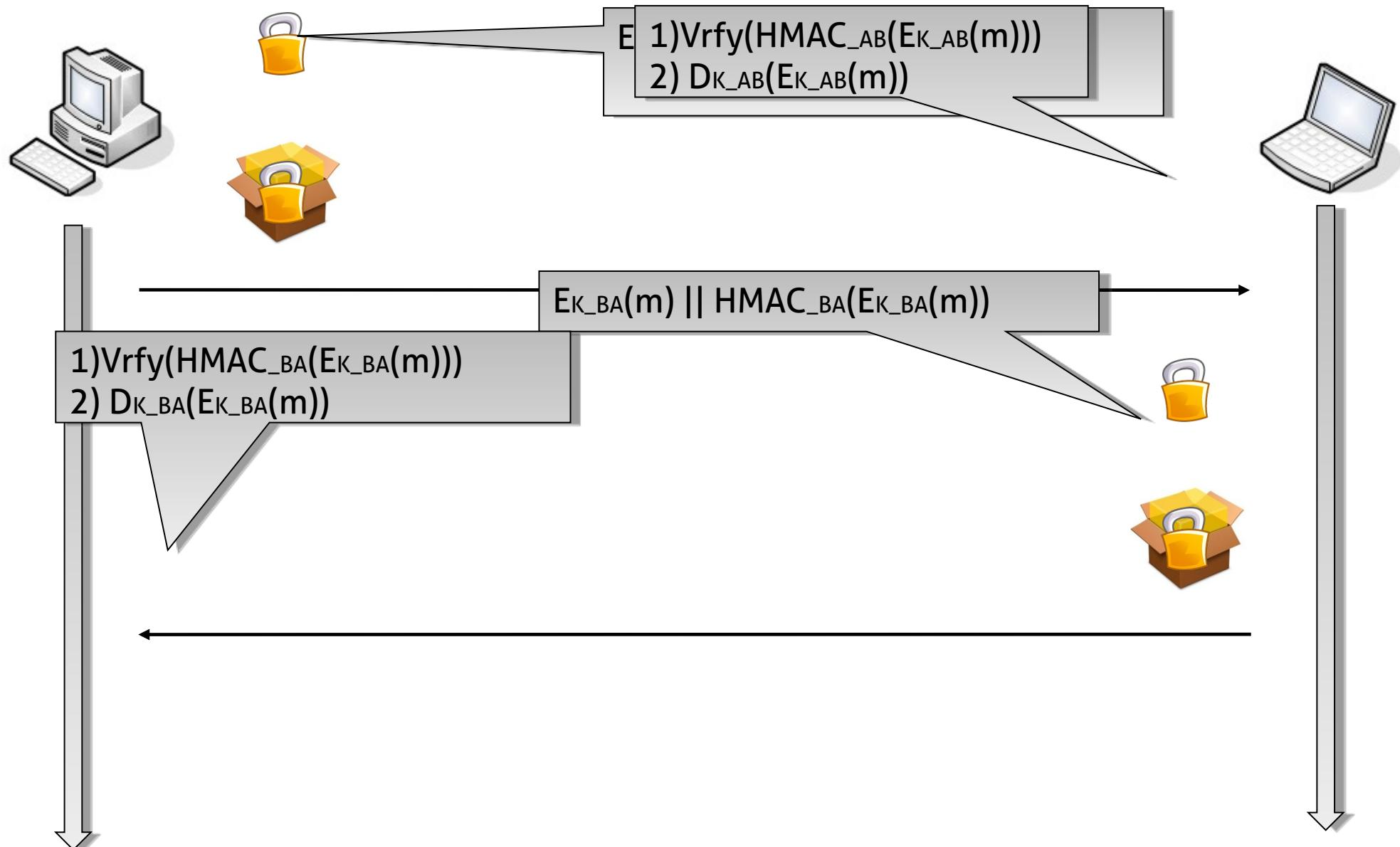
- The endpoints share a key generated with openvpn command
- Very easy to configure
- No CA or certificates
- NOTE: requires a secure channel to exchange the keys
- The key never changes: no forward secrecy



OpenVPN static key: keys

- Uses 4 independent keys:
 - K_{AB} (to encrypt $A \rightarrow B$)
 - $HMAC_{AB}$ (to authenticate $A \rightarrow B$)
 - K_{BA} (to encrypt $B \rightarrow A$)
 - $HMAC_{BA}$ (to authenticate $B \rightarrow A$)
- This is required to reduce the risks of Replay and DoS attacks

OpenVPN static key: traffic exchange





Standard crypto

Blowfish

Block cipher: **64-bit block**

Key length: **32 bits to 448 bits**

Designed by **Bruce Schneier**

Much **faster** than DES and IDEA

Unpatented and royalty-free

No license required

Free **source code available**

- OpenVPN standard
- 128 bit keys
- CBC (Cipher-block Chaining)
- You can choose the default with the cipher option in the configuration file
- Many others available
 - `openvpn --show-ciphers`

SHA-1

Published: **1995**

Designed by: **NSA**

Output length: **160 bit**

- OpenVPN standard
- Uses a different key than the encryption one



OpenVPN static key: key generation

- Generate the shared key on one side of the tunnel (say r1)
 - `openvpn --genkey --secret secret.key`
- Exchange the `secret.key` file with scp
- OR, if you don't send it with scp:
 - Encrypt the key (because we'll use an insecure channel)
 - `openssl enc -aes-128-cbc -e -a -in secret.key -out secret.key.enc`
 - Exchange the shared key
 - Prepare to receive the shared key on the other side of the tunnel (say r2):
 - `r2# nc -l -p 9000 > secret.key.enc`
 - Send the shared key
 - `r1# nc <r2IPaddress> -p 9000 < secret.key.enc`
 - Decrypt the key on the other side of the channel
 - `openssl enc -aes-128-cbc -d -a -in secret.key.enc -out secret.key`



OpenVPN static key: file conf

r1

```
port 1194
proto udp
dev tun
secret secret.key
cipher AES-256-CBC
ifconfig 10.10.10.1 10.10.10.2
```

r2

```
remote <r1IpAddress>
port 1194
proto udp
dev tun
secret secret.key
cipher AES-256-CBC
ifconfig 10.10.10.2 10.10.10.1
```

- r1 plays the role of the passive actor → waits for connections
- Create a new file r1.conf/r2.conf and use the above conf
 - You can check the path `/usr/share/doc/openvpn/examples/`



OpenVPN static key: file conf

- Start openvpn
 - `r1# openvpn --config r1.conf`
 - `r2# openvpn --config r2.conf`
- Check the connectivity on the new interfaces and analyze the traffic with wireshark
- To give visibility of another subnet (i.e., the `lanA` network), you can use the `route` option
 - `route <network> <netmask> <host>`
 - Network and netmask in dotted decimal
 - Host is the next-hop for reaching the network



OpenVPN dynamic key

- Uses SSL/TLS and certificates for authentication and key exchange
- Certificates for both endpoints
- If the certificates are valid
 - HMAC and encryption keys are dynamically generated with OpenSSL
 - This assures Forward Secrecy
- Both parties contribute to key generation





OpenVPN dynamic key: cert generation

- We should have a certification authority issuing the certs
 - This should be done using the → **easy-rsa scripts**, that can be found in the `/usr/share/easy-rsa` directory
 - Alternatively, we can use the ones provided by openvpn for test purposes in `/usr/share/doc/openvpn/examples/sample-keys`
- Needed ingredients/files
 - `{client,server}.crt` : CA signed public key
 - `{client,server}.key`: CA signed private key
 - Certificates are issued after signing the requests (**client!=server**)
 - `dh.pem`: Diffie-Hellman key exchange parameters
 - `ta.key`: for TLS HMAC authentication (optional)



OpenVPN dynamic key: file conf

- Start from the sample configuration files in the directory

`/usr/share/doc/openvpn/examples/sample-config-files/`

- Server (VPN gateway):

`- .../sample-config-files/server.conf`

- client:

`- .../sample-config-files/client.conf`



OpenVPN dynamic key: run

- Start openvpn
- Server:
 - **openvpn --config server.conf**
- Client:
 - **openvpn --config client.conf**
- Check the connectivity on the new interfaces and analyze the traffic with wireshark



That's all for today

- **Questions?**
- Resources:
 - <https://openvpn.net/community-resources/how-to/>
 - <https://wiki.wireshark.org/OpenVPN>
 - Chapter 24 textbook
 - Virtual private networking, Gilbert Held, Wiley ed.
 - Guide to IPsec VPNs, NIST800-77
 - Guide to SSL VPNs, NIST-SP800-113
 - http://www.tcpipguide.com/free/t_IPSecurityIPSecProtocols.htm

Practical Network Defense

Master's degree in Cybersecurity 2021-22

Proxies

Angelo Spognardi
spognardi@di.uniroma1.it

*Dipartimento di Informatica
Sapienza Università di Roma*



Proxies!

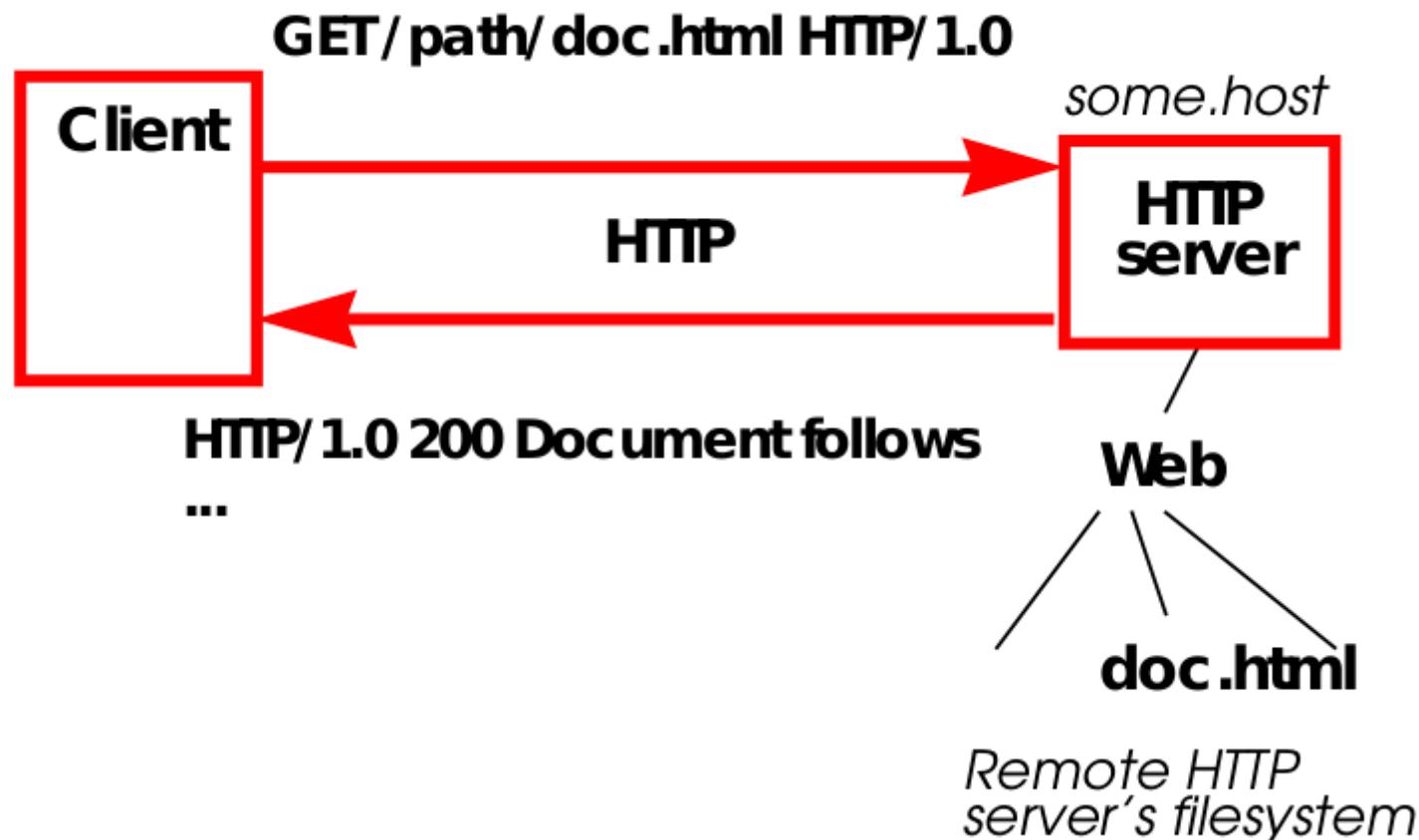
- Forward proxy
- Reverse proxy
- Application proxy
- Transparent proxy



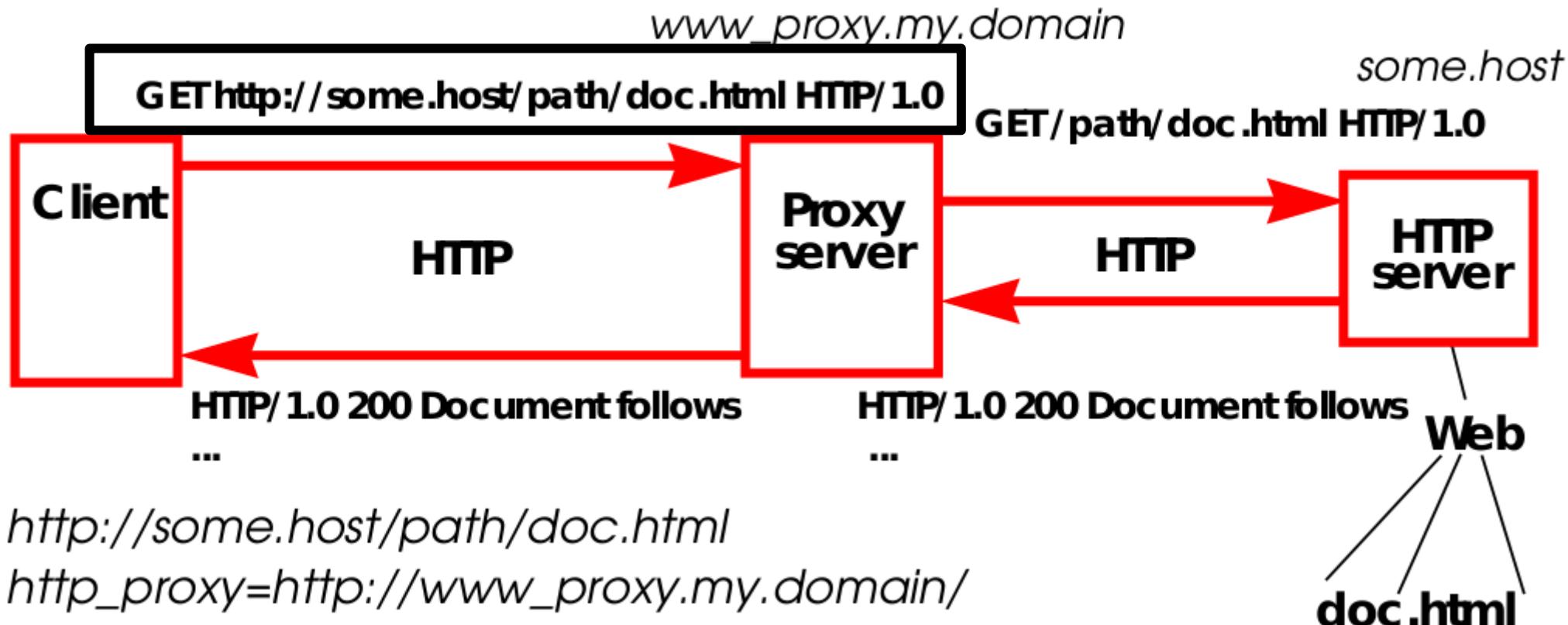
Proxy history: forward proxy

- A WWW proxy server provides access to the Web for people on closed subnets who can only access the Internet through a firewall machine
 - Ari Luotonen, CERN Kevin Altis, Intel, April 1994
- Original idea: *An application-level proxy makes a firewall safely permeable for users in an organization, without creating a potential security hole through which “bad guys” can get into the organizations’ net.*
 - Namely: one single host handling requests from several users.

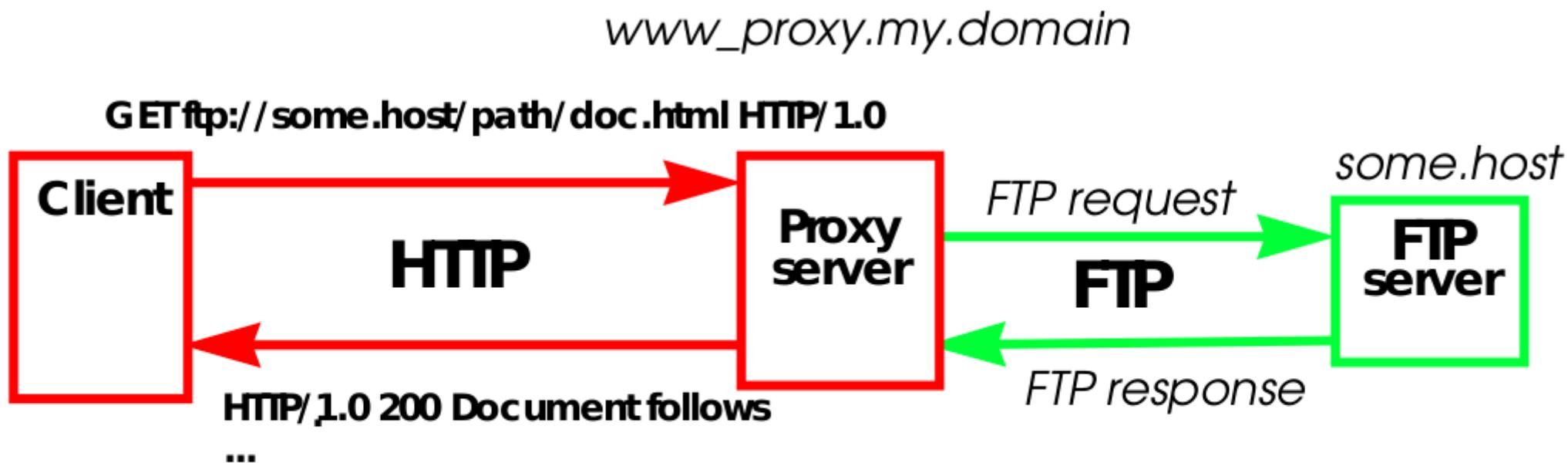
Normal HTTP transaction



Proxied HTTP transaction



Proxied FTP transaction



`ftp://some.host/path/doc.html`

`ftp_proxy=http://www_proxy.my.domain/`



Other benefits of forward proxy

- Authentication, Authorization, Auditing, whitelisting, blacklisting...
- Caching
 - store the retrieved document into a local file for further use so it won't be necessary to connect to the remote server the next time that document is requested
 - Problems:
 - How long is it possible to keep a document in the cache and still be sure that it is up-to-date?
 - How to decide which documents are worth caching and for how long?
 - Solutions:
 - HEAD http request (very inefficient)
 - `If-Modified-Since` request header



Forward proxy

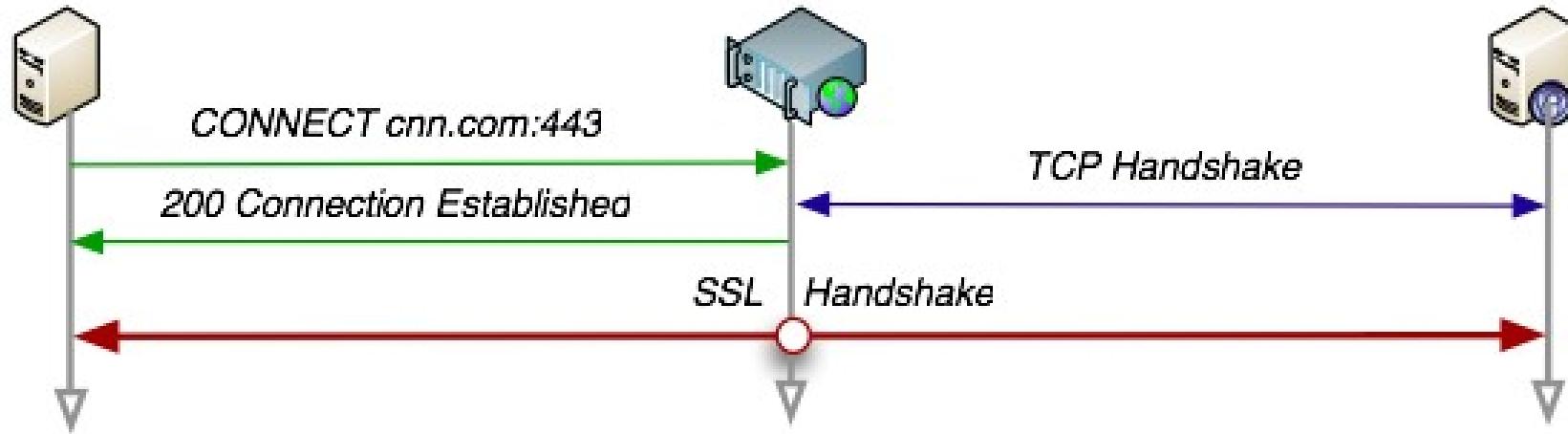
- HTTP requests
 - Standard request in absolute-form to the proxy
 - The proxy will be the middle-point, forwarding the request towards the final termination
- Other (non-HTTP requests)
 - HTTP tunneling
 - HTTP CONNECT request with absolute-form to the proxy
 - The proxy establishes the TCP connection and becomes the middle-point



HTTP tunneling: HTTP CONNECT

- Allow the use of any protocol that uses TCP
 - <https://tools.ietf.org/html/draft-luotonen-web-proxy-tunneling-01>
- Idea: the proxy simply receives the destination host the client wants to connect to and establishes the connection on its behalf
- Then, when the connection is established, the proxy server continues to proxy the TCP stream **unmodified** to and from the client
- Clearly, the proxy can perform authentication, whitelisting, and so on before accepting to forward the stream of data

HTTP CONNECT method



- <https://tools.ietf.org/html/rfc7231#section-4.3.6>
- Anything that uses a two-way TCP connection can be passed through a CONNECT tunnel
 - Example: HTTP forwarding SSL/TLS
- Not all proxy servers support the CONNECT method or limit it to port 443 only



Content-filtering proxy

- After user authentication, HTTP proxy controls over the content that may be relayed
 - In schools, no Facebook or porn websites
 - Blacklists or semantic searches
 - Virus, malware scan
 - No files with executable or watermarking and so on



Anonymizer proxy

- A proxy server that acts as an intermediary and privacy shield between a client computer and the rest of the Internet
- It accesses the Internet on the user's behalf, protecting personal information by hiding the client computer identifying information (IP address, firstly)
- The server sees requests coming from the proxy address rather than the actual client IP address
 - Typical use for accessing restricted content (like *The pirate bay* and similar)
 - <http://spys.one/en/>
 - <https://free-proxy-list.net/>

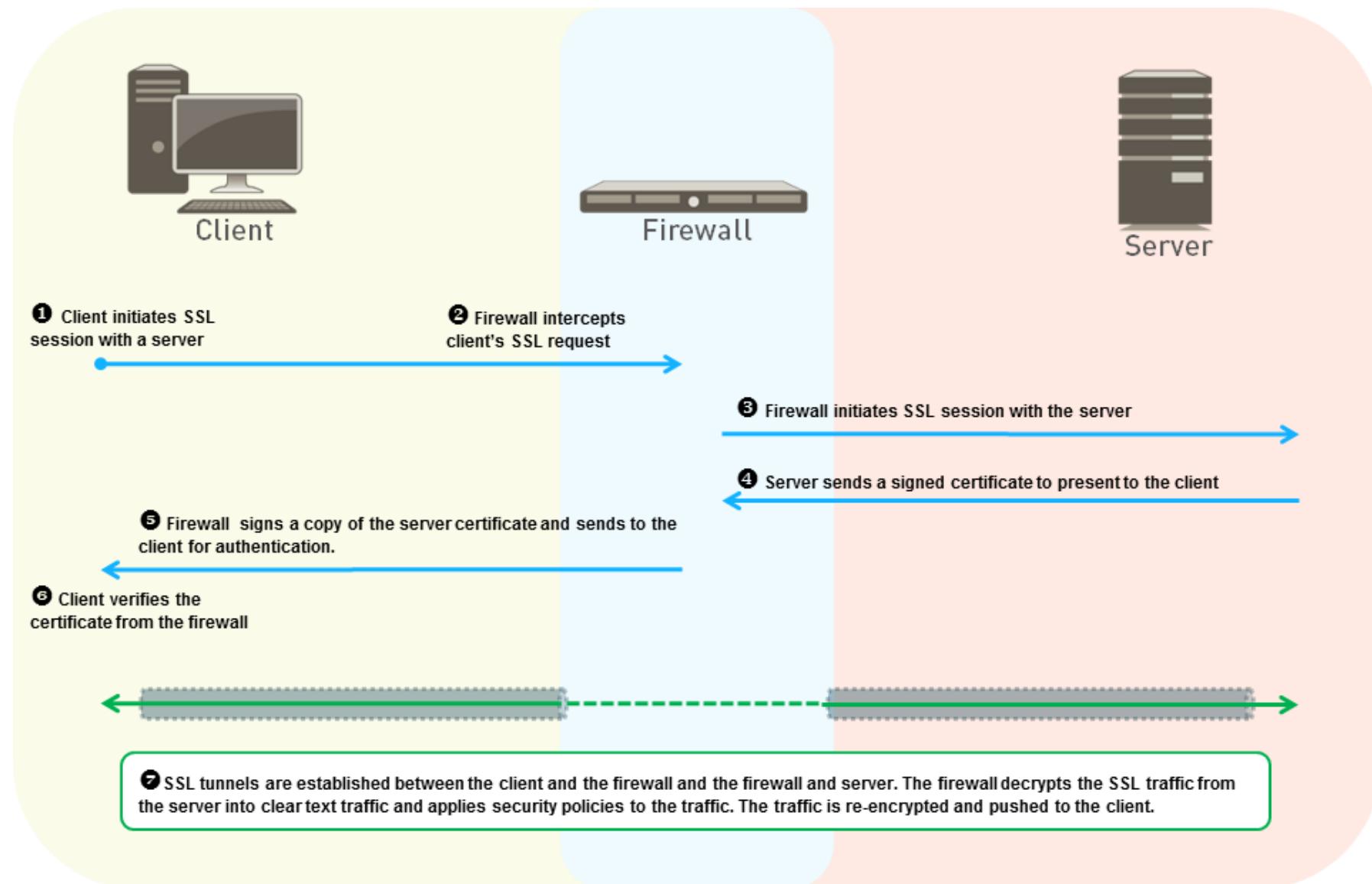


SSL Forward proxy

- A way to decrypt and inspect SSL/TLS traffic from internal users to the web, generally implemented in firewalls
- SSL Forward Proxy decryption prevents malware concealed as SSL encrypted traffic from being introduced in the network
- How it works:
 - The proxy uses certificates to establish itself as a trusted third party to the session between the client and the server
 - As the proxy continues to receive SSL traffic from the server that is destined for the client, it decrypts the SSL traffic into clear text traffic and applies decryption and security profiles to the traffic
 - The proxy, then, re-encrypts and forwards the traffic to the client
 - What about trust?



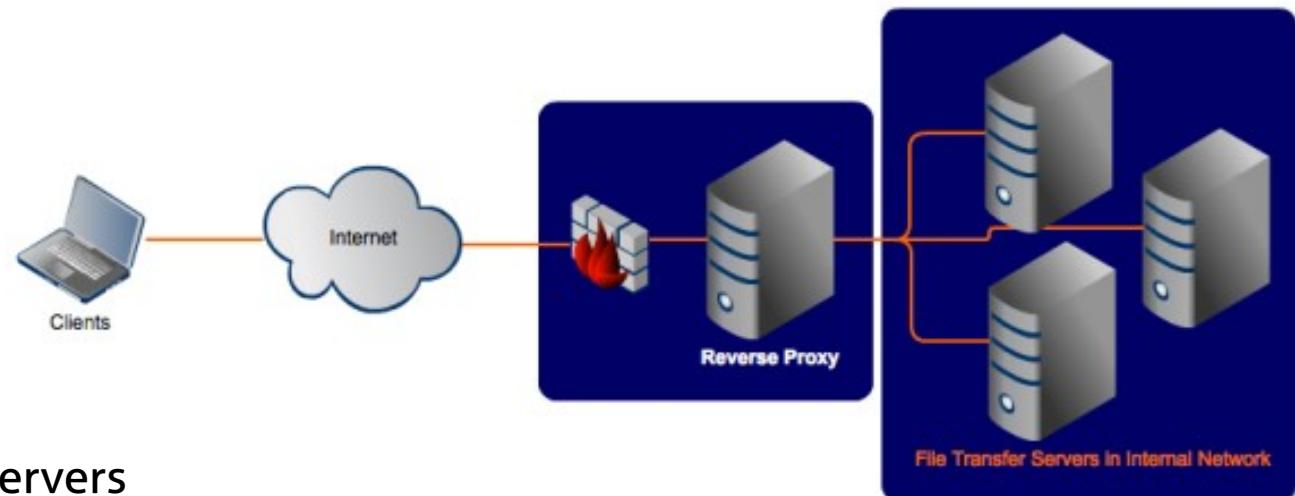
SSL Forward proxy with a figure



<https://docs.paloaltonetworks.com/pan-os/9-1/pan-os-admin/decryption/decryption-concepts/ssl-forward-proxy>

Reverse proxy

- Forward proxy operates on behalf of the client
- Reverse proxy operates on behalf of the server
- It receives the requests from the outside as if it were the server and then forwards the request to the actual destination (origin) server
- Typical functions:
 - Load balancing
 - Cache static content
 - Compression
 - Accessing several servers into the same URL space
 - Securing of the internal servers
 - Application level controls
 - TLS acceleration



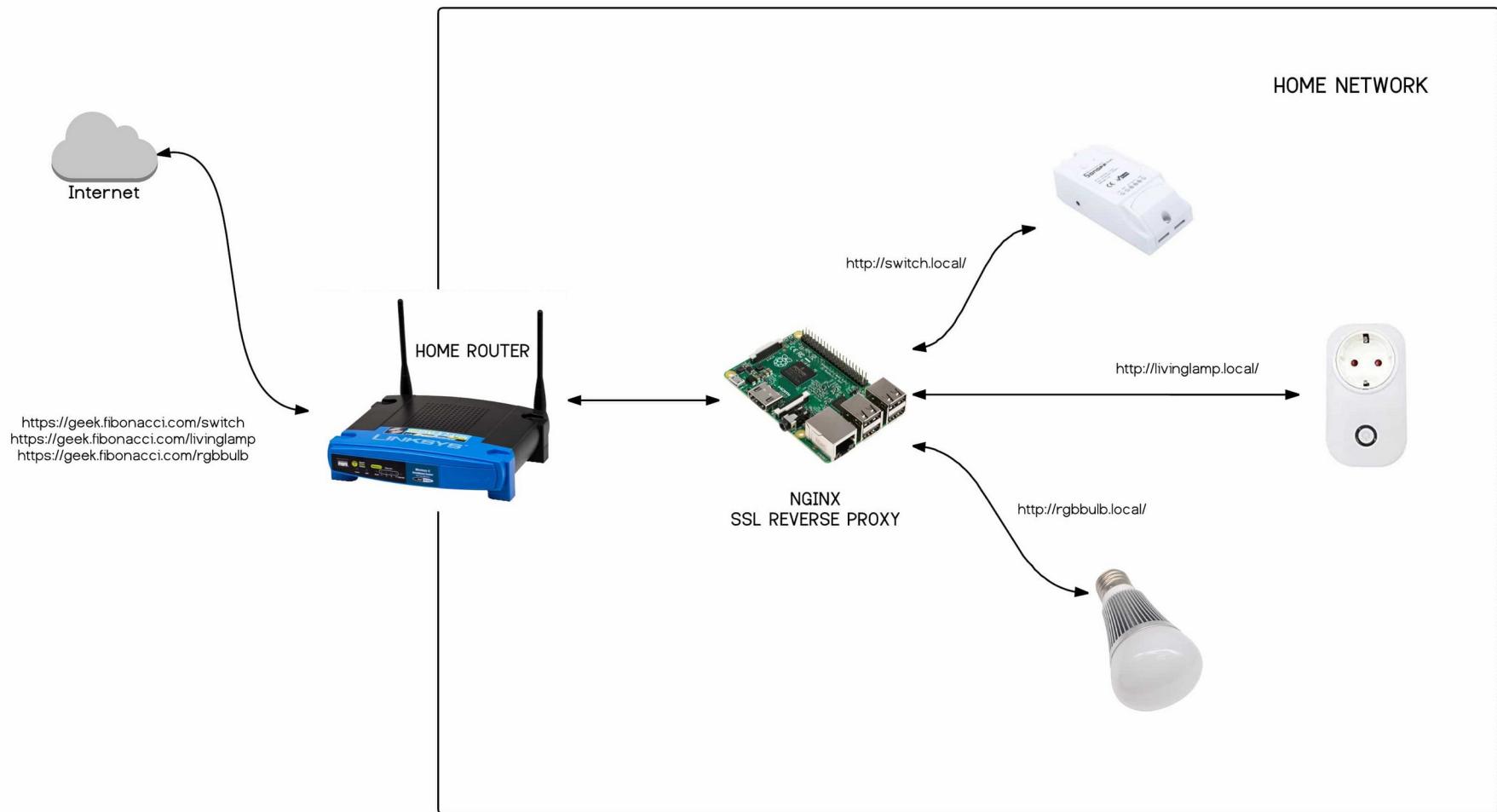


Internal server protection

- The reverse proxy receives the requests from the clients and then issues new, prim and proper requests to the real server
- No direct connection with the outside also means defense against DoS
- Can provide support for HTTPS to servers that only have HTTP
- Can add AAA to services that do not have them
 - Example: a IoT device behind a firewall that must be accessible from the Internet



Reverse proxy for IoT access



<https://tinkerman.cat/post/secure-remote-access-to-your-iot-devices>
However: don't forget client authentication!!



Reverse proxy for application control: application firewall

- Application layer firewall operates at the application layer of a protocol stack
 - A WAF (*Web Application Firewall*) inspects the HTTP traffic and prevents attacks, such as SQL injection, cross-site scripting (XSS), file inclusion, and other types of security issues
 - Example: ModSecurity for apache webserver
- It can block application input/output from detected intrusions or malformed communication, or block contents that violate policies
- It can detect whether an unwanted protocol is being provided through on a non-standard port or whether a protocol is being abused in any harmful way



TLS acceleration

- The SSL/TLS "handshake" process uses digital certificates based on asymmetric or public key encryption technology
- Public key encryption is very secure, but also very processor-intensive and thus has a significant negative impact on performance
 - SSL bottlenecks
- Possible solutions:
 - SSL acceleration: use hardware support to perform modular operations with large operands
 - SSL offloading: use a dedicated server only for SSL handshake



SSL offloading

- SSL Termination
 - The proxy decrypts the TLS/SSL-encrypted data and then sends it on to the server in an unencrypted state
 - This also allows IDS or application firewall inspection
- SSL Forwarding (or Bridging or Initiation)
 - The proxy intercepts and decrypts TLS/SSL-encrypted traffic, examines the contents to ensure that it doesn't contain malicious code, then re-encrypts it before sending it on to the server
 - This only allows inspection of TLS/SSL-encrypted data before it reaches the server to prevent application layer attacks hidden inside



Proxy and HTTPS

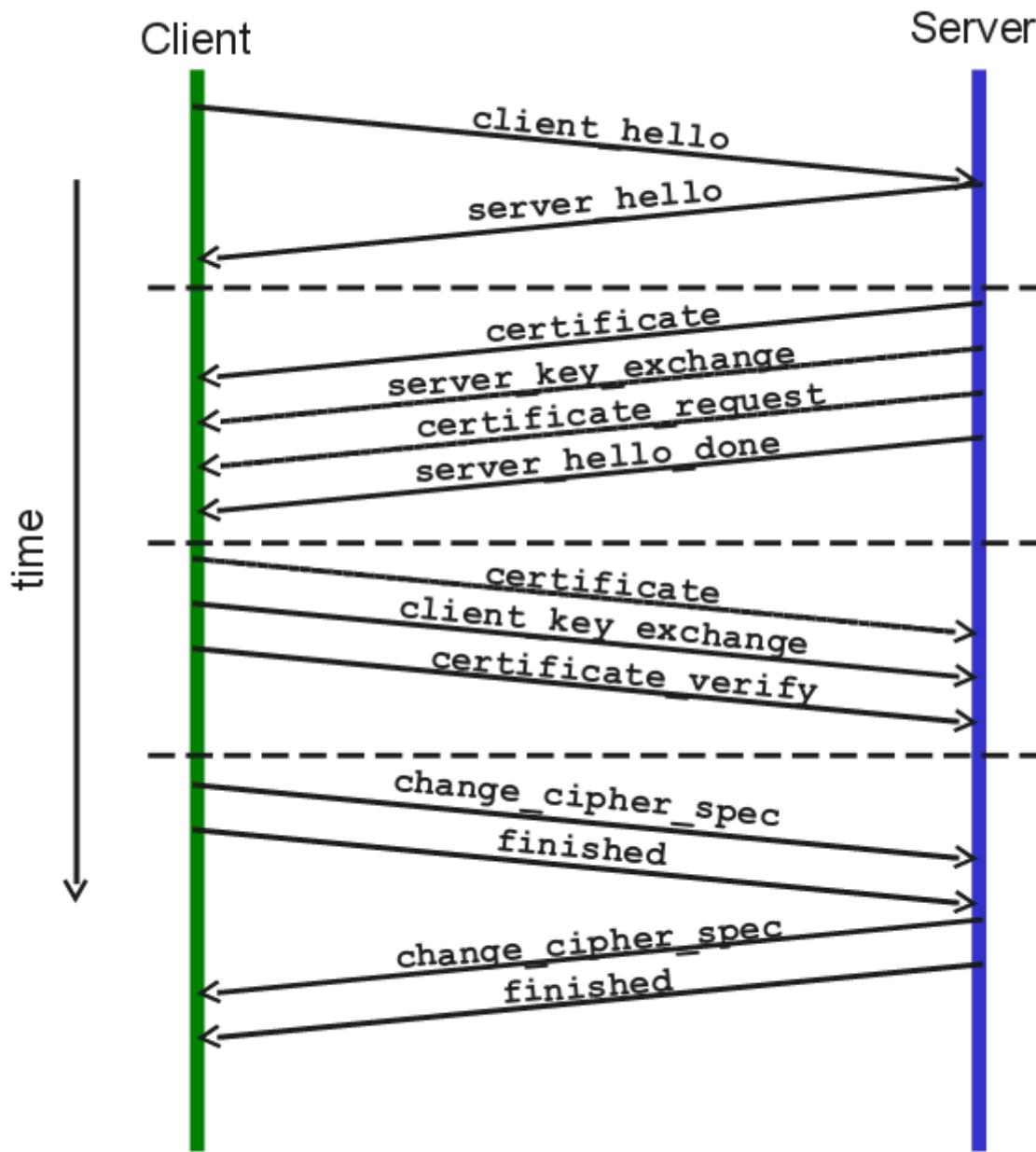
- HTTPS traffic (fortunately) cannot be read
- What if we WANT to read a client HTTPS traffic?
- We need to perform a Man-in-the-middle attack
 - PRETEND to be the real server and be the termination of the SSL/TLS connection
- A possible solution is the **SSL bump**
- It consists in using the requested host name to dynamically generate a server certificate and then impersonate the named server
- But: what if we don't know the host name?
 - Remember that we start the TLS handshake using an IP address and the hostname is sent in the HTTP request...



HTTPS certificate dilemma

- Virtual host: the same webserver can host multiple websites
 - According to the hostname field in the HTTP header, the server understands the requested website
- If HTTPS is used, the SSL/TLS connection requires a certificate to be sent by the server, but...
- Which certificate has to be sent?
 - A certificate valid for all the websites is out of discussion
 - The hostname is within the HTTP traffic but it is **encrypted**
- Then?

TLS Handshake → BEFORE ANY HTTP packet



1) Hello phase

2) Server authentication

3) Client authentication

4) Finish

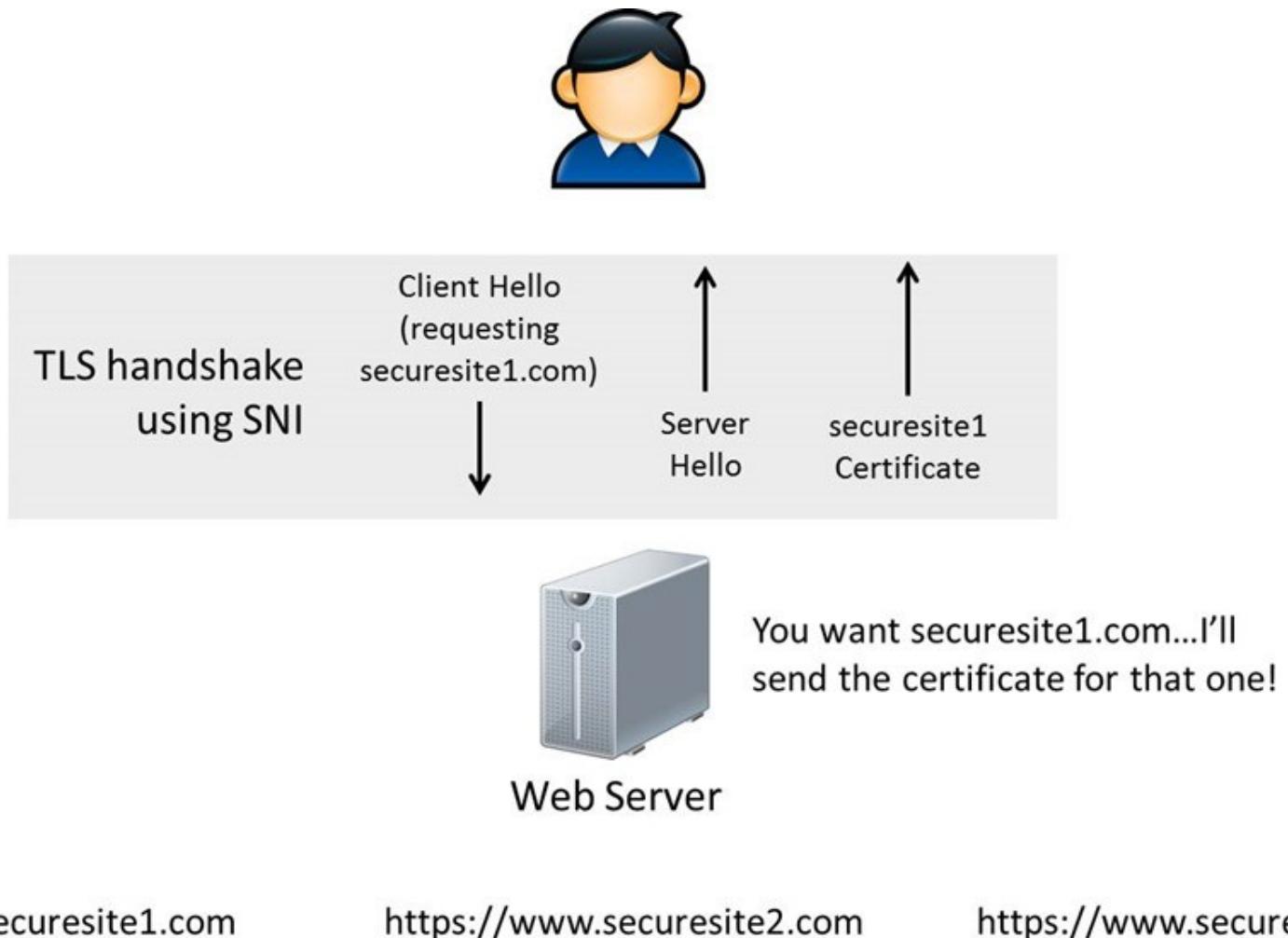


Server Name Indication

- Server Name Indication (SNI) is an extension to TLS by which a client indicates which hostname it is attempting to connect to at **the start of the handshaking process**
 - It is in clear text
- TLS Extensions RFC:
 - <https://tools.ietf.org/html/rfc3546#section-3.1>
- This allows a server to present **multiple certificates** on the same IP address and TCP port number and hence allows multiple secure (HTTPS) websites (or any other service over TLS) to be served by the same IP address **without requiring** all those sites to use the same certificate



SNI example





SNI capture

The screenshot shows a Wireshark capture of an SSL/TLS handshake. The packet list pane shows several TLSv1.2 frames. The selected packet is the Client Hello (packet 134), which includes a server_name extension. This extension is highlighted with a red box in the details pane, showing the value "vesta.web.telegram.org".

No.	Time	Source	Destination	Protocol	Length	Info
68	5.240903...	149.154.167.99	151.100.179.61	TLSv1.2	595	[TCP ACKed unseen segment] , Application Data
78	5.294274...	151.100.179.61	149.154.167.99	TLSv1.2	556	Application Data
134	6.854975...	151.100.179.61	149.154.167.99	TLSv1.2	651	Client Hello
138	6.893017...	149.154.167.99	151.100.179.61	TLSv1.2	5268	Server Hello, Certificate, Server Name
140	6.901700...	151.100.179.61	149.154.167.99	TLSv1.2	192	Client Key Exchange, Change Cipher Spec
141	6.902038...	151.100.179.61	149.154.167.99	TLSv1.2	588	Application Data
142	6.937548...	149.154.167.99	151.100.179.61	TLSv1.2	340	New Session Ticket, Change Cipher Spec
148	7.111207...	149.154.167.99	151.100.179.61	TLSv1.2	595	Application Data
150	7.113453...	149.154.167.99	151.100.179.61	TLSv1.2	595	Application Data
158	7.176662...	151.100.179.61	149.154.167.99	TLSv1.2	668	Application Data
160	7.428280...	69.164.215.152	151.100.179.61	TLSv1.2	100	Application Data

Details pane (highlighted area):

- ▼ Extension: server_name (len=27)
 - Type: server_name (0)
 - Length: 27
- ▼ Server Name Indication extension
 - Server Name list length: 25
 - Server Name Type: host_name (0)
 - Server Name length: 22
 - Server Name: vesta.web.telegram.org
- ▼ Extension: extended_master_secret (len=0)
 - Type: extended_master_secret (23)
 - Length: 0
- ▼ Extension: renegotiation_info (len=1)
 - Type: renegotiation_info (65281)

Hex and ASCII panes are also visible at the bottom of the interface.



SNI implications

- An eavesdropper can see which site is being requested
- This helps security companies provide a filtering feature and governments implement censorship
 - Trick: [Domain Fronting](#)
 - In the SNI use the name of a CDN server and then use a different host in the real HTTP request, always within the same CDN
- An upgrade called [Encrypted SNI \(ESNI\)](#) is being rolled out in an "experimental phase" to address this risk of domain eavesdropping

<https://attack.mitre.org/techniques/T1172/>

<https://www.cloudflare.com/learning/ssl/what-is-encrypted-sni/>



SOCKS proxy

- Circuit level gateway
 - Works at the session layer of the OSI model, or as a "shim-layer" between the application layer and the transport layer of the TCP/IP stack
- Similar to the HTTP CONNECT proxy
 - A bit more versatile:
 - Many authentication mechanisms
 - Can tunnel TCP, but also UDP and IPv6 (SOCKS5)
 - Can also work as a reverse proxy
- Implemented in SSH, putty and Tor



Transparent proxy

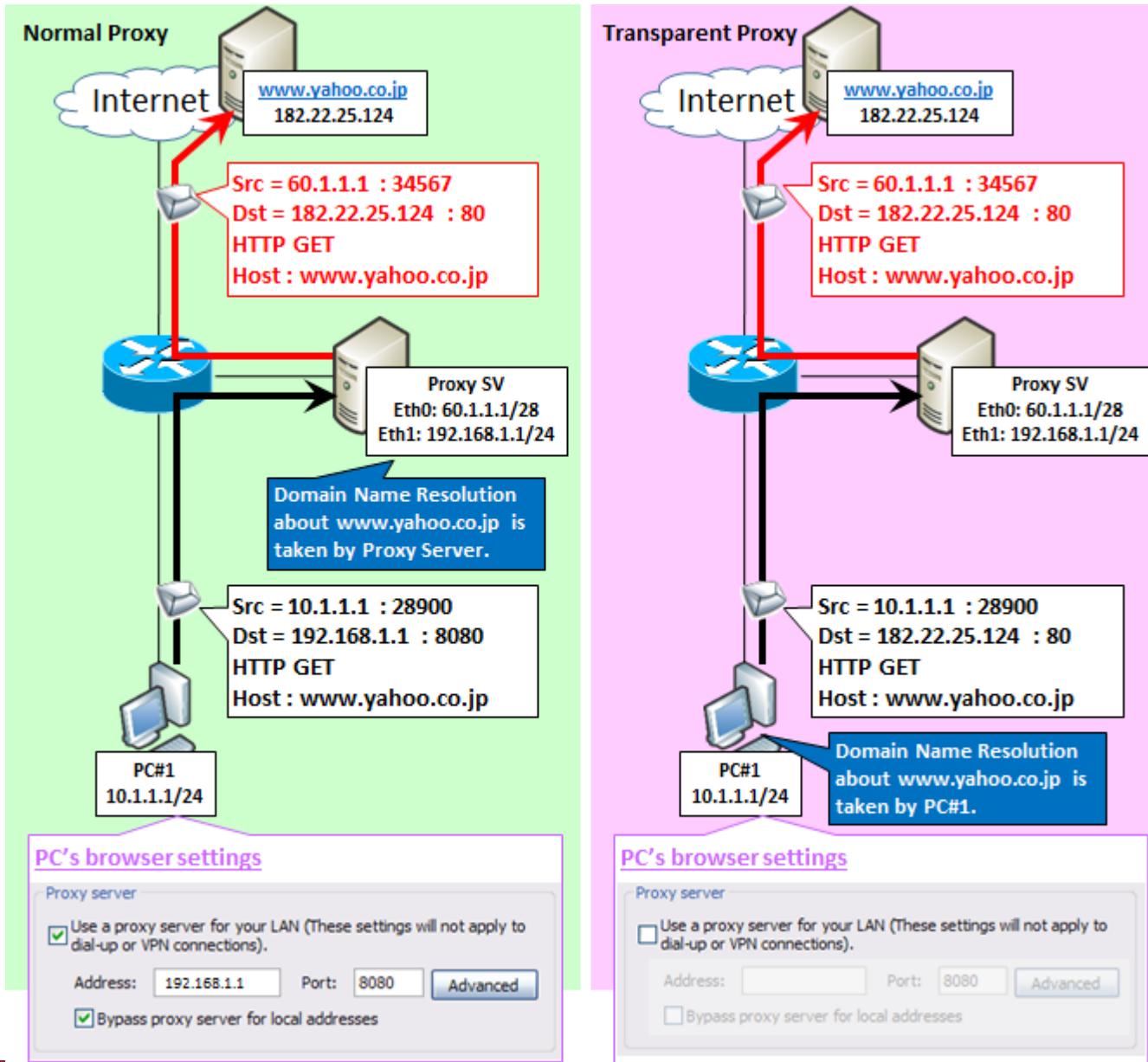
- Forward proxy need proxy-capable client programs and/or user education so that users know how to use the proxies
- A transparent proxy is made for normal user procedures and normal client applications
 - *A transparent application proxy is often described as a system that appears like a packet filter to clients, and like a classical proxy to servers* (RFC1919)
- Alternative names:
 - Intercepting proxy
 - Inline proxy
 - Forced proxy
- Common practice for some ISPs (mainly for mobile users)



How transparent proxy works

- At the start of a session, a TCP packet with a source address of Client and a destination address of Server travels to the proxy system, expecting to cross it just like a normal IP gateway
- The proxy's TCP/IP software stack sees this incoming packets for a destination address that is NOT one of its own addresses
- Forward or drop the packet? → ACCEPT!
 - It PRETENDS to be the Server
 - And then operates like a standard proxy, as a middle-point between Client and Server
- What is the difference with NAT?

Normal proxy vs. Transparent proxy

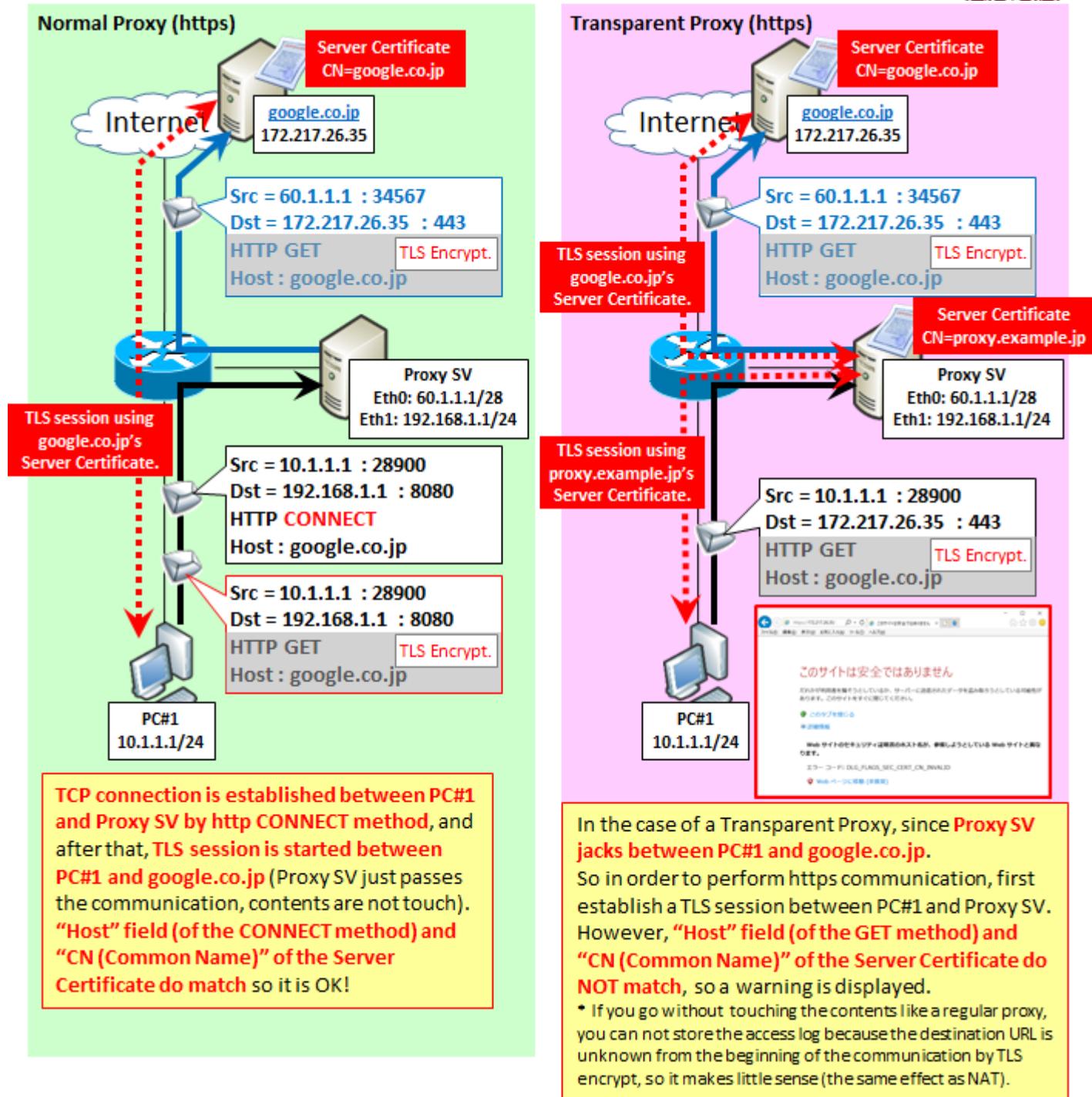




Transparent proxy caveats

- Proxy intercepts the HTTP requests that has the IP address of the Server
 - Again, it does not know the hostname, but has to look for it within the HTTP request
 - What about HTTPS? → See reverse proxy and SSL bump
- How to intercept the HTTP/HTTPS requests?
 - If we do the interception at the default route, we will break other protocols such as SMTP, POP, IMAP
 - Then?
 - Policy-based routing (PBR)

HTTPS in normal vs. Transparent Proxy





Policy based routing

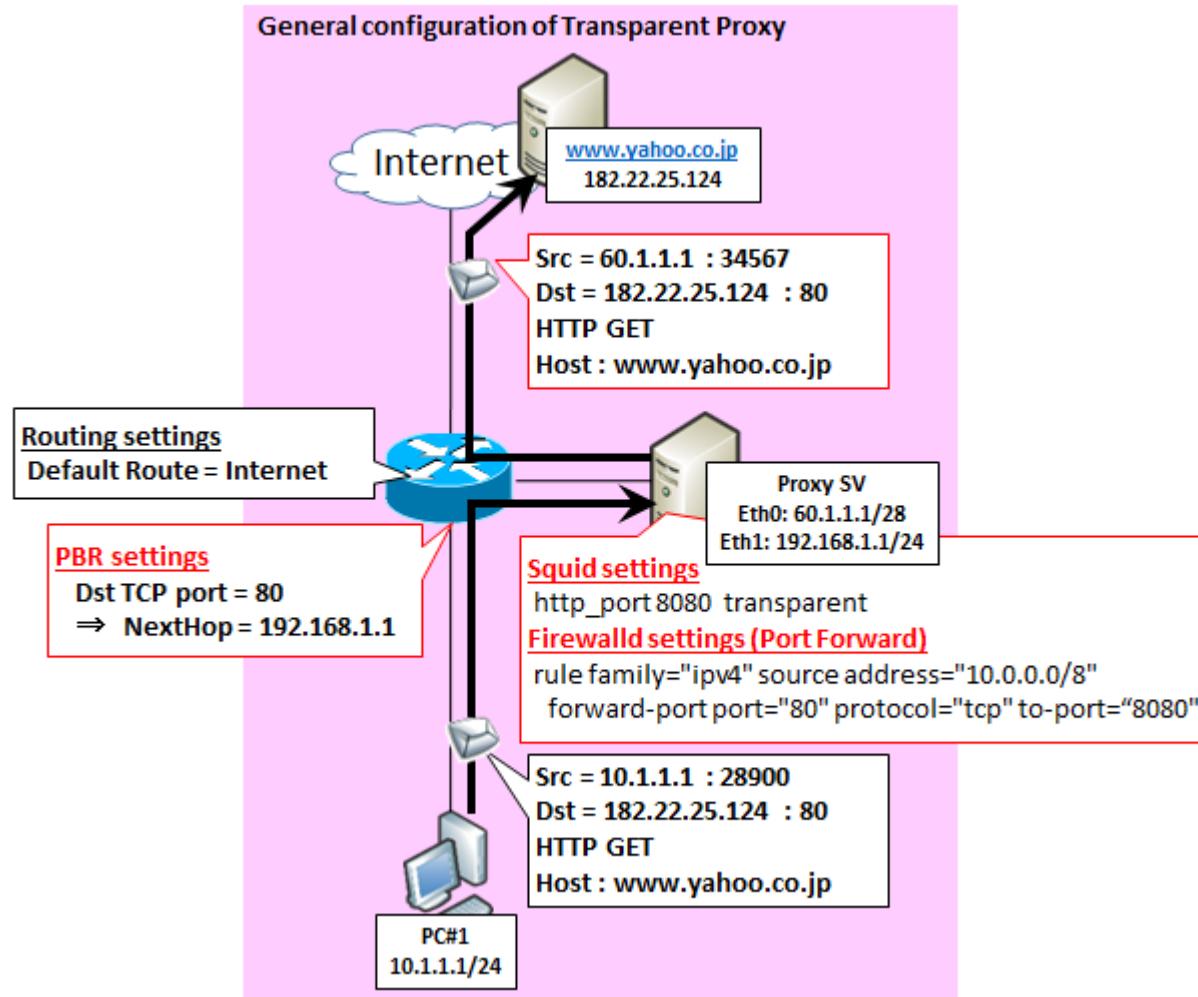
- In traditional routing: "All routing is a destination-driven process"
- What about routing decisions based on other info?
 - Quality of service
 - Source routing
 - Traffic shaping
 - Etc.
- Policy based routing goes beyond simple destination-driven routing



PBR and transparent proxy

- We can use as policy:
 - any TPC connection with destination port 80 MUST be forwarded to the proxy
- Then we need a different routing table, with a different next hop → the proxy
- This can be done via iptables and the “packet marking”

Configuration for a Transparent Proxy



<https://milestone-of-se.nesuke.com/en/sv-advanced/server-software/transparent-proxy-2/>



ICAP: Internet Content Adaptation Protocol

- Defined in RFC-3507, it is a protocol aimed at providing simple object-based content vectoring for HTTP services
- ICAP is, in essence, a lightweight protocol for executing a "remote procedure call" on HTTP messages
- It allows ICAP clients to pass HTTP messages to ICAP servers for some sort of transformation or other processing ("adaptation")
- The ICAP server executes its transformation service on messages and sends back responses to the client, usually with modified messages
 - Typically, the adapted messages are either HTTP requests or HTTP responses



ICAP examples

- Coupled with transparent proxy (acting as a ICAP client), it allows to transparently provide additional functionalities, using a standardized interface towards ICAP servers
- Examples:
 - Simple transformations of content can be performed near the edge of the network instead of requiring an updated copy of an object from an origin server
 - Ex: translations to other languages, formatting for different devices, inclusions of different advertisements, and so on
 - Expensive operations on the content can be performed by “surrogates” instead of the origin server
 - Ex: file scan for viruses, check file upload/download, and so on
 - Checking if requested URIs are allowed or not
 - Ex: parental control, content filtering, and so on



That's all for today

- Questions?
 - See you next lecture!
- References:
 - Ari Luotonen, Kevin Altis, [World-Wide Web Proxies](#), 1994
 - http://httpd.apache.org/docs/current/mod/mod_proxy.html
 - https://en.wikipedia.org/wiki/Proxy_server
 - [Classical vs Transparent IP Proxies, RFC 1919](#)
 - SOCKS 5, [RFC 1928](#)
 - HTTP 1.1, [RFC 7230](#)
 - Policy based routing and [Linux advanced routing and traffic control](#)
 - ICAP, [RFC 3507](#)

Practical Network Defense

Master's degree in Cybersecurity 2021-22

IPSec activity

Angelo Spognardi
[*spognardi@di.uniroma1.it*](mailto:spognardi@di.uniroma1.it)

*Dipartimento di Informatica
Sapienza Università di Roma*



Aim of the lab

- 1) Realize a IPsec configuration
- 2) Transport mode
- 3) Tunnel mode

Credits:

Part of the slides are taken from the Network Security (NetSec) course
IN2101 – WS 19/20 of Technical University of Munich:
http://netsec.net.in.tum.de/slides/12_ipsec.pdf



IPsec (RFC 4301)

- A Network Layer protocol suite for providing security over IP.
- Part of IPv6; an add-on for IPv4.
- Can handle all three possible security architectures:

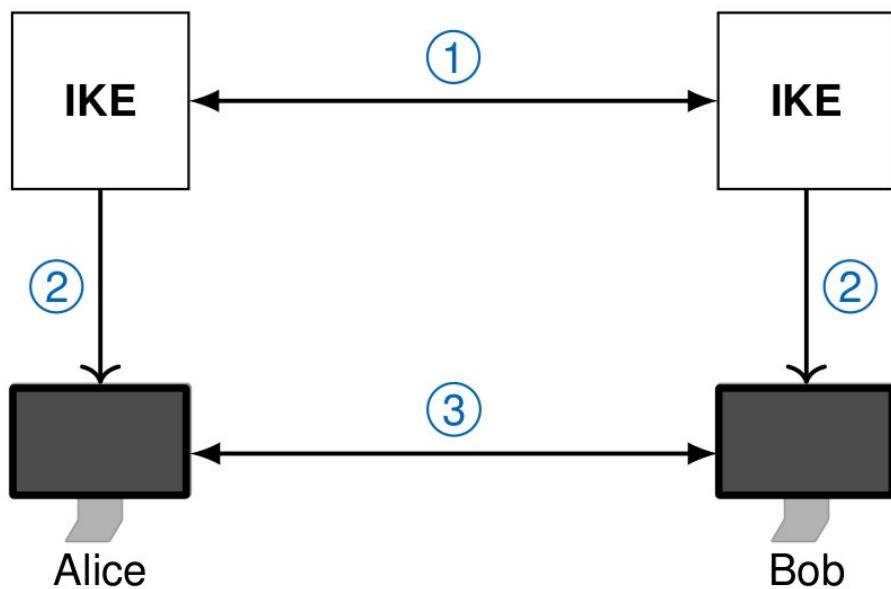
Feature	Gateway-to-Gateway	Host-to-Gateway	Host-to-Host
Protection between client and local gateway	No	N/A (client is VPN endpoint)	N/A (client is VPN endpoint)
Protection between VPN endpoints	Yes	Yes	Yes
Protection between remote gateway and remote server (behind gateway)	No	No	N/A (client is VPN endpoint)
Transparency to users	Yes	No	No
Transparency to users' systems	Yes	No	No
Transparency to servers	Yes	Yes	No



Fundamentals of IPsec

- Data origin authentication
 - It is not possible to spoof source / destination addresses without the receiver being able to detect this
 - It is not possible to replay a recorded IP packet without the receiver being able to detect this
- Connectionless Data Integrity
 - The receiver is able to detect any modification of IP datagrams in transit
- Confidentiality
 - It is not possible to eavesdrop on the content of IP datagrams
 - Limited traffic flow confidentiality
- Security Policies
 - All involved nodes can determine the required protection for a packet
 - Intermediate nodes and the receiver will drop packets not meeting these requirements

IPsec overview



1) Authentication, key exchange and negotiation of crypto algorithms

- Manual
- Automated: ISAKMP, Internet Key Exchange (IKE), IKEv2

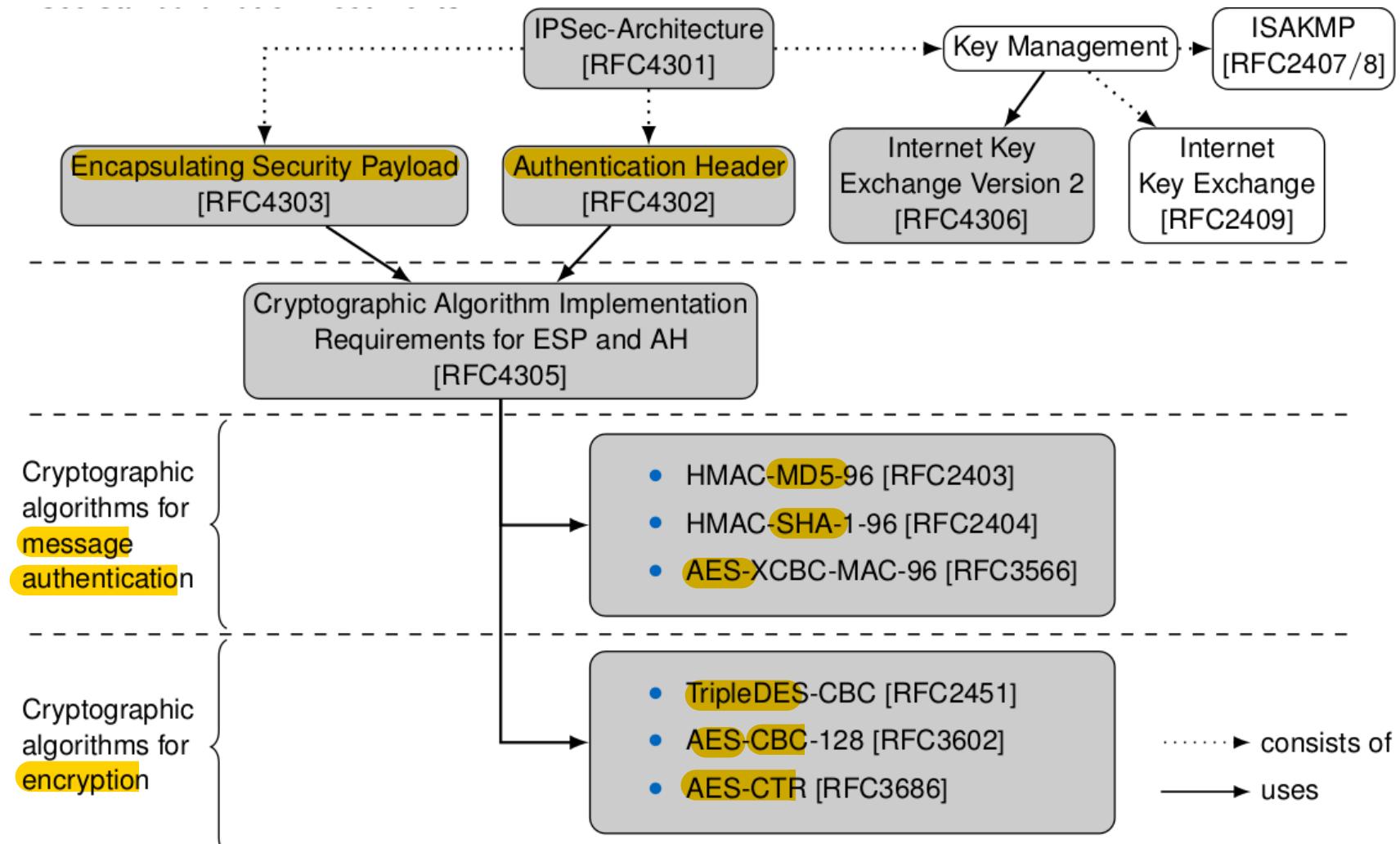
2) Set up of key and crypto-algorithms

3) Use of the secure channel, with:

- Data Integrity via Authentication Header (AH) or Encapsulating Security Payload (ESP)
- Confidentiality using ESP
 - ESP can provide both data integrity and encryption while AH only provides data integrity



IPSec Standardization Documents



List of IPSec related RFCs: <https://datatracker.ietf.org/wg/ipsec/documents/>



IPSec architecture (RFC 4301)

- Concepts
 - Security Association (SA) and Security Association Database (SAD)
 - Security Policy (SP) and Security Policy Database (SPD)
- Fundamental Protocols
 - Authentication Header (AH)
 - Encapsulation Security Payload (ESP)
- Protocol Modes
 - Transport Mode
 - Tunnel Mode
- Key Management Protocols
 - ISAKMP, IKE, IKEv2



IPsec services

- Basic functions, provided by separate (sub-)protocols:
 - Authentication Header (AH): Support for data integrity and authentication of IP packets.
 - Encapsulated Security Payload (ESP): Support for encryption and (optionally) authentication.
 - Internet Key Exchange (IKE): Support for key management etc.

Service	AH	ESP (encrypt only)	ESP(encrypt+authent.)
Access Control	+	+	+
Connectionless integrity	+		+
Protection between VPN endpoints	+		+
Data origin authentication	+		+
Reject replayed packets		+	+
Payload confidentiality		+	+
Metadata confidentiality		partial	partial
Traffic flow confidentiality		(*)	(*)



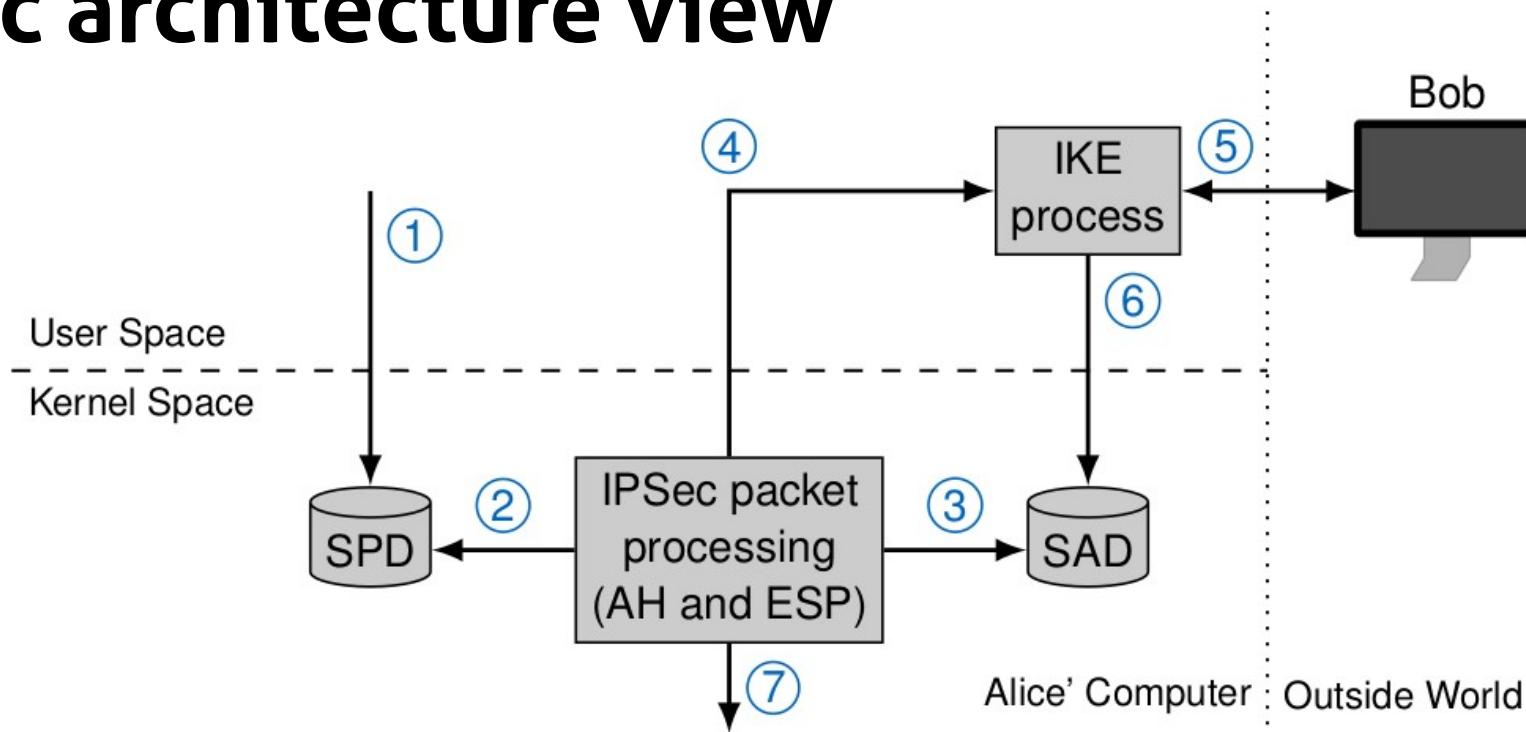
IPsec modes

- **Transport Mode**
 - Provides protection for a T-layer packet embedded as payload in an IP packet.

- **Tunnel Mode**
 - Provides protection for an IP packet embedded as payload in an IP packet.

	Transport Mode SA	Tunnel Mode SA
AH	Authenticate IP payload and selected parts of IP header and IPv6 extension headers.	Authenticate entire inner IP packet and selected parts of outer IP header and outer IPv6 extension headers.
ESP	Encrypt IP payload + any IPv6 extension headers after ESP header.	Encrypt inner IP packet.
ESP + authent.	Encrypt IP payload + any IPv6 extension headers after ESP header. Authenticate IP payload.	Encrypt and authenticate inner IP packet.

IPSec architecture view



1. The administrator sets a policy in SPD
2. The IPSec processing module refers to the SPD to decide on applying IPSec on packet
3. If IPSec is required, then the IPSec module looks for the IPSec SA in the SAD
4. If there is no SA yet, the IPSec module sends a request to the IKE process to create an SA
5. The IKE process negotiates keys and crypto algorithms with the peer host using the IKE/IKEv2 protocol
6. The IKE process writes the key and all required parameters into the SAD
7. The IPSec module can now send a packet with applied IPSec



Security Policies

- A Security Policy (SP) specifies which security services should be provided to IP packets and their details
 - A SP affects only a specific IP stream
 - For each stream, it includes several security attributes:
 - The security protocol (AH / ESP)
 - The protocol mode (Transport / Tunnel)
 - Other parameters, like policy lifetime, port number for specific applications, etc.
 - The actions (e.g. Discard, Secure, Bypass)
- Security Policies are stored in the Security Policy Database (SPD)



Security Policy example

```
root@r2:/# /usr/sbin/setkey -PD
100.90.0.100[any] 100.60.0.100[any] 255
    out prio def ipsec
    esp/transport//require
    created: May 10 13:42:22 2022  lastused: May 10 13:43:08 2022
    lifetime: 0(s) validtime: 0(s)
    spid=441 seq=1 pid=546
    refcnt=1
100.60.0.100[any] 100.90.0.100[any] 255
    fwd prio def ipsec
    esp/transport//require
    created: May 10 13:42:22 2022  lastused:
    lifetime: 0(s) validtime: 0(s)
    spid=434 seq=2 pid=546
    refcnt=1
100.60.0.100[any] 100.90.0.100[any] 255
    in prio def ipsec
    esp/transport//require
    created: May 10 13:42:22 2022  lastused: May 10 13:43:08 2022
    lifetime: 0(s) validtime: 0(s)
    spid=424 seq=0 pid=546
    refcnt=1
root@r2:/# █
```



Security Policy actions

- **Discard**: reject to send/receive the packet
- **Bypass** (none): do not handle with IPSec → send in clear
- **Secure** (ipsec): handle with IPSec
 - In this case, the security policy is in the form **protocol/mode/src-dst/level** where:
 - Protocol → **ah**, **esp** or **ipcomp** (for payload compression)
 - Mode → **tunnel** or **transport**
 - Src-dst → endpoints of the tunnel (if needed)
 - Level → **default**, **use**, **require**, or **unique**
 - This specifies the level of the **SA**, when a keying daemon is used



Security Associations

- A Security Association (SA) is a simplex channel that describes the way how packets need to be processed
 - A SA specifies encryption/authentication algorithms and keys
 - A SA is associated with either AH or ESP (not both)
- Bidirectional communication requires two security associations
- SAs can be setup as
 - Host Host
 - Host Gateway
 - Gateway Gateway
- Security Associations are stored in the Security Association Database (SAD)



Security Association Database (SAD)

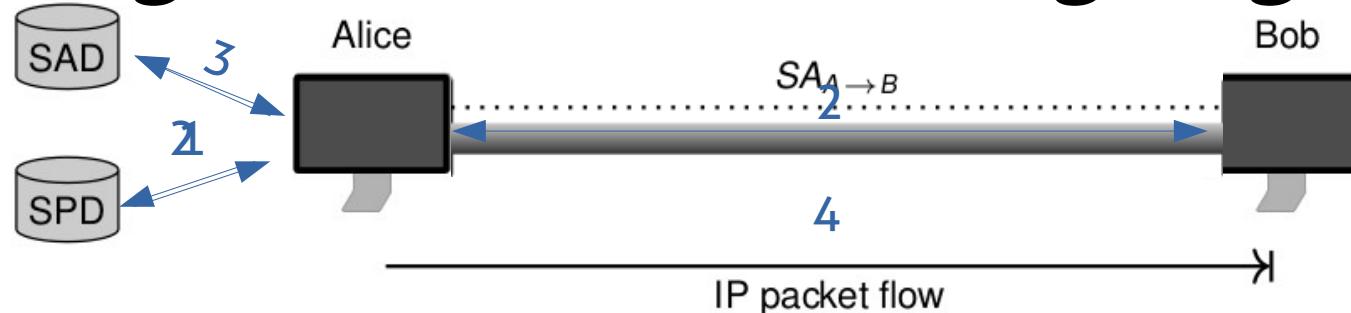
- An entry (SA) is uniquely identified by a Security Parameter Index (SPI)
 - The SPI value for construction of AH/ESP headers is looked up for outbound SAs
 - The SPI value is used to map the traffic to the appropriate SA for inbound traffic
- An SA entry in the SAD includes
 - Security Parameter Index (SPI)
 - Source/destination IP addresses
 - Identified protocol (AH / ESP)
 - IPSec protocol mode (tunnel / transport)
 - Protocol algorithms, modes, IVs and keys
 - Security Association Lifetime
 - Current sequence number counter (replay protection)
 - Other pieces of information (see RFC4301, Section 4.4.2.1)



Security Association example

```
root@r2:/# /usr/sbin/setkey -D
100.60.0.100 100.90.0.100
    esp mode=transport spi=801(0x00000321) reqid=0(0x00000000)
        E: aes-cbc 32eac750 a8ab4acd d922c085 356799ce efe3768e f1979fa5 27b1bcb4 33568e52
        seq=0x00000000 replay=0 flags=0x00000000 state=mature
        created: May 10 13:42:22 2022 current: May 10 13:45:07 2022
        diff: 165(s) hard: 0(s) soft: 0(s)
        last: May 10 13:43:02 2022 hard: 0(s) soft: 0(s)
        current: 448(bytes) hard: 0(bytes) soft: 0(bytes)
        allocated: 7 hard: 0 soft: 0
        sadb_seq=1 pid=544 refcnt=0
100.90.0.100 100.60.0.100
    esp mode=transport spi=800(0x00000320) reqid=0(0x00000000)
        E: aes-cbc 4ac06c9c d4a61a6b ade1dcf3 b5c2731b e0c09cd1 2385bb8e be04ffdb 990eb9e0
        seq=0x00000000 replay=0 flags=0x00000000 state=mature
        created: May 10 13:42:22 2022 current: May 10 13:45:07 2022
        diff: 165(s) hard: 0(s) soft: 0(s)
        last: May 10 13:42:23 2022 hard: 0(s) soft: 0(s)
        current: 2816(bytes) hard: 0(bytes) soft: 0(bytes)
        allocated: 44 hard: 0 soft: 0
        sadb_seq=0 pid=544 refcnt=0
root@r2:/# █
```

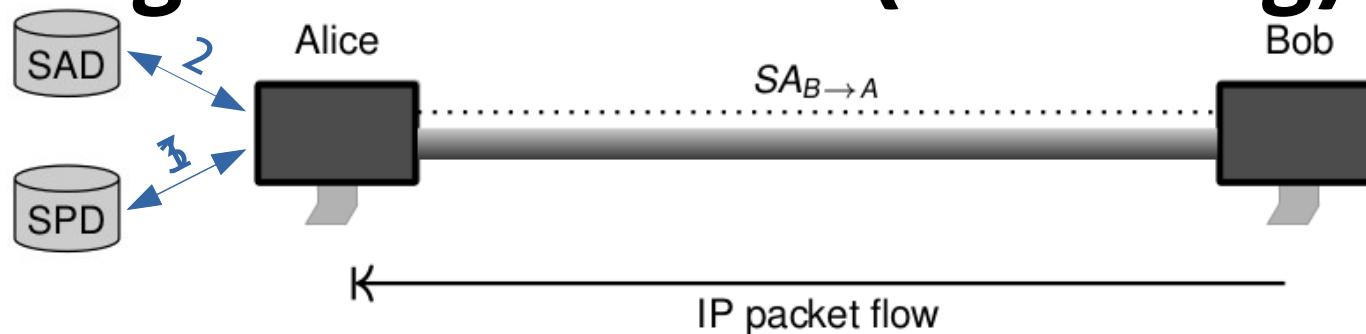
Processing of IPSec Traffic (outgoing)



- Alice wants to send data to Bob, then IP layer of Alice has to:
 - 1) Determine if and how the outgoing packet needs to be secured
 - Perform a lookup in the SPD based on traffic selectors
 - If the policy specifies discard then drop the packet
 - If the policy does not need to be secured, send it
 - 2) Determine which SA should be applied to the packet
 - If no SA is established perform IKE
 - There may be more than one SA matching the packet (e.g. one for AH, one for ESP)
 - 3) Look up the determined or freshly created SA in the SAD
 - 4) Perform the security transforms, specified in the SA
 - This results in the construction of an AH or ESP header
 - Possibly a new (outer) IP header will be created (tunnel mode)
 - 5) Send the resulting packet



Processing of IPSec Traffic (incoming)



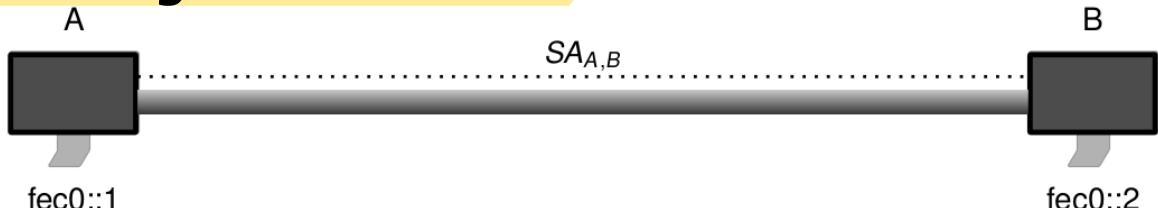
- Alice receives data from Bob, then the IP layer of Alice has to:
 - 1) If packet contains an IPSec header
 - Perform a lookup in the SPD, if Alice is supposed to process the packet
 - Retrieve the respective policy
 - 2) If Alice is supposed to process the packet
 - Extract the SPI from the IPSec header, look up the SA in the SAD and perform the appropriate processing
 - If there's no SA referenced by the SPI ⇒ Drop the packet
 - 3) Determine if and how the packet should have been protected
 - Perform a lookup in the SPD, evaluating the inner IP header in case of tunneled packets
 - If the respective policy specifies discard ⇒ Drop the packet
 - If the protection of the packet did not match the policy ⇒ Drop the packet
 - 4) Deliver to the appropriate protocol entity (e.g. network / transport layer)



ipsec-tools (deprecated)

- The framework ipsec-tools included two main tools:
 - **setkey**, used to manually manipulate the IPsec SA/SP database
 - default config file: `/etc/ipsec-tools.conf`
 - <https://manpages.debian.org/testing/ipsec-tools/setkey.8.en.html>
 - **racoon**, the daemon for the IKE protocol
 - <https://manpages.debian.org/testing/racoon/racoon.8.en.html>
- The ipsec-tools have been deprecated time ago
 - We see them only for educational purposes

Setup of IPsec Security Policies



- Example:
IPv6 connection with ESP and Transport Mode

- Configuration at Host A

source IP

dest. IP

upper-layer
protocol

this policy is for
outgoing packets

ipsec processing rule, as
protocol/mode/src-dst/level

```
spdadd fec0::1 fec0::2 any -P out ipsec esp/transport//require;
```

```
spdadd fec0::2 fec0::1 any -P in ipsec esp/transport//require;
```

- Configuration at Host B

```
spdadd fec0::2 fec0::1 any -P out ipsec esp/transport//require;
```

```
spdadd fec0::1 fec0::2 any -P in ipsec esp/transport//require;
```

this policy is for
incoming packets



Another Security Policy



- Example

IPv6 connection with ESP/Transport applied first and
AH/Transport applied next

- Configuration at Host A

```
spdadd fec0::1 fec0::2 any -P out ipsec
    esp/transport//require ah/transport//require;
```

```
spdadd fec0::2 fec0::1 any -P in ipsec
    esp/transport//require ah/transport//require;
```

- Configuration at Host B:

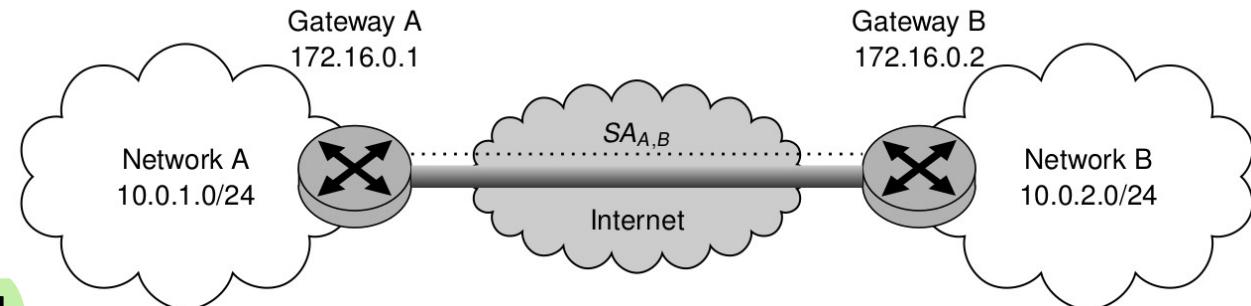
```
spdadd fec0::2 fec0::1 any -P out ipsec
    esp/transport//require ah/transport//require;
```

```
spdadd fec0::1 fec0::2 any -P in ipsec
    esp/transport//require ah/transport//require;
```



Yet another

- Example
ESP Tunnel for VPN
- Configuration at Gateway A



```
spdadd 10.0.1.0/24 10.0.2.0/24 any -P out ipsec  
esp/tunnel/172.16.0.1-172.16.0.2/require;
```

```
spdadd 10.0.2.0/24 10.0.1.0/24 any -P in ipsec  
esp/tunnel/172.16.0.2-172.16.0.1/require;
```

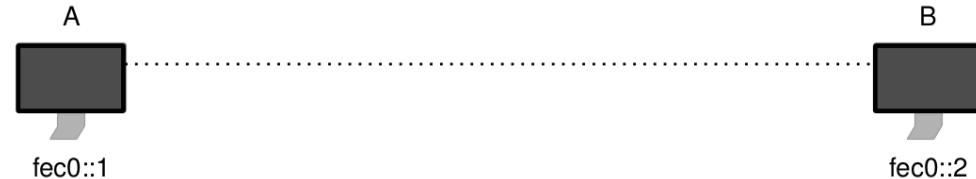
- Configuration at Gateway B:

```
spdadd 10.0.2.0/24 10.0.1.0/24 any -P out ipsec  
esp/tunnel/172.16.0.2-172.16.0.1/require;
```

```
spdadd 10.0.1.0/24 10.0.2.0/24 any -P in ipsec  
esp/tunnel/172.16.0.1-172.16.0.2/require;
```



Manual setup of Security Associations



- Example

Manually setting up an AH SA

```
# basic syntax: add src dst proto spi -A authalgo key;  
add fec0::1 fec0::2 ah 700 -A hmac-md5  
    0xbff9a081e7ebdd4fa824c822ed94f5226;  
  
add fec0::2 fec0::1 ah 800 -A hmac-md5  
    0xbff9a081e7ebdd4fa824c822ed94f5226;
```

- Manually setting up an ESP SA:

```
# basic syntax: add src dst proto spi -E encalgo key;  
add fec0::1 fec0::2 esp 701 -E 3des-cbc  
    0xdafb418410b2ca6a2ba144561fab354640080e5b7a;  
  
add fec0::2 fec0::1 esp 801 -E 3des-cbc  
    0xdafb418410b2ca6a2ba144561fab354640080e5b7a;
```



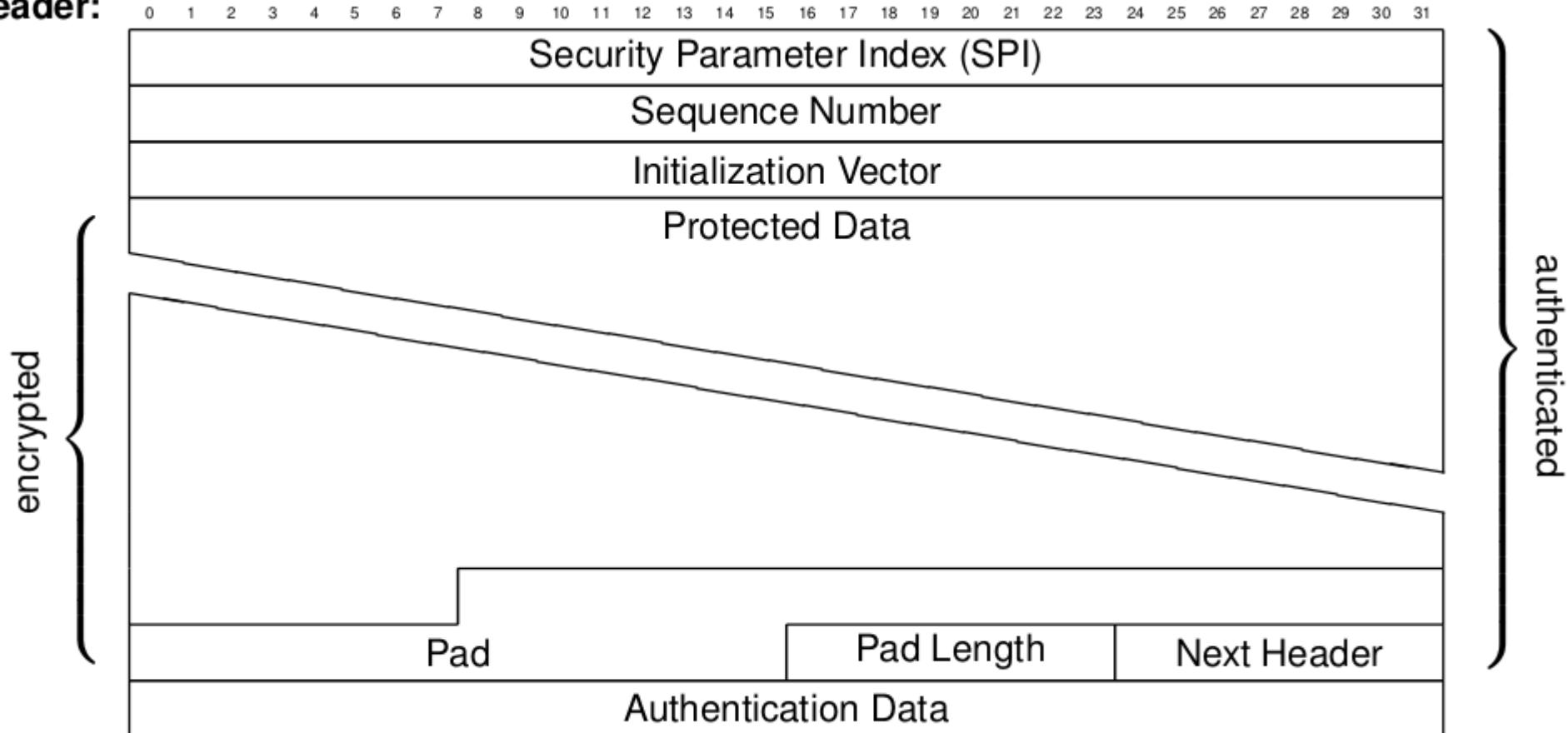
Disclaimer

WARNING: Setting up an SA manually is error prone!

- The administrator might choose insecure keys
- The set of SAs might be inconsistent
- It is better to rely on an IKE daemon for setting up SAs
- **We do it only for educational purposes!**

Encapsulating Security Payload

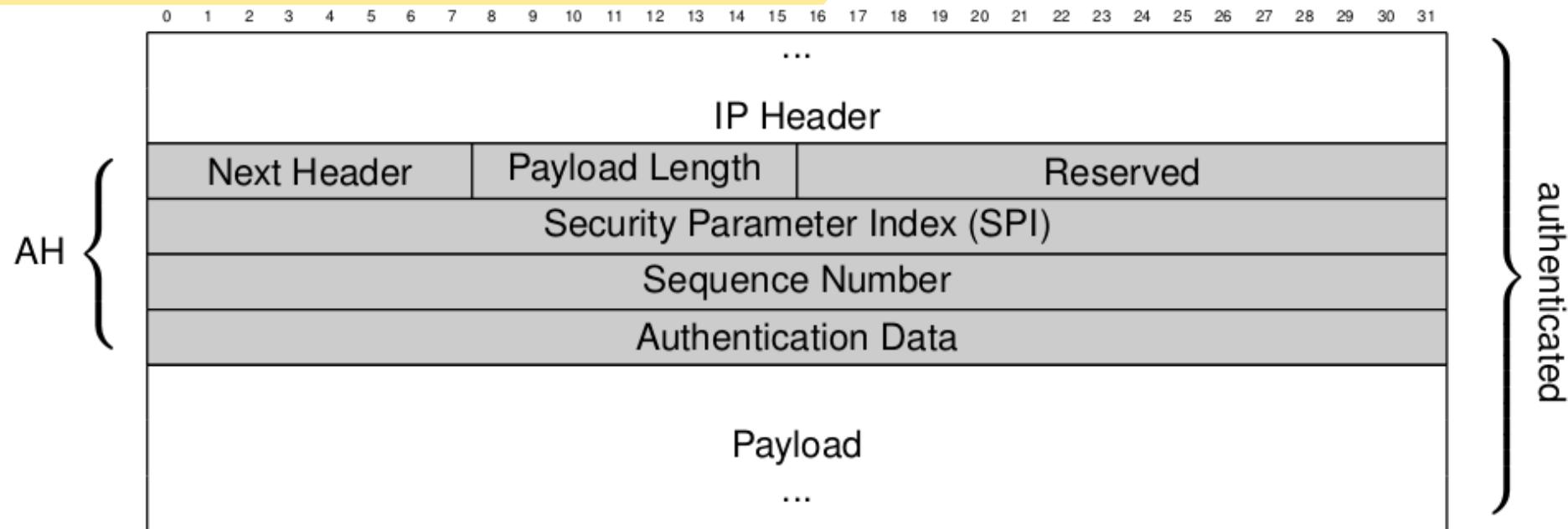
ESP Header:



The ESP immediately follows an IP/AH header and is indicated by Next Header = 50



Authentication Header



The AH immediately follows an IP Header and is indicated by Next Header = 51

Both ESP and AH can be applied at the same time with different ordering

- If ESP is applied first, AH is outer header
 - Advantage: ESP is also protected by AH
 - Consequence: Two SAs (one for each of AH/ESP) are needed for each direction



AH: fields changing in transit

- Although the AH protects the outer IP Header, some of its fields must not be protected, as they are subject to change during transit
 - This also applies to mutable IPv4 options or IPv6 extensions
- Such fields are assumed to be zero for MAC computation

Outer IP Header	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31						
	Version		IHL		Type of Service			Total Length																														
	Identification										Flags		Fragment Offset																									
	Time To Live				Protocol			Header Checksum																														
	Source Address																																					
	Destination Address																																					

All immutable fields, options and extensions (gray) are protected



Internet Key Exchange protocol v2

- A standardized authentication & key management protocol to dynamically establish SAs between two endpoints
- Standardized in [RFC4306] in December 2005
- Parts of IKEv1 poorly specified and spread over multiple RFCs
- IKEv2 provides unified authentication and key establishment
- Tries to achieve trade-off between features, complexity and security under realistic threat model
- [RFC5996] obsoletes [RFC4306]

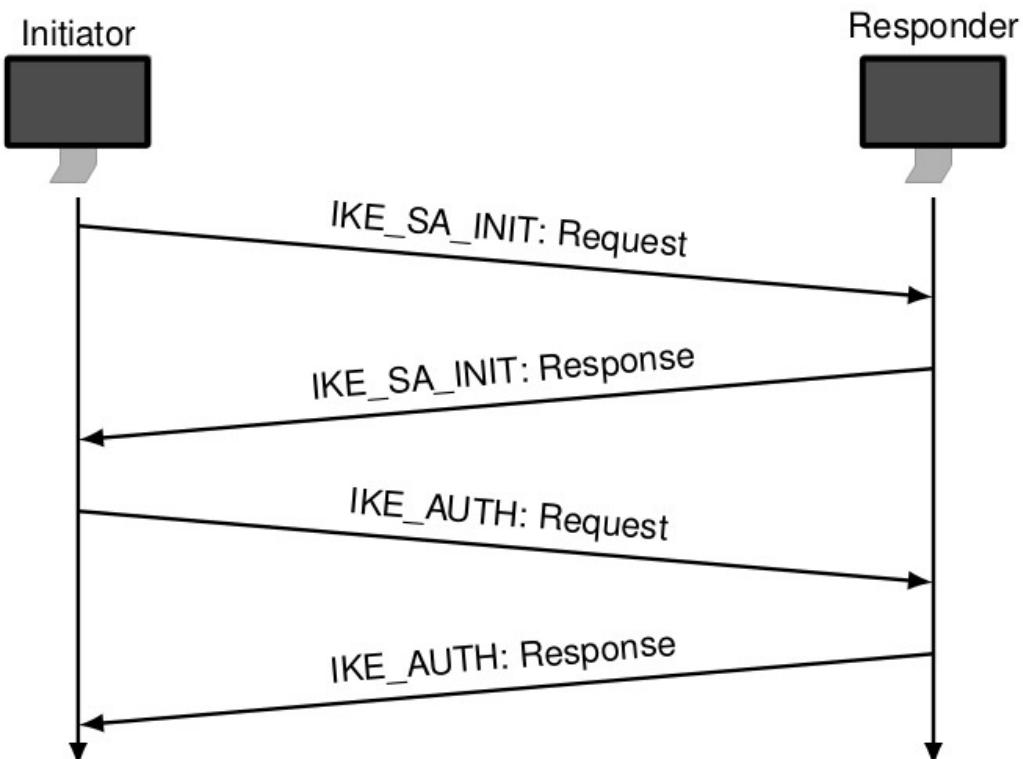


IKEv2 General Properties

- Runs on UDP ports { 500, 4500 }
- Mutual authentication of the Initiator and Responder
- Negotiation of cryptographic suites
 - i.e., a complete set of algorithms used for SAs
- Support for DoS mitigation through use of cookies
- Integrated support for requesting an IP address (useful for VPNs)
- IKEv2's latency is 2 round trips (i.e., 4 messages) in the common case

IKEv2 exchanges (phases)

- IKEv2 communication consists of message pairs
- Request and response
- One pair (request, response) is called an exchange
- An IKEv2 protocol run starts with two exchanges
 - IKE_SA_INIT
 - IKE_AUTH





SA_INIT and AUTH

IKE_SA_INIT (phase 1)

- Negotiates security parameters for a security association (IKE_SA)
- Sends nonces and Diffie-Hellman values
- IKE_SA is a set of security associations for protection of remaining IKE exchanges

IKE_AUTH (phase 2)

- Authenticates the previous messages
- Transmits identities
- Proves knowledge of corresponding identity secrets
- Creates first CHILD_SA
 - A CHILD_SA is a set of SAs, used to protect data with AH/ESP
 - The term CHILD_SA is synonymous to the common definition of an SA for IPSec AH and ESP



Other exchanges

CREATE_CHILD_SA

- Used to create another CHILD_SA
- Can also be used for rekeying

INFORMATIONAL

- Keep-Alive
- Deleting an SA
- Reporting error conditions, ...



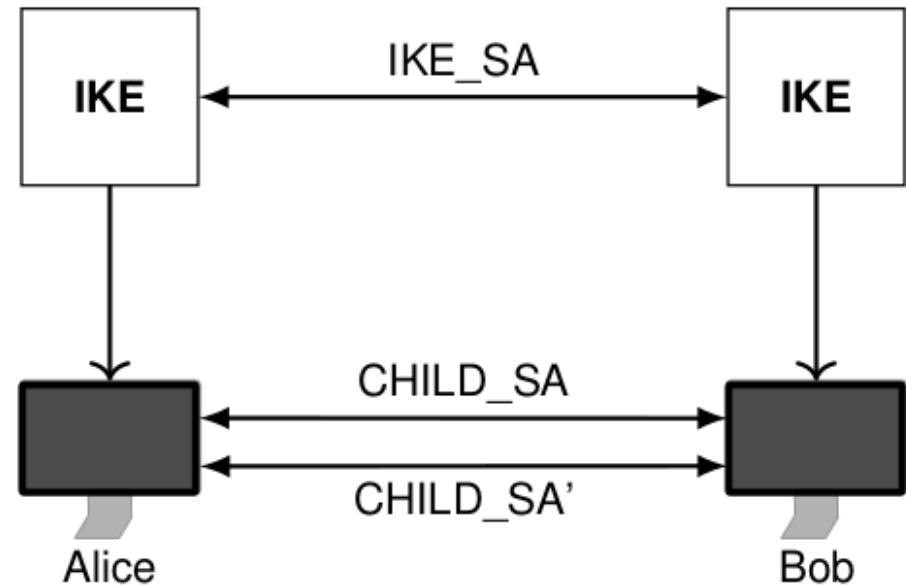
Negotiation and authantication

IKE_SA_INIT

- IKE_SA_INIT negotiates:
 - Encryption algorithmn
 - Integrity protection algorithm
 - Diffie-Hellman group (i.e. DH parameters p and g)
 - Pseudo-Random function prf
- IKE_AUTH realizes authentication via public key signatures or long-term pre-shared secret
 - Authentication by signing (or calculating MAC K of) a block of data
 - The resulting value (AUTH) is transmitted in the IKE_AUTH exchange
 - Authentication is conducted by verifying the validity of the received AUTH payload

IKE outcome

- IKE_SA is a set of Security Associations established after the initial IKEv2 exchange (IKE_SA_INIT)
- IKE_SA is used to encrypt and integrity protect all the remaining IKEv2 exchanges
- CHILD_SA is a set of Security Associations used to protect IP traffic with the AH/ESP protocol
 - AH provides data integrity and replay protection
 - ESP provides data integrity, replay protection and encryption





IPsec in Linux kernel

- Supported natively, but...
- Hard to setup
- Deprecated tools, still useful and used (→Android):
 - ipsec-tools and racoon (IKEv1 daemon)
- Complete packages
 - Strongswan
 - Libreswan



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>
- You can move in the directory and run **lstart**
 - **NOTE:** launch docker first or the first **lstart** attempt can (...will...) fail



Preparation for ex3, ex4, ex5, ex6

- In the host kathara you need to install ipsec-tools and racoon
 - ipsec-tools for using the setkey program
 - racoon is the IKE daemon
- Both are deprecated, so you need to start the labs with an old kathara docker image
 - Find and tag a possible docker image
 - docker images (show the installed images)
 - docker tag 6b9b242d2656 kathara/quagga:ipsec-tools
 - Modify the lab.conf files to use the tagged image
 - pc1[image]=kathara/quagga:ipsec-tools



Ipsec references

- IPsec reference for linux:
 - <http://www.ipsec-howto.org/x299.html>
- Have a look at the manuals:
 - man setkey
 - <https://linux.die.net/man/8/setkey>
 - man racoon
 - <https://linux.die.net/man/8/racoon>
- When a pre-shared key is required, you can use
 - `dd if=/dev/random count=16 bs=1 | xxd -ps`

num of blocks = 16

block size = 1 byte

Remember:
128b = 16B
192b = 24B
256b = 32B

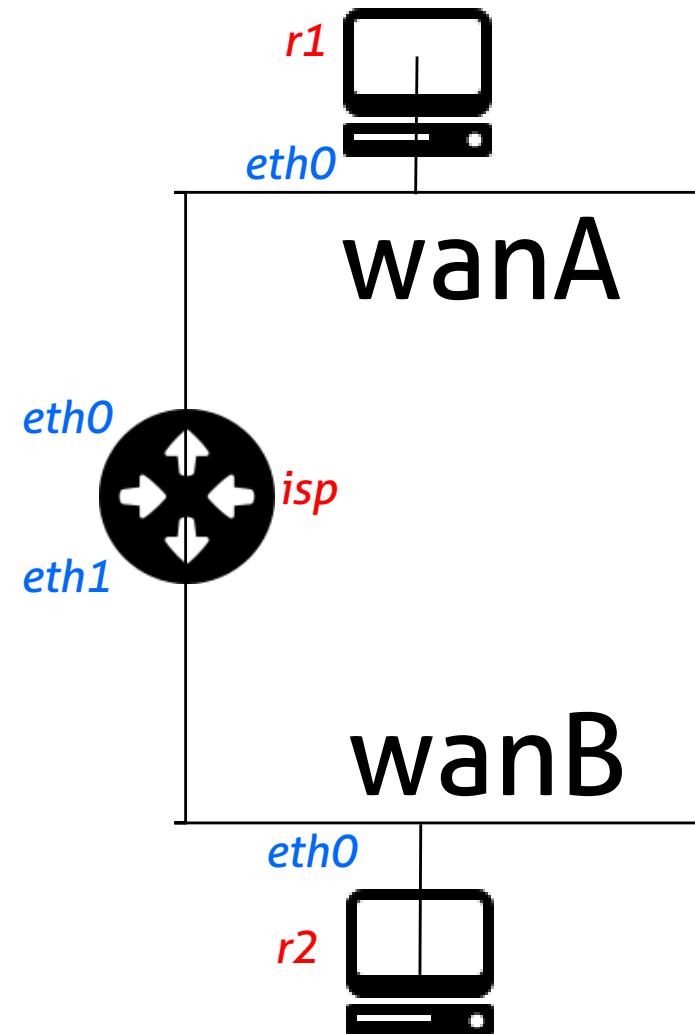


Lab activity: ex3, ex4



pnd-labs/lab5/ex3 and ex4

- You have to setup the addressing
 - Follow README instructions
- IPv4 in ex3
- IPv6 in ex4
- See the differences in how AH and ESP are implemented





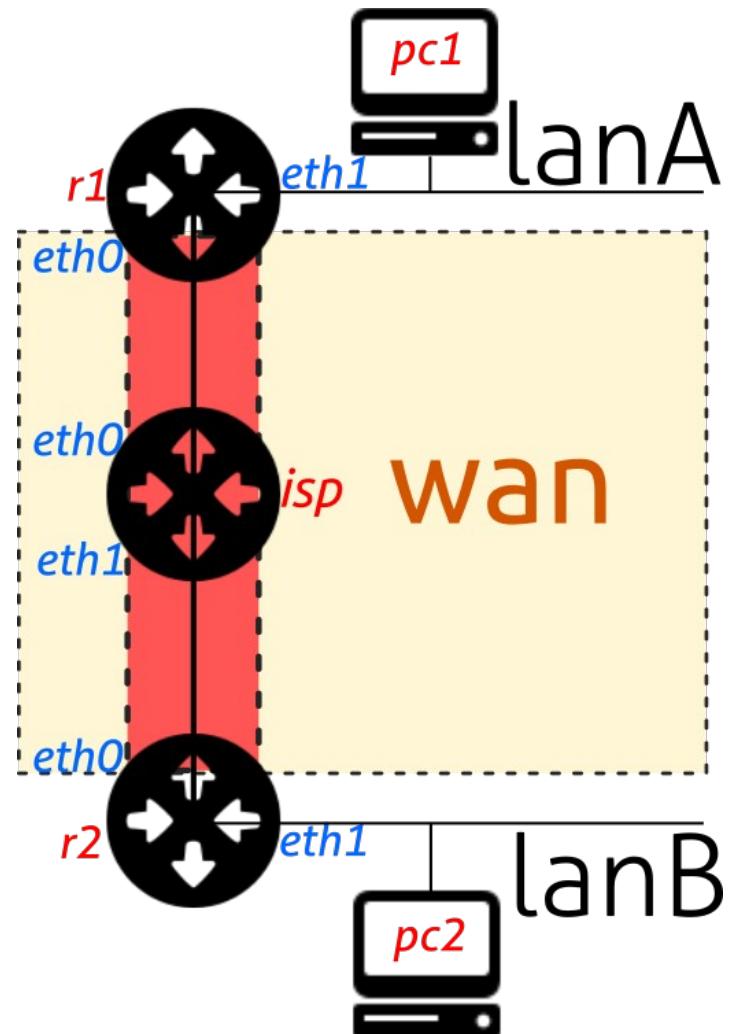
Notes about setkey

- You can run it with the command
 - `/etc/init.d/setkey start`
 - It can also be run using the `-c` flag, that accepts the directives from stdin
 - It can also be run using the `-f` flag, that accepts the directives from a file
- The default configuration file is located at
 - `/etc/ipsec-tools.conf`

Lab activity: ex5, ex6

pnd-labs/lab5/ex5 and ex6

- IPv4 addressing already setup
- Configure r1 and r2 to manage a VPN tunnel with IPsec between lanA and lanB
- Ex6: do it with the **raccoon daemon**



Lab activity: ex7, ex8



strongSwan

- strongSwan is basically a keying daemon
 - It uses IKEv1 and IKEv2 to establish security associations (SA) between two peers
 - The IKE daemon is **charon**, configured with the “VICI” control interface (Versatile IKE control interface)
- A full configuration, then, is called a CHILD_SA made of:
 - the actual IPsec SAs (algorithms and keys used to encrypt and authenticate the traffic)
 - the policies that define which network traffic shall use such an SA
- The actual IPsec traffic is handled by the network and IPsec stack of the operating system kernel
 - <https://docs.strongswan.org/docs/5.9/howtos/introduction.html>



Authentication options

- Public Key Authentication
 - RSA, ECDSA or EdDSA X.509 certificates to verify the authenticity of the peer
- Pre-Shared-Key (PSK)
 - A pre-shared-key is an easy to deploy option but it requires strong secrets to be secure
- Extensible Authentication Protocol (EAP)
 - This covers several possible authentication methods, based on username/password authentication (EAP-MD5, EAP-MSCHAPv2, EAP-GTC) or on certificates (EAP-TLS), some can even tunnel other EAP methods (EAP-TTLS, EAP-PEAP)
- eXtended Authentication (Xauth)
 - XAuth provides a flexible authentication framework within IKEv1



Configuration Files

- strongSwan is configured with
 - The `swanctl` command line tool
 - The `swanctl.conf` configuration file in the `swanctl` directory
- Global strongSwan settings as well as plugin-specific configurations are defined in `strongswan.conf`
 - Alternatively, the legacy ipsec stroke interface and its `ipsec.conf` and `ipsec.secrets` configuration files may be used.



Useful commands

- ipsec
 - start|stop|restart|status|statusall
 - listcerts|listalgs|listpubkeys
- swanctl
 - --load-creds|--load-conns|--load-all
 - --list-sas
 - --initiate --child <connection>

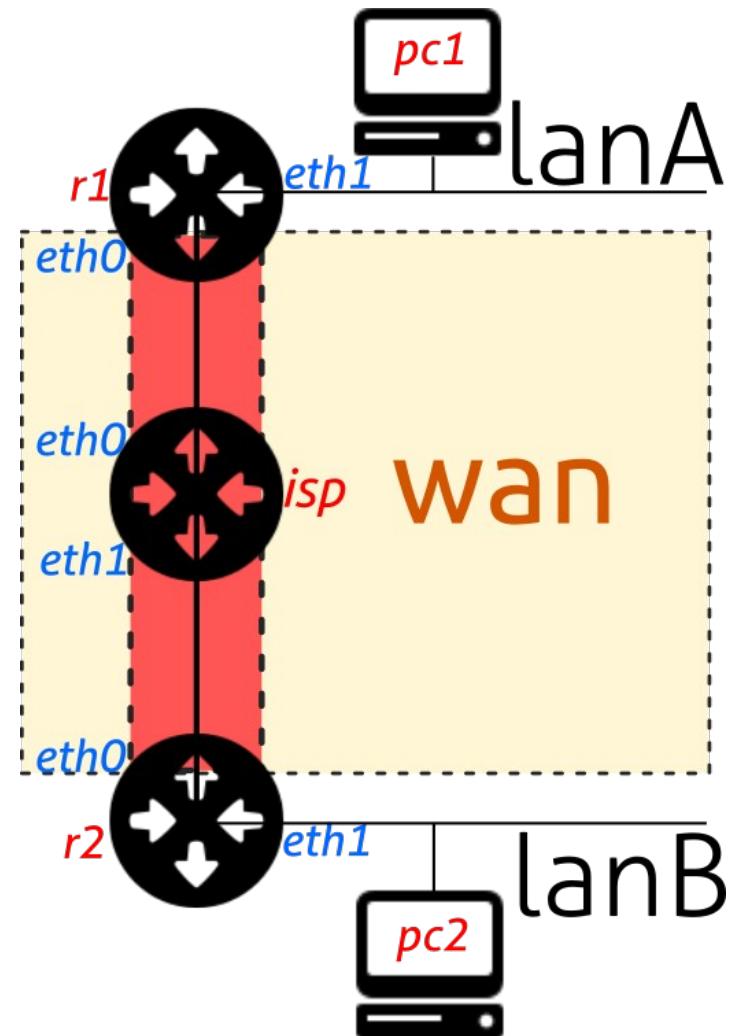


Proposals

- The list of preferences, please refer here:
 - <https://wiki.strongswan.org/projects/strongswan/wiki/IKEv2CipherSuites>
- Not so straightforward, use with caution
 - Safe choice: default

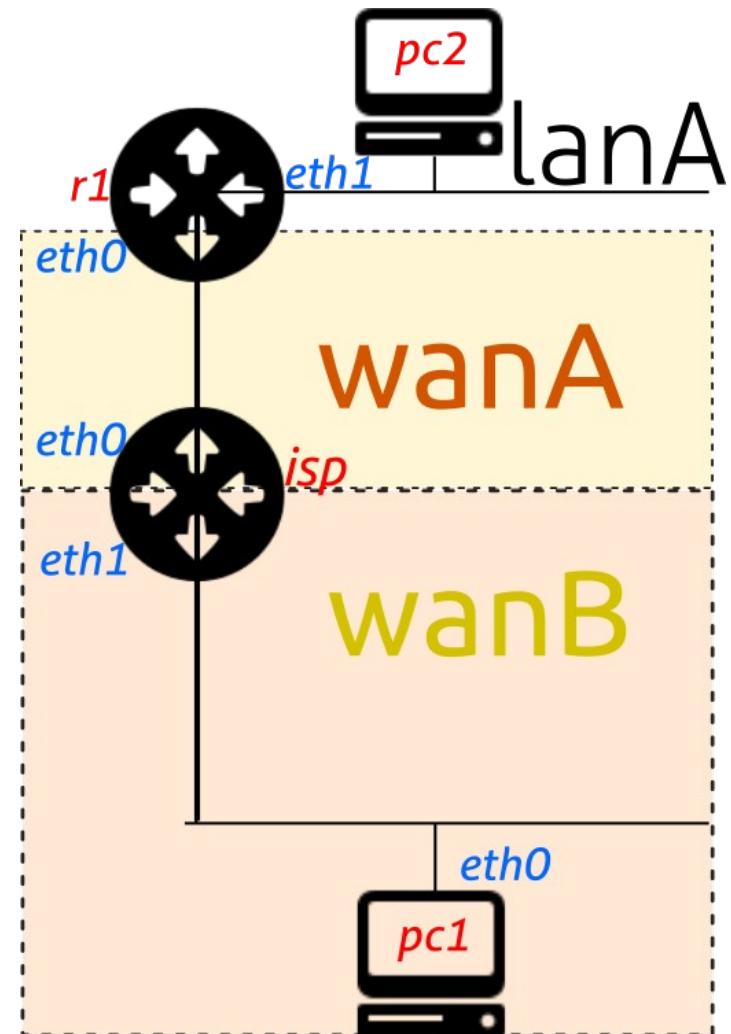
pnd-labs/lab5/ex7

- IPv4 addressing already setup
- Configure r1 and r2 to manage a VPN tunnel with IPsec between lanA and lanB
- Use the strongSwan framework:
 - ipsec start
 - swanctl



pnd-labs/lab5/ex8

- IPv4 addressing already setup
- Configure r1 to be the VPN GW for pc1 in order to access lanA
- Use the strongSwan framework:
 - ipsec start
 - swanctl





That's all for today

- **Questions?**
- See you on Monday
- Resources:
 - <http://www.ipsec-howto.org/>
 - <http://www.unixwiz.net/techtips/iguide-ipsec.html>
 - Chapter 24 textbook
 - Virtual private networking, Gilbert Held, Wiley ed.
 - Guide to IPsec VPNs, NIST800-77
 - Guide to SSL VPNs, NIST-SP800-113
 - http://www.tcpipguide.com/free/t_IPSecurityIPSecProtocols.htm
 - <https://docs.strongswan.org/docs/5.9/howtos/introduction.html>

Practical Network Defense

Master's degree in Cybersecurity 2021-22

Forward proxy activity

Angelo Spognardi
spognardi@di.uniroma1.it

*Dipartimento di Informatica
Sapienza Università di Roma*

Squid activity: as a forward proxy



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



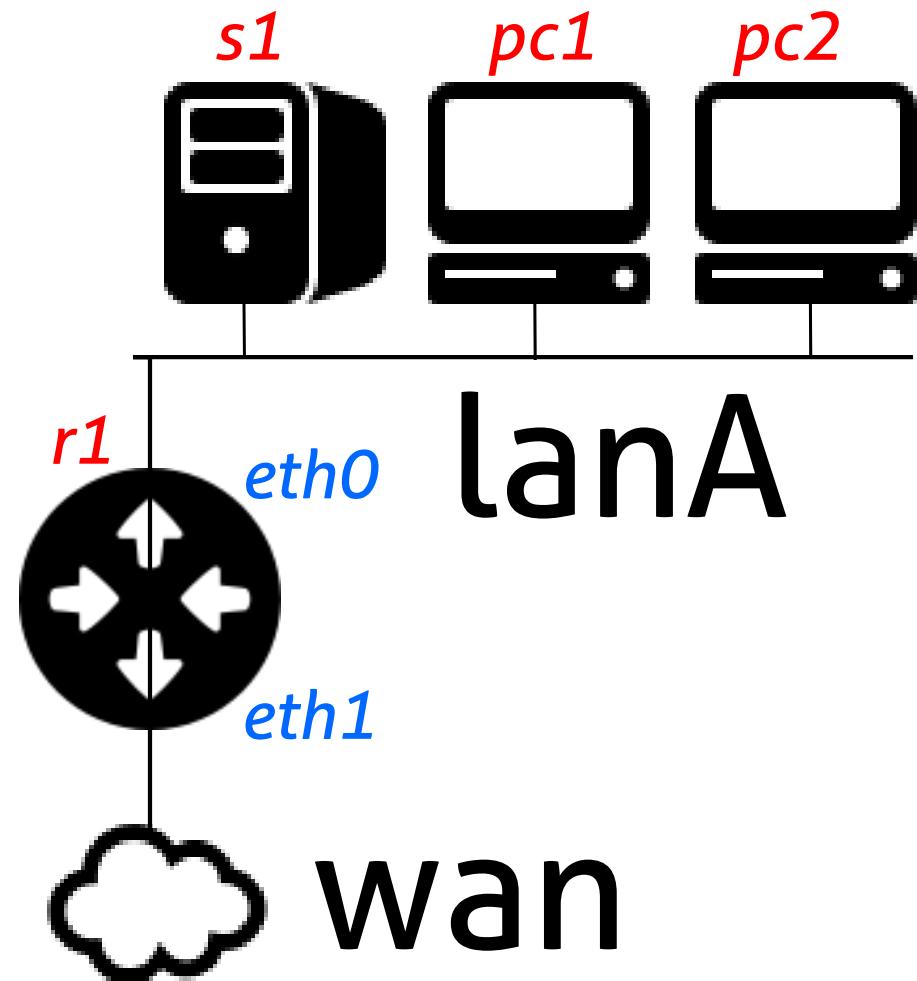
The kathara VM

- 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>
- You can move in the directory and run **lstart**
 - **NOTE:** launch docker first or the first **lstart** attempt can (...will...) fail

Lab activity: lab6/ex1

pnd-labs/lab6/ex1: squid proxy

- In this lab you have to incrementally build the squid configuration
- You can start reading the following resource page:
 - <https://www.howtoforge.com/squid-proxy-on-rhel5-centos-everything-that-you-should-know-about>
 - Most of the activity can be solved looking at the above resource
- Firstly, in r1 enforce the policy that only the proxy can use http and https (and obviously DNS) with iptables
 - Verify that pc1 and pc2 cannot use internet
- Take a look at the simple squid configuration file at /s1/etc/squid/squid.conf





Activity 1

- Configure pc1 and pc2 to use the squid proxy
- pc1\$ export http_proxy=192.168.100.80:3128
- Verify you can connect with http to a website (that uses http!)
 - Ex: <http://www.columbia.edu/~fdc/sample.html>
 - Check with wireshark what happens
- Modify the squid.conf so that only pc1 can use http
 - Check with wireshark what happens
- Modify again the squid.conf to use a file with blacklisted websites



Activity 2

- Configure squid so that it can also allow https
- pc1\$ export https_proxy=192.168.100.80:3128
- To work, this requires the use of the CONNECT method
- Extra details are provided in the original squid.conf file, found at s1/etc/squid/squid.conf.bak
 - Reference:
 - https://wiki.squid-cache.org/SquidFaq/SecurityPitfalls#The_Safe_Ports_and_SSL_Ports_ACL
 - When done, check with wireshark what happens



Activity 3

- Configure squid so that it requires the users to authenticate with username and password
- You can find more info about authentication methods on this resource:
 - <http://www.squid-cache.org/Doc/man/>
- You can use the ncsa method



Activity 4

- Configure squid to perform SSL Bump, in order to intercept the https traffic generated by the client pc1
- Reference:
 - <https://wiki.squid-cache.org/Features/HTTPS>



Activity 5

- Configure squid and the topology to realize the configuration of a transparent firewall



That's all for today

- **Questions?**
- See you next lecture!
- References:
 - Ari Luotonen, Kevin Altis, [World-Wide Web Proxies, 1994](#)
 - http://httpd.apache.org/docs/current/mod/mod_proxy.html
 - https://en.wikipedia.org/wiki/Proxy_server
 - Classical vs Transparent IP Proxies, [RFC 1919](#)
 - SOCKS 5, [RFC 1928](#)
 - HTTP 1.1, [RFC 7230](#)
 - Policy based routing and Linux advanced routing and traffic control
 - ICAP, [RFC 3507](#)
 - <https://wiki.squid-cache.org/FrontPage>

Practical Network Defense

Master's degree in Cybersecurity 2021-22

SIEM

Angelo Spognardi
spognardi@di.uniroma1.it

*Dipartimento di Informatica
Sapienza Università di Roma*



Security Information and Event Management (SIEM)

- An approach to cybersecurity combining:
 - Security information management (SIM)
 - Collects log data for analysis, alerting responsible individuals of security threats and events
 - Security event management (SEM)
 - Conducts real-time system monitoring, notifies network admins of important issues, and establishes event correlations
- Generally made of multiple monitoring and analysis components meant to help organizations detect and mitigate threats
 - Not a single tool or application, but a set of different building blocks that all constitute part of a system



No SIEM standard

- There is no standard SIEM protocol or established methodology, but most SIEM systems know how to:
 - automatically collect and process information from distributed sources
 - store it in one centralized location
 - correlate between different events
 - produce alerts and reports based on this information
 - help for compliance and security incident management (digital forensics)
- They can be agent-based or agentless



Security Information and Event Management (SIEM)

- Then, it should be providing the following collection of services:
 - Log management
 - IT regulatory compliance
 - Event correlation
 - Active response
 - Endpoint security
- **Log != Event**
 - but we talk about “event logging”





Log management

- Nodes in an IT system, particularly the more important or critical nodes, send relevant system and application events (logs) to a centralized database that is managed by the SIEM application
- This SIEM database application first parses and normalizes the data sent by the numerous and very different types of nodes on an IT system
- Then the SIEM typically provides **log storage, organization, retrieval, and archival services** to satisfy the log management requirements that businesses may have



Logs enable analysis

- The SIEM system lends itself to the additional use of near real-time analysis and data mining on the health and security status of all the IT systems feeding their data into the SIEM system
- The more nodes that feed into your SIEM system, the more **complete** and **accurate** your vision is of the IT system as a whole



IT Regulatory Compliance

- Once logs are stored, you can build filters or rules and timers to **audit** (monitor against a standard) and validate **compliance**, or to identify violations of compliance requirements imposed upon the organization
 - Examples: monitoring the frequency of password changes, identifying operating system (OS) or application patches that fail to install, and the auditing frequency of antivirus, antispyware, and IDS updates for compliance purposes...
- SIEM generally can produce **reports** often needed by businesses to provide evidence of self-auditing and to validate their level of compliance



Event Correlation

- Consider various conditions before triggering an alarm
- The correlation engine on a SIEM can investigate and consider (**correlate**) other events that are not necessarily homogeneous
- It can provide a more complete picture of the health status of the system to rule out specific theories on the cause of given events



Active response

- Activate procedures after the identification of given (security) events
 - Automatic response
 - Manual response
- The SIEM triggered, automated, and active response to the perceived threat would probably occur much faster
 - Like: adding IP and port filters on the access control list (ACL) on a router or firewall



Endpoint Security

- Most SIEM systems can monitor endpoint security to centrally validate the security “health” of a system
- Some SIEM systems can even manage endpoint security, actually making adjustments and improvements to the node’s security on the remote system
 - Ex: configuring firewalls and updating and monitoring AV, antispyware, and antispam
- Some SIEM systems can push down and install the updates, or in Active Response mode, adjust the ACL on a misconfigured personal firewall



Logs are fundamental

- Logs are the events that your network produces
- Needed to extract information about the events
- Without them, it is impossible to achieve any security management
- Typical questions:
 - How long must you retain the logs?
 - Data retention and data destruction
 - How much log information will you be required to retain?
 - What kind of information system logs are you required to retain (and eventually analyze)?



Log sources

- Log management apparently easy
- Complicated task as varied sources of information are included and higher levels of functionality, such as filtering, correlating, and reporting, are enabled
- Examples:
 - Syslog of servers and end-user computers
 - Alerts from IDS/IPS and antivirus
 - Flow data
 - Domain controllers
 - Databases
 - Switches and routers
 - VPN gateways
 - Firewalls
 - Web filters and proxies
 - ...



Other points for log

- Which devices will you collect events from?
 - Critical servers, devices providing access to critical servers, IDS
 - Optional: network endpoints (maybe only aggregated via other services)
- Which events will you collect?
 - Debug info, log-in records, configuration changes, alerts...
- How long will you keep the logs?
 - Balance between needs and desires
 - Usually, agree on the regulatory
- Where will you store the logs?
 - Local storage, cloud, hybrid solutions



Event data vs state data

- Event data (logs) provide you with an exact list of all events that occurred on your server, network, or website
 - Managing logs tells what happened and when
- State data gives you the view of the overall state of the system
 - Configurations
 - Current applications
 - Active users
 - Processes
 - Registry settings
 - Vulnerabilities



Logging solutions

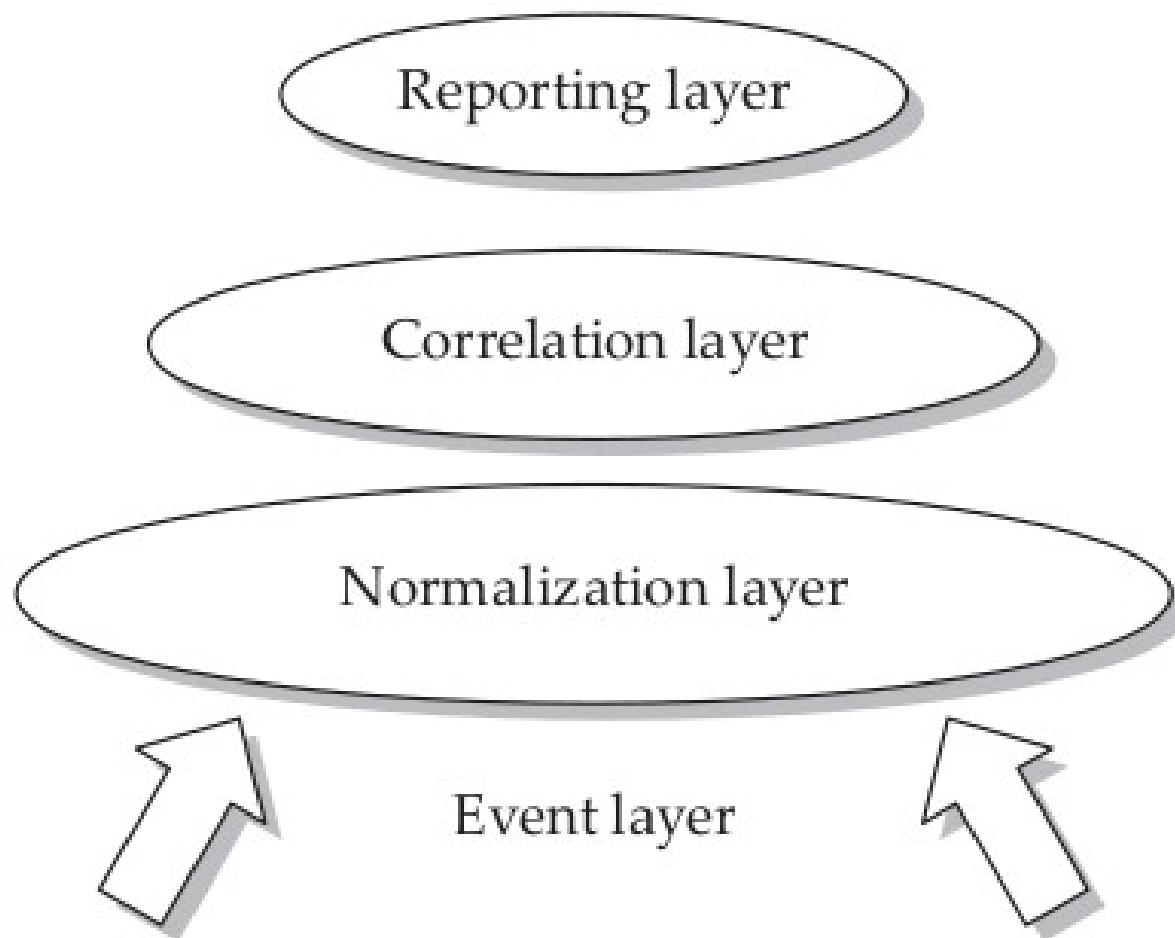
- Syslog, syslog-NG
- Splunk
- LogStash
- Graylog



logstash



SIEM stack





Log correlation means

- Monitoring the incoming logs for
 - Logical sequences
 - Patterns
 - Relationships
 - Values
- The ultimate goal is to analyze and identify events invisible to individual systems
- Generally make use of “supporting data”



SIEM supporting data

- Data collected by other sources that can be imported into the SIEM to make comparative determinations
- Example: asset management data
 - Names, IP addresses, operating systems, software versions
 - Geo-location information
- They can be used as weights to prioritize and escalate alerts



Event correlation

- Correlation engines usually are the most distinguishing feature of SIEM
 - And also what the vendors want to sell...
 - Mainly closed source
- Before they have to perform event normalization
 - Message logs are in standard formats, but are not homogeneous
 - “drop” from one vendor firewall, “block” from another
 - Define a common syntax to represent events in the SIEM and apply it to the logs
- Make use of correlation rules
 - Rules that can trigger alerts or actions
 - Example: Perl SEC (Simple Event Correlator), SolarWinds
 - Machine Learning



Endpoint security

- Patching the operating system and major applications
- Antivirus and antispyware updates
- Firewalls—making sure they are on and configured properly
- Host Intrusion Detection Systems (HIDS) and Host Intrusion Protection Systems (HIPS)
- Configuration management
- Management of removable media, such as USB drives and CD and DVD burners
- Network Access Control (NAC)
- Network Intrusion Detection Systems (NIDS) and Network Intrusion Protection Systems (NIPS)



IT regulatory compliance

- All forms of compliance ask the fundamental question related to diligence:
 - Have you taken the steps to perform your responsibilities to securely manage the information in your control—which a reasonable person would expect of someone in your position?
- In other words, if you had to defend your actions in this regard in front of a jury of your peers, would you be comfortable stating that you had used available best practices and sufficient effort to perform your duties?
 - Think about GDPR



Provide evidences of best practices

- Implementing technologies to protect and detect intrusions is not enough
- This should be **provable**
- The log server has to be reliable. For example
 - Use TCP transport
 - Use encrypted storage
- Moreover, it could be important to also sign the logs
 - Authentication and integrity



Compliance tools

- SIEM can also include compliance checklists
 - Ex: SPLUNK
- The reports that SIEM can generate can be used as evidences for the IT regulatory compliance
 - Example: <https://gdpr.eu/checklist/>
- Proper configuration can be quite complex → professionals of regulatory compliance



Additional features

- Support for open-source threat intelligence feeds
- Real-time analysis and alert (IDS-like)
- Optionally, automated response (IPS-like)
- Advanced search capabilities
 - Elasticsearch
- Historical and forensic analysis



Threat intelligence

- Threat intelligence feeds and reports help security officers in making decisions concerning organizational security
- Threat intelligence is the analysis of data using tools and techniques to generate meaningful information about existing or emerging threats targeting the organization that helps mitigate risks
- Threat Intelligence helps organizations make faster, more informed security decisions and change their behavior from reactive to proactive to combat the attacks



Digital forensics

- Digital forensic science is a branch of forensic science focused on the recovery and investigation of material found in digital devices and cybercrimes
- Digital forensics was originally used as a synonym for computer forensics but has expanded to cover the investigation of all devices that store digital data
- Digital forensics is concerned with the identification, preservation, examination and analysis of digital evidence, using scientifically accepted and validated processes, to be used in and outside of a court of law

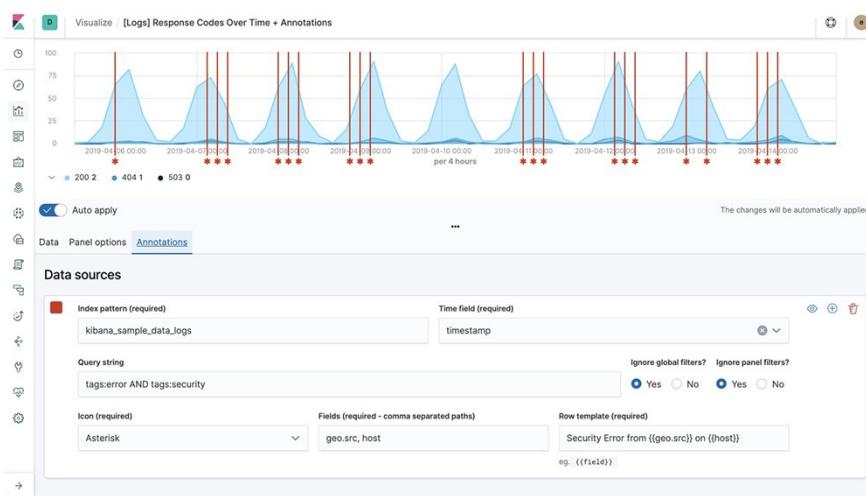
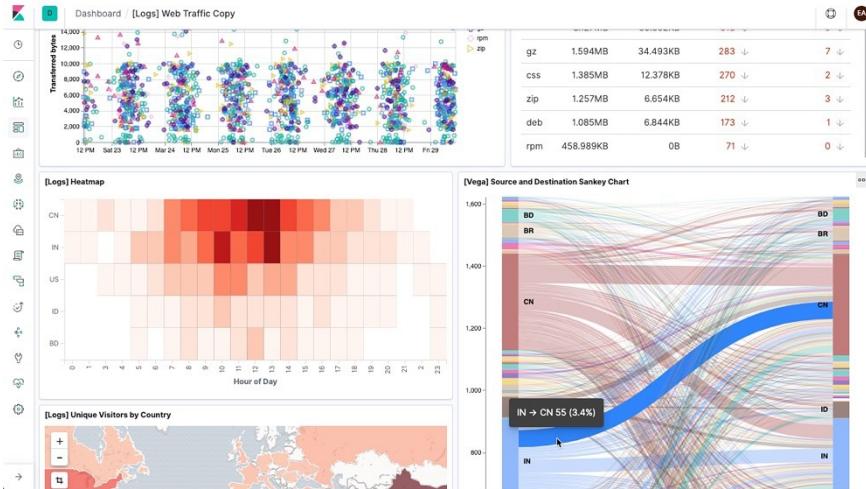


SIEM operational interfaces

- Dashboards and maps → pull of information
 - A way to present information in a way that administrators can understand at glance
 - It is a graphical and organized representation of alerts, event data, and statistical information
 - It allows administrators to see patterns, understand trends, identify unusual activity
 - Dashboards are also key differentiator among the different SIEM products on the market
- Alerts → push of information
 - Do not require human diligence to notice something important is happening



Dashboards and maps

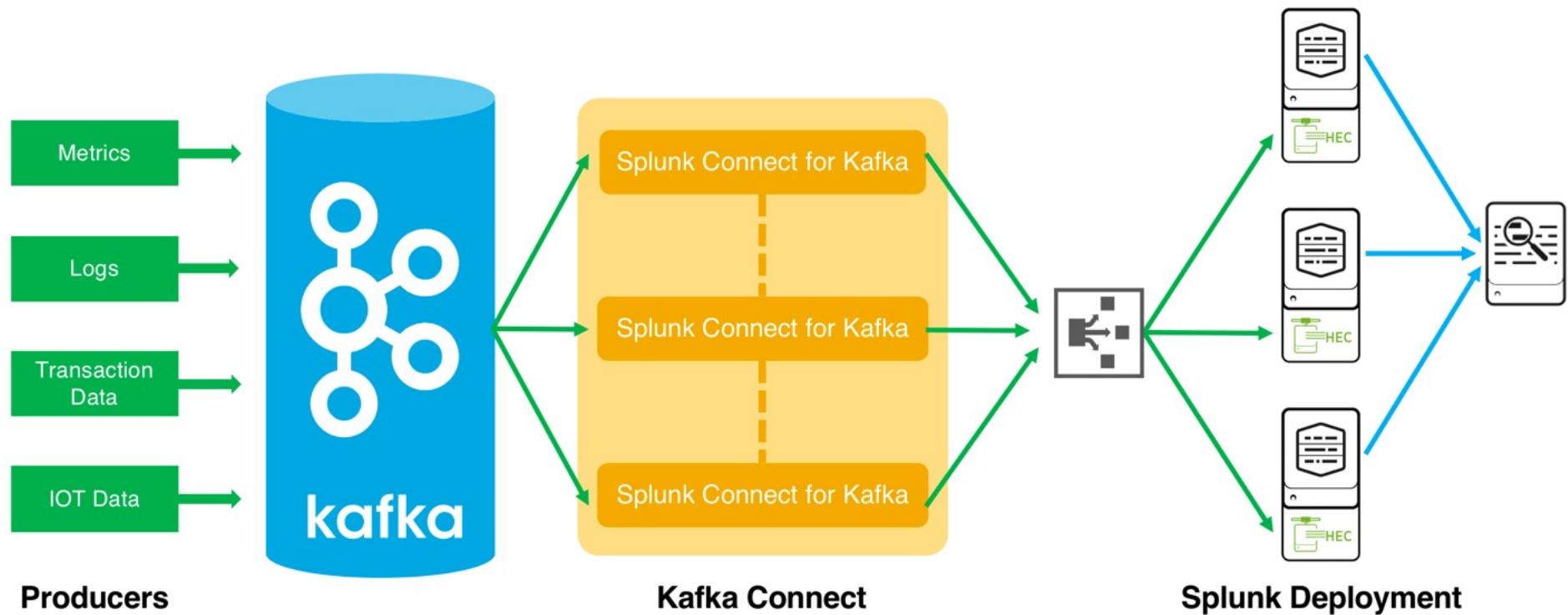




SIEM → complexity!

- A survey conducted in 2013 by elQnetworks revealed that:
 - “managing the complexity of the product is considered the biggest headache when using SIEM
 - followed by lack of trained personnel to manage the product
 - and lack of integration with other products”
 - 31 percent would prefer to replace their existing SIEM solution for better cost savings
 - 25 percent have invested more than month in professional services since the implementation of their current SIEM solution
 - 52 percent require two or more full-time employees to manage the chosen solution

SIEM complexity: clustered setting



<https://www.splunk.com/blog/2018/04/25/unleashing-data-ingestion-from-apache-kafka.html>



SIEM tools

- Splunk
- OSSIM – Alien Vault
- Security Event Manager (SEM) – SolarWinds
- AT&T Cybersecurity
- IBM QRadar
- RSA NetWitness Platform (RSA NWP) – Dell Technologies
- Security Management Platform (SMP) – Exabeam
- ArcSight ESM – Micro Focus
- LogRhythmNextGen SIEM Platform
- SELKS distribution
- Graylog
 - http://www.cryptos.com.mx/mx/sites/default/files/Gartner_MO_SIEM_2020.pdf



That's all for today

- **Questions?**
- See you next lecture!
- Suggested reading:
<https://www.sans.org/reading-room/whitepapers/detection/paper/37477>
- References:
 - Security Information and Event Management (SIEM) Implementation, D. Miller, S. Harris, A. Harper, S Vandyke, C. Blask, McGrawHill, 1st ed. 2011
 - Security Event Correlator: <https://simple-evcorr.github.io/>
 - SEC ruleset (check the thesis):
<https://github.com/markuskont/SagittariuSEC/>
 - Kibana 5 introduction [video](#), [live demo](#)
 - SELKS distribution: <https://github.com/StamusNetworks/SELKS>



Student's Opinions Questionnaires (OPIS)

- For the Practical Network Defense course
- Two options:
 - the infostud app (probably best option)
 - the infostud website
 - follow the following instructions
https://www.uniroma1.it/sites/default/files/field_file_allegati/vadevecum_opis_eng_27_11_2018_002_modalita_compatibilita.pdf
 - use this course code **ZI9J6C4M**
- **Be (pro-)positive!**



Practical Network Defense

Master's degree in Cybersecurity 2021-22

Reverse proxy activity

Angelo Spognardi
spognardi@di.uniroma1.it

*Dipartimento di Informatica
Sapienza Università di Roma*



Reverse proxy!

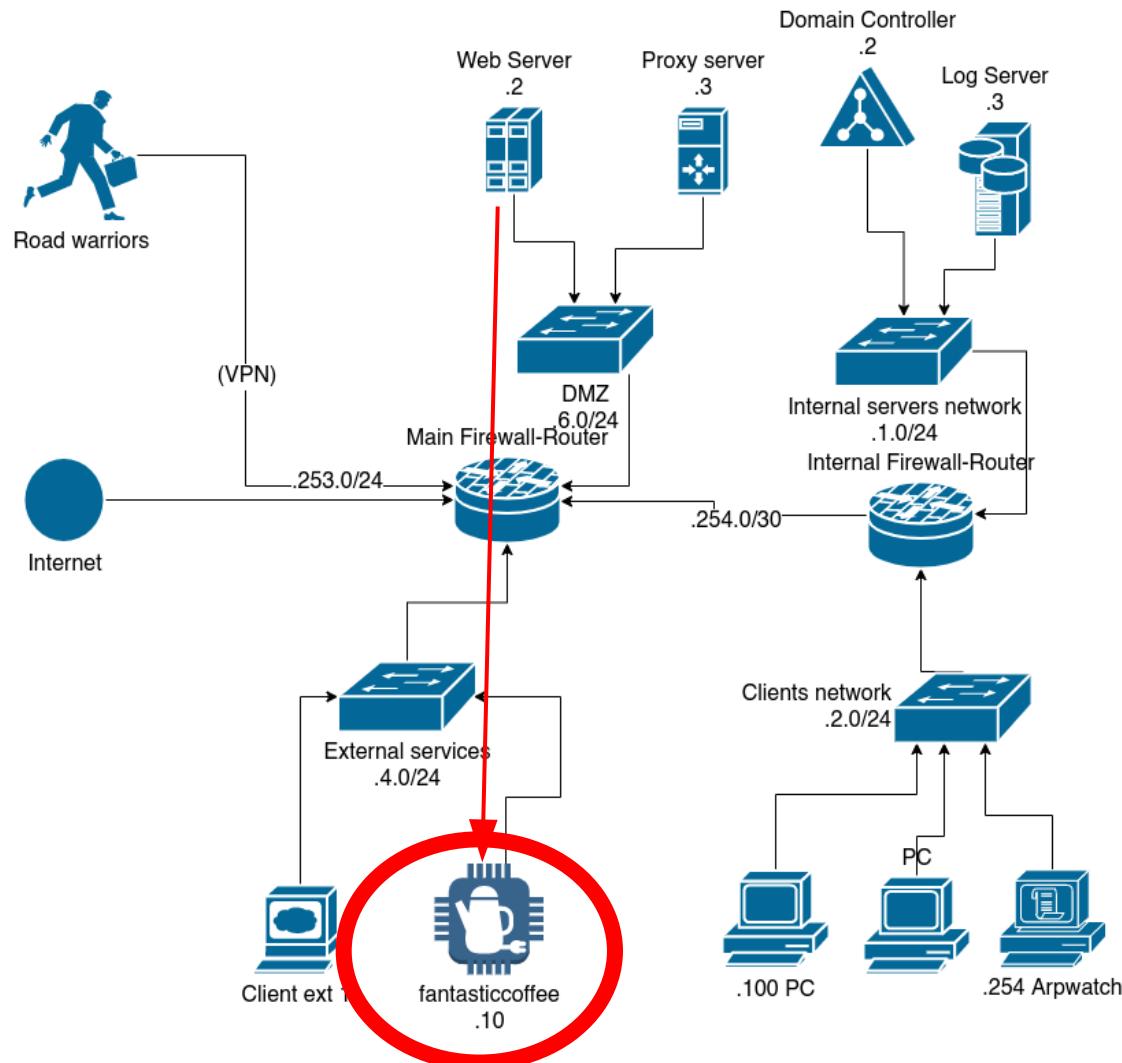


Activity: protect the coffee machine

- You have to create a reverse proxy to give access to the manufacturer of the ACME's coffee machine
- The machine ONLY uses HTTP and is, very likely, prone to security breaches (→ it has vulnerabilities, 100% guarantee!)
- Coffee machine can only be reached via the webserver of the company
- Use the apache mod_proxy module with TLS settings
 - You can generate certificate and keys within opnsense



Coffee machine in ACME co.





Hints

- You should enable the related modules and configure the SSL module

```
root# a2enmod ssl
root# a2enmod proxy
root# a2enmod proxy_http
```



Hints

- You should have something like this

```
<VirtualHost *:443>
    # ...default parameters...

    SSLCertificateFile      /etc/ssl/fantasticcoffee.acme-d.test.crt
    SSLCertificateKeyFile   /etc/ssl/fantasticcoffee.acme-d.test.key

    ProxyPreserveHost       On
    ProxyPass                /coffee/ http://100.64.4.10/
    ProxyPassReverse         /coffee/ http://100.64.4.10/
</VirtualHost>
```



One step more

- Configure modsecurity in your webserver so that you can filter the request to be forwarded to the coffee machine
- You should only allow requests with proper parameters
- Hints:
 - reverse engineer the web interface of the machine
 - derive the correct parameters
 - set up modsecurity to check the validity of requests



That's all for today

- **Questions?**
- See you next lecture!
- References:
- http://httpd.apache.org/docs/current/mod/mod_proxy.html
- https://en.wikipedia.org/wiki/Proxy_server
- [Modsecurity handbook](#)
- [Google](#)

Practical Network Defense

Master's degree in Cybersecurity 2021-22

Intrusion Detection Systems

Angelo Spognardi
spognardi@di.uniroma1.it

*Dipartimento di Informatica
Sapienza Università di Roma*



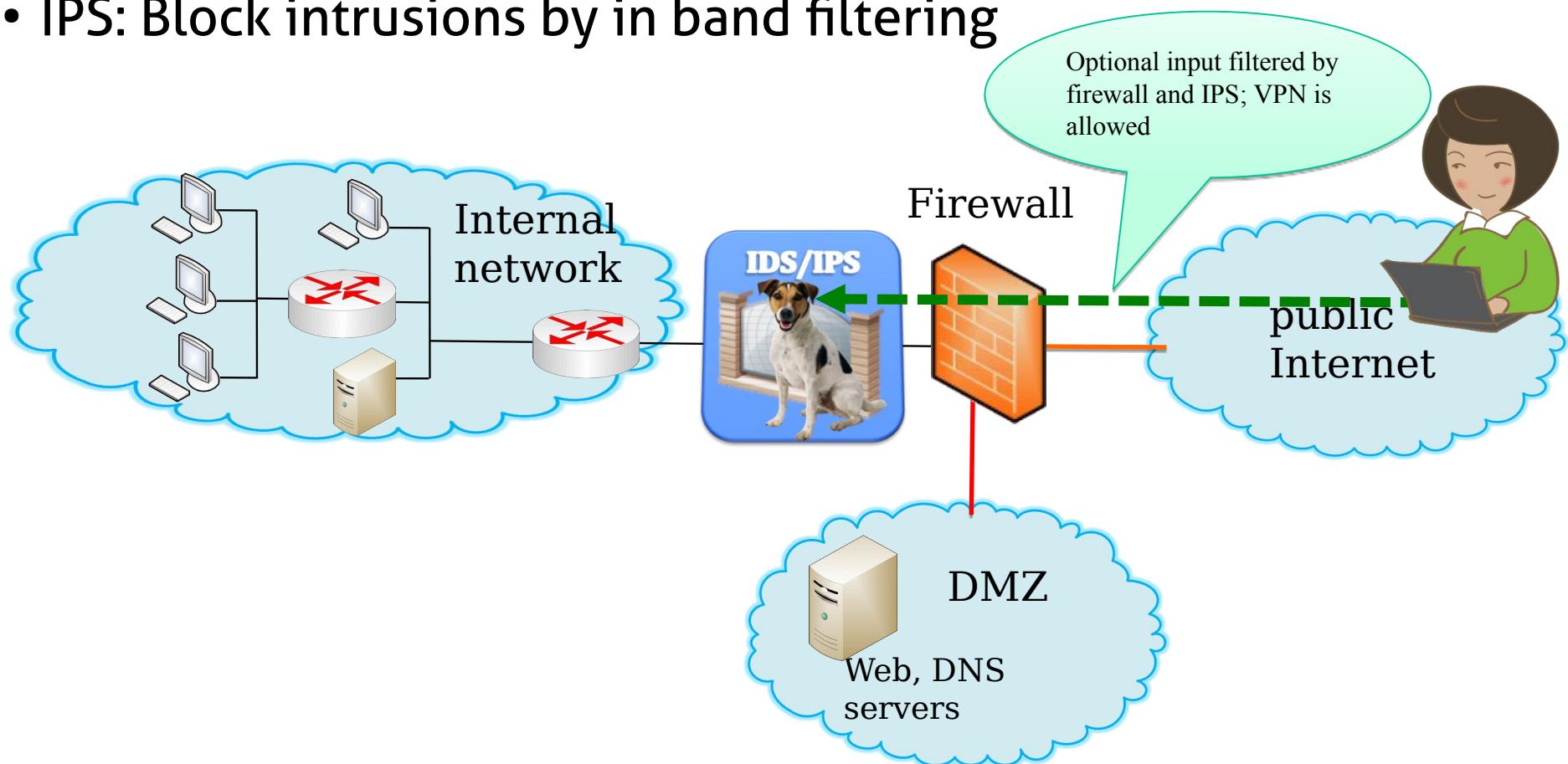
Intrusion detection/prevention systems

- An Intrusion Detection System (IDS) aims at detecting the presence of intruders before serious damage is done.
- The ultimate purpose of an intruder could be to:
 - Prevent the legitimate users from using the system
 - Reveal confidential information
 - Use the system as a stepping stone to attack other systems
- Second generation IDS are IPS, Intrusion Prevention Systems, also produce **responses** to suspicious activity, for example, by modifying firewall rules or blocking switches ports

Intrusion Detection/Prevention system (IDS/IPS)

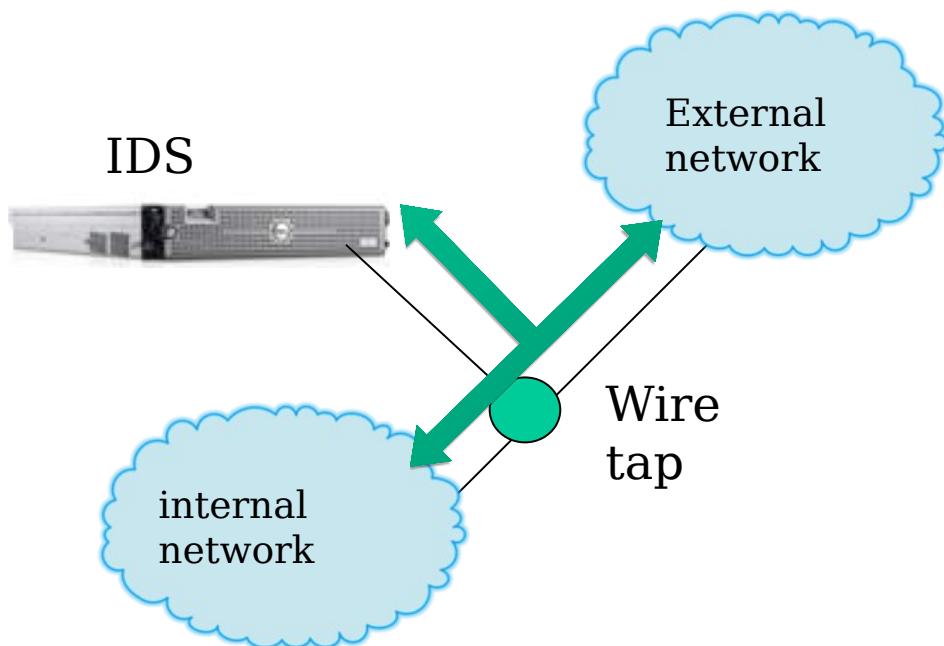


- Deep packet inspection (payload)
- IDS: report intrusions by out of band detection
- IPS: Block intrusions by in band filtering

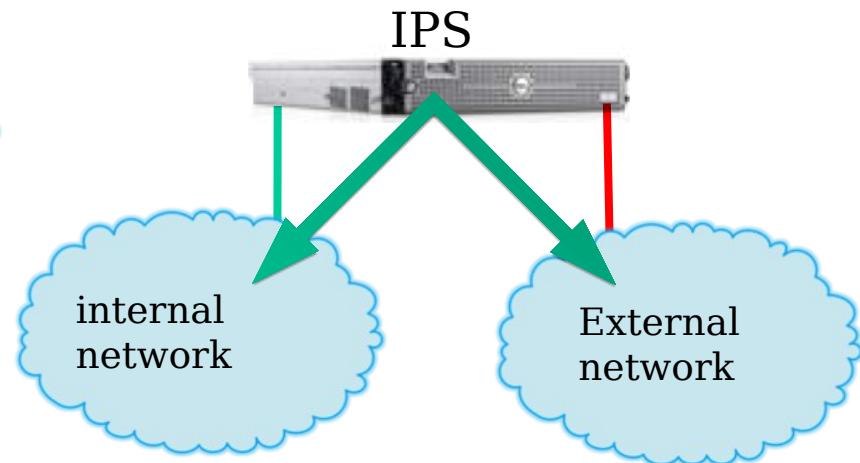


IDS vs. IPS

IDS: out of band



IPS: in line



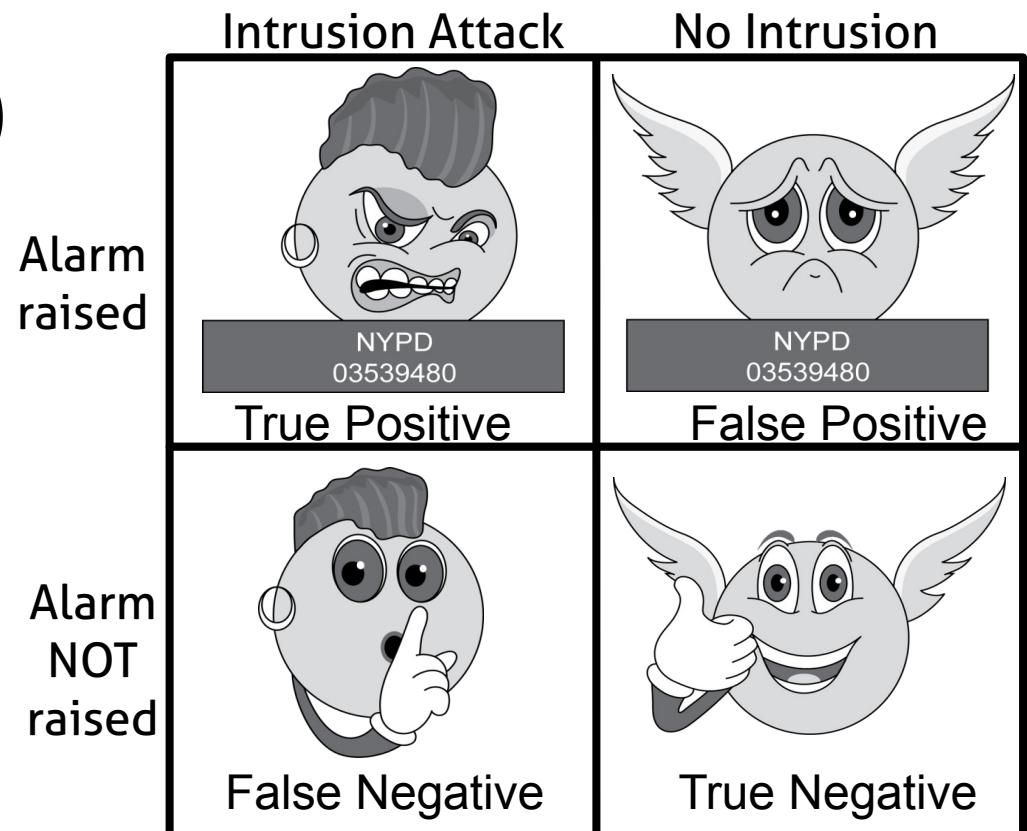


IDS and IPS

- IDS is passive: detects and raises alarms
- IPS is active: detects and reacts
- Typically IPS is placed in-line, able to actively prevent/block intrusions in real time
 - However, functionalities of both have blurred: NIST (National Institute of Standards and Technology) uses the term IDP

Beware to the alarms

- Alarms can be raised (positive) or not (negative)
- IDS needs to detect a substantial percentage of intrusions with few false alarms
- If too few intrusions detected → no security
- If too many false alarms → ignore

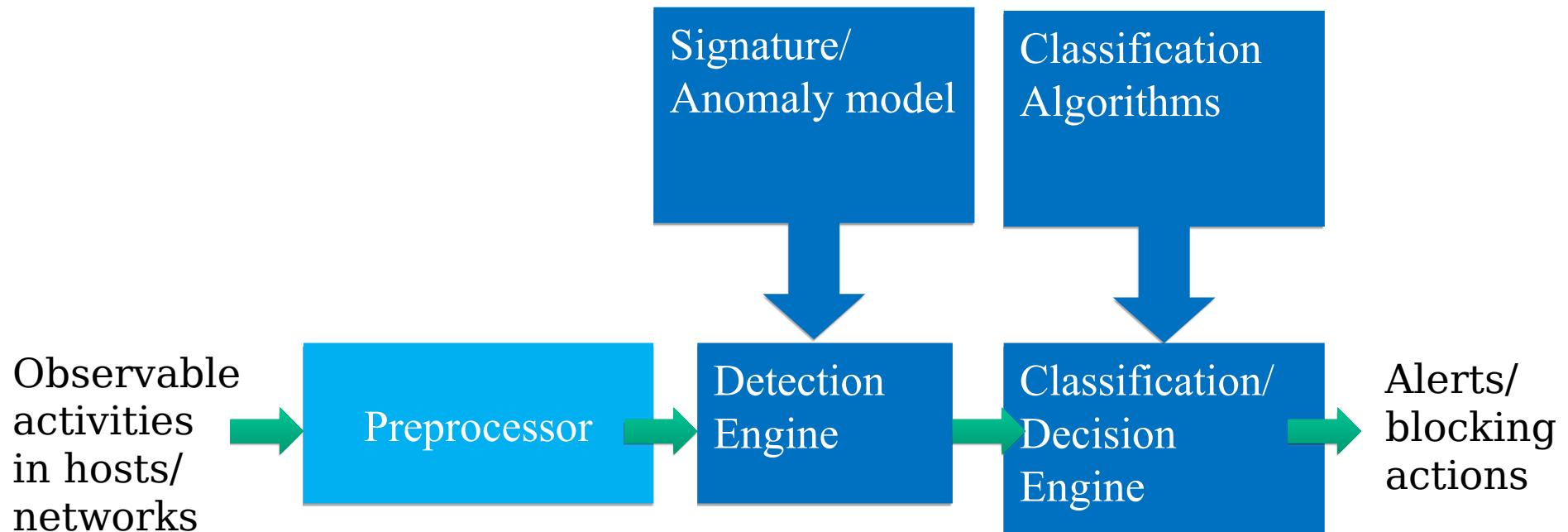




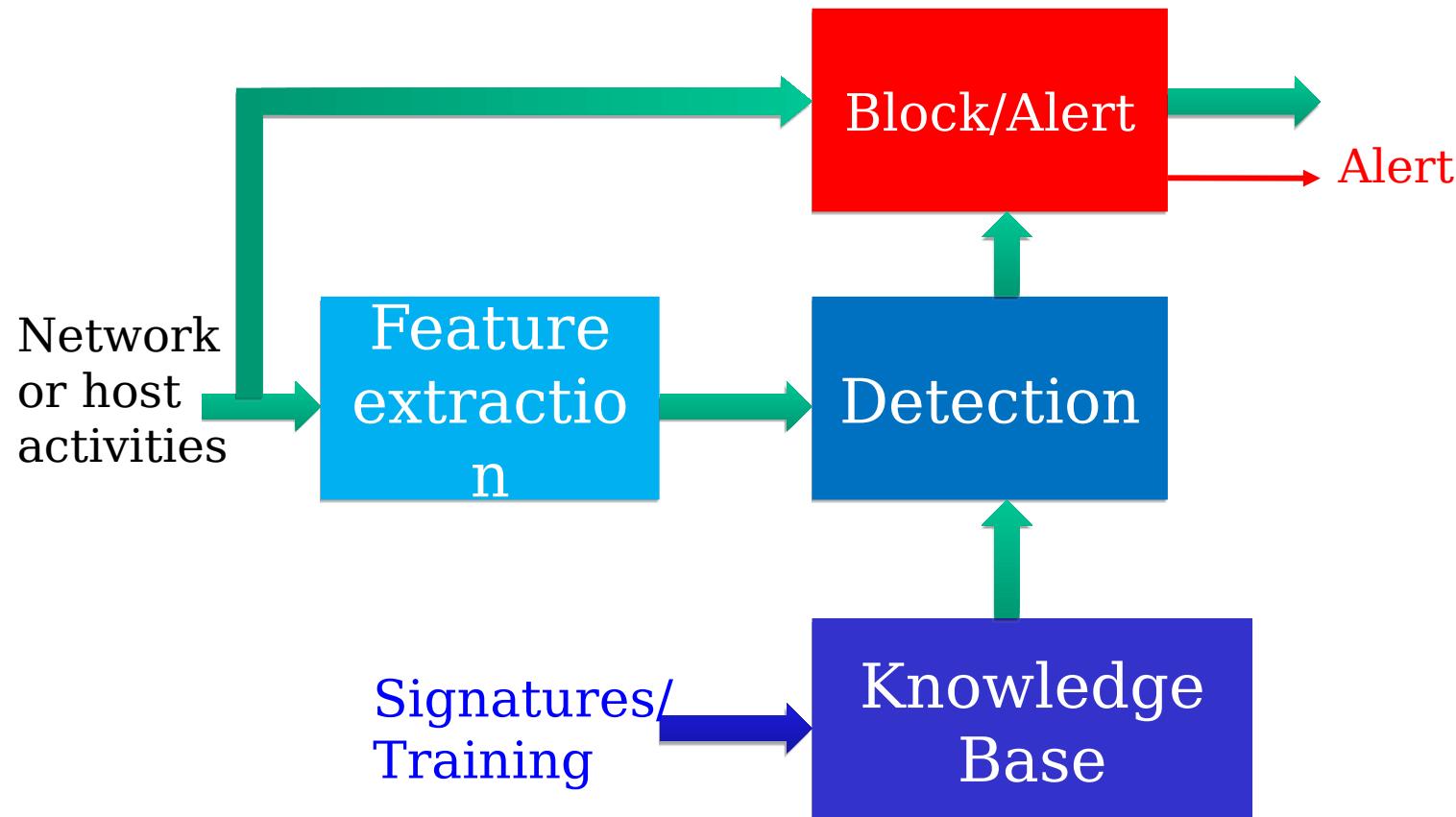
Other functionalities

- Recording information related to observed events.
- Notifying (alert) security administrators of important observed events.
- Producing reports
 - Reports summarize the monitored events or provide details on particular events of interest.
- In case of IPS also changing the network activity
 - Drop connections, block accesses, change configurations of other devices, change of the content of packets (normalization of the requests) and so on

IDS/IPS block diagram



IDS/IPS Function Blocks



Activities Monitored by IDS/IPS (1)

- Any activity sensitive to occurrences of any events deemed to be **security concerns**
- Attempted and successful breach
 - Reconnaissance
 - Patterns of specific commands in application sessions
 - e.g., a successful remote login session should contain authentication commands
 - Login and location frequency
 - Content types with different fields of application protocols
 - e.g., the password for an application must be 7-bit ASCII of 8 to 64 allowed characters to avoid buffer overflow and SQL injection
 - Network packet patterns between protected servers and the outside world
 - Client application, protocol and port, volume, and duration
 - Rate and burst length distributions for traffic
 - Privilege escalation



Activities Monitored by IDS/IPS (2)

- Attacks by legitimate users/insiders
 - Illegitimate use of root privileges
 - Unauthorized access to resources and data
 - Command and program execution
 - Mouse, keyboard, CPU, disks, I/O patterns
 - Programs/system calls/processes execution frequencies, resource access (exhaustion), denied executions
 - File/database access activity
 - Read/write/create/delete frequency; records read/written; failed reads, writes, creates, deletes; resource exhaustion
- Malware:
 - Rootkits/Trojans/Spyware
 - Viruses, zombie and worms
 - Scripts
 - Hard to handle mutations
 - Polymorphic and metamorphic viruses: each copy has a different body
- Denial of service attacks
 - Rate and burst length distributions for all types of traffic



Types of IDS



- **Host-based (HIDS):**
 - Monitors events in a single host to detect suspicious activity.

 - Typically deployed on critical hosts offering public services
 - Advantage: better visibility into behavior of individual applications running on the host
- **Network-based (NIDS):**
 - Analyses network, transport and application protocol activity
 - Often placed behind a router or firewall that is the entrance of a critical asset
 - Advantage: single NIDS/IPS can protect many hosts and detect global patterns
- **Wireless (WIDS):**
 - Analyses wireless networking protocol activity (not T- or A-layers)
 - Typically deployed in or near an organization's wireless network

HIDS

- Only monitors traffic on one specific system
 - No promiscuous mode
- It looks for unusual events or patterns that may indicate problems
 - Unauthorized access and activity
 - Unexpected activity
 - Changes in configurations
 - Software changes
 - Ex. Tripwire





NIDS



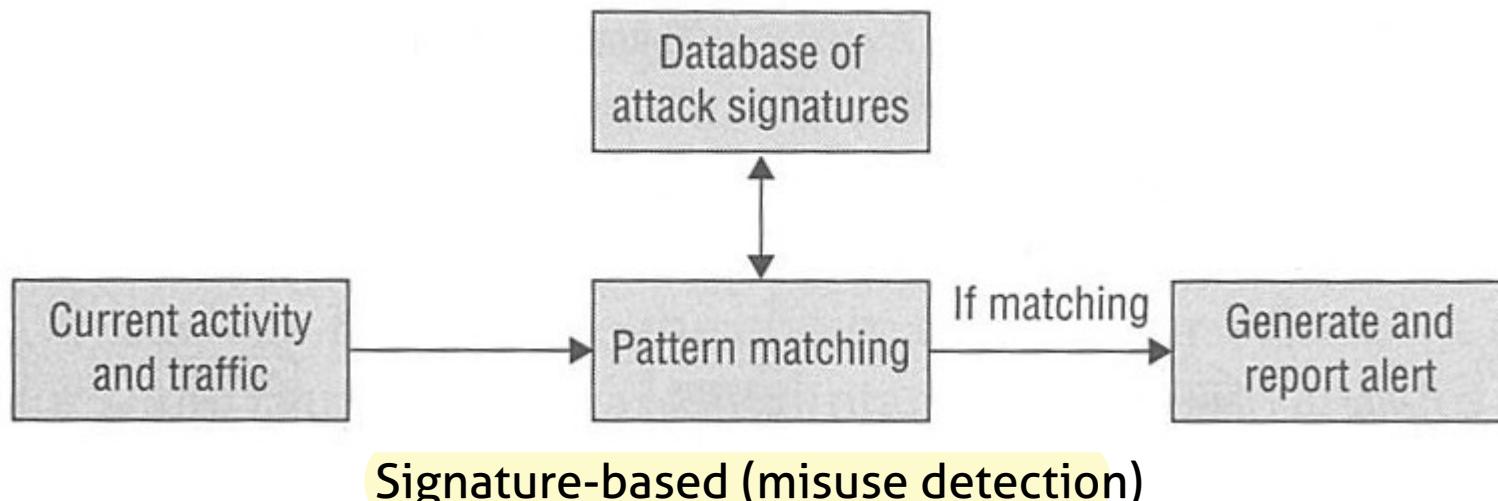
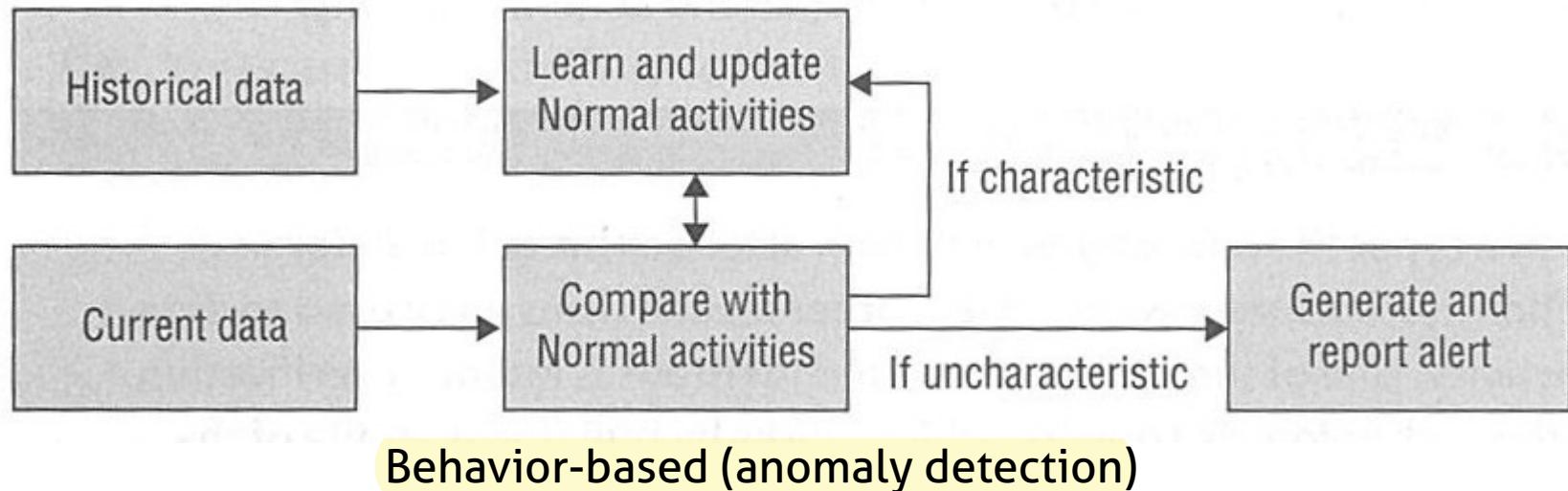
- Usually operates in promiscuous mode → sniffers
 - Can have multiple NICs to monitor multiple network segments
- Usually connected to switches with ports mirrored (or in SPAN mode, Switch Port ANalizer):
 - All the traffic generated within ALL the ports of the switches are replicated on the mirrored port where the NIDS is placed
- Often it has a series of sensors placed in different networks (DMZ, internal net, specific nodes...)
 - Distributed detection system



Other types of IDS

- **File integrity monitors** (e.g. Tripwire, AFICK)
 - Monitor changes to key system configuration files
- **Flow-based IDS** (NetFlow)
 - Tracks network connections
 - Establishes patterns of normal traffic
 - Alert when unusual services/patterns/protocols/behaviors seen
 - Can give a good overall situational view on large network(s)
- **Hybrid detection capabilities**
 - Augment or replace signature-based detection
 - Usually anomaly/behavior-based (pseudo-artificial intelligence)
 - Often require “training” periods to establish a baseline

IDS approaches





How to recognize an intrusion



- Behavior-based (anomaly detection):
 - Define behavioral characteristics of normal behavior
 - Compare actual behavior with these. If there are significant differences, **raise an alarm**
 - Difficult to define all possible normal behavior. New activities often give “false positives” (i.e. normal behavior classified as intrusion)
- Signature based (misuse detection):
 - Define characteristics of various types of abnormal activities
 - Compare actual behavior with these. If any of them match, **raise alarm!**
 - Difficult to produce a complete catalog of abnormal activities.
 - If any are missing, there will be “false negatives” (i.e. undetected intrusions)



Learn and classify anomalies

- Behavior is typically described in terms of a set of features
- The feature set should describe all relevant aspects of the behavior to be recognized
- Anomaly detection requires some form of learning (or “training”), usually based on **data mining** in actual observations
- A too large feature set means that both training and classification will take unnecessarily long time
 - Require more observations in order to deal with more features



Example: Lee & Stolfo feature set

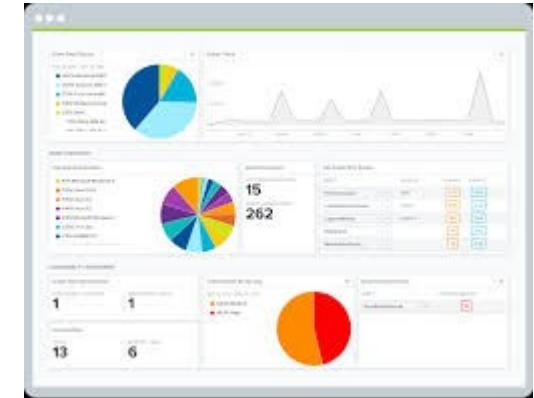
- A classic set of features for NIDS anomaly detection
- Constructed by data mining on identified attack patterns in network traffic (collected via tcpdump)
- Features fall into three classes:
 - Intrinsic features: Data about a particular connection or network flow
 - e.g. connection lifetime, amount of data, illegal fragments
 - Traffic features: Statistical information about connections
 - e.g. % of connections to same host with "SYN" errors
 - Content features: Application-related statistics
 - e.g. No. of file creations, failed login attempts.
- Rather out of date by now ([published in 2000](#)), but still useful inspiration



Other feature sets

Particular technologies need their own feature sets, e.g.:

- Wireless networks:
 - Signal power
 - Sequence number “jumps”
 - Round-trip time (RTT)
- Grid/cloud systems:
 - GridFTP connections
 - GridFTP mode of operation
 - Number of GridFTP clients (evidence of “Flash crowds”)
 - Traffic entropy
 - Type of LDAP operations





Behavior-based IDS

- Intruders may behave in a different manner from ordinary users or ordinary programs:
 - Many types of attack are characterized by abnormal patterns of OS use or network use.
- Recognizing abnormal behavior enables us to detect attacks as they take place.
- Big questions:
 - How to recognize normal and abnormal behavior patterns?
 - How quickly can recognition take place?
 - How do we deal with abnormally behaving systems?

Behavioral w.r.t. anomalies

- Take sequence of observed behavioral elements (system calls, network packets or others)
- Derive feature values from the behavioral elements
- Derive the “normal” behavior, generated using statistics or with a set of rules (like parameters or procedures) or with a Machine Learning approach
- Compare the traffic against the normal behavior:
 - 1 Distance measures (statistics and thresholds):
 - Hamming distance: How many elements have to be changed?
 - Mahalanobis distance: Generalized distance in n dimensions.
 - Kolmogorov complexity: Difference in information density?
 - 2 Probability measures (“how likely is this sequence?”), e.g.:
 - Markov model
 - Neural Networks
 - Any other ML mechanism
 - 3 Rule sets (“does the sequence follow a set of pre-defined rules?”)





Adaptive and self-learning profile

- Adaptive profiles can account for normal network changes to avoid raising false alarms
- Self-learning is critical to ensure wide and successful deployment of anomaly-based detection mechanisms
 - Manually set the profiles is difficult because of the complexity of dynamically changing traffic statistics
- Need to apply anomaly-based detection at various levels of traffic aggregation to achieve the most accurate protection
 - A single server, a server farm, a business division, an enterprise



Signature-based IDS

- Starts from the idea that intruders/attacks may have a characteristic appearance which makes it possible to identify them
- The idea is to screen the PAYLOADS of the packets looking for specific patterns → signatures
- Suppliers of IDSs maintain huge databases of signatures (code or data fragments) which characterize various classes of intruder.
- Rapid recognition involves searching for matches for one or more of the known signatures from a collection of many thousands of signatures



Signature-based (Rule-based) IDS

- Dominant technology in commercial systems
- Rules express actions on given conditions, possibly with complex predicates, including timing, payload content etc etc.

Act.	proto	src...	dest...	options...
alert	tcp	any	any	-> 10.1.1.0/24 80 (content: "/cgi-bin/phf";)
alert	tcp	any	any	-> 10.1.1.0/24 6000:6010 (msg:"X traffic";)
alert	tcp	any	any	-> any 21 (msg:"FTP ROOT";content:"USER root";)

- Typically supplied ready-made by manufacturer of IDS
- Requires considerable effort by manufacturer to find right rules and distribute them to subscribers as new attack forms are analyses
- A job for real experts! (But easy for the user...)
- Rules can in fact be derived from more automatic IDS technologies (Markov, ANN, clustering, etc.)! This is an active area for research.



Signature-based IDS principles

- A packet sniffer “on steroids”
- Captures the packets in a LAN and applies some fairly complex logic to decide whether an intrusion has taken place
 - SNORT is one of the best known intrusion detectors
 - Easy-to-learn and easy-to-use rule language for intrusion detection.
 - The rules are stored in `/etc/snort/rules` directory
- **Con:** can not inspect encrypted traffic (VPNs, SSL)
- **Con:** not all attacks arrive from the network
- **Con:** record and process huge amount of traffic



IDS detection capability

- Decode packets, namely DPI: deep packet inspection
- Decode application and protocol headers to look at high-layer activity → the payload containing the application protocol
- Protocol decoding to detect anomalies



Misuse detection

- Set of rules defining a behavioral signature likely to be associated with attack of a certain type
- Example: buffer overflow
 - A setuid program spawns a shell with certain arguments
 - A network packet has lots of NOPs in it
 - A very long argument to a string function
- Example: SYN flooding (denial of service)
 - Large number of SYN packets without ACKs coming back
 - ...or is this simply a poor network connection?
- Attack signatures are usually very specific and may miss variants of known attacks
 - Why not make signatures more general?



Extract misuse signatures

- Use invariant characteristics of known attacks
 - Bodies of known viruses and worms, port numbers of applications with known buffer overflows, RET addresses of stack overflow exploits
 - Hard to handle malware mutations
 - Metamorphic viruses: each copy has a different body
- Challenge: fast, automatic extraction of signatures of new attacks
- Honeypots are useful for signature extraction
 - Try to attract malicious activity, be an early target



Honeypot



- Definition:
 - A security resource whose value lies in it being attacked, probed or compromised.
- A **honeypot** is (usually) a single computer, whereas
- A **honeynet** is a network of computers, usually protected by a firewall to regulate traffic.
- The idea is to attract the attackers



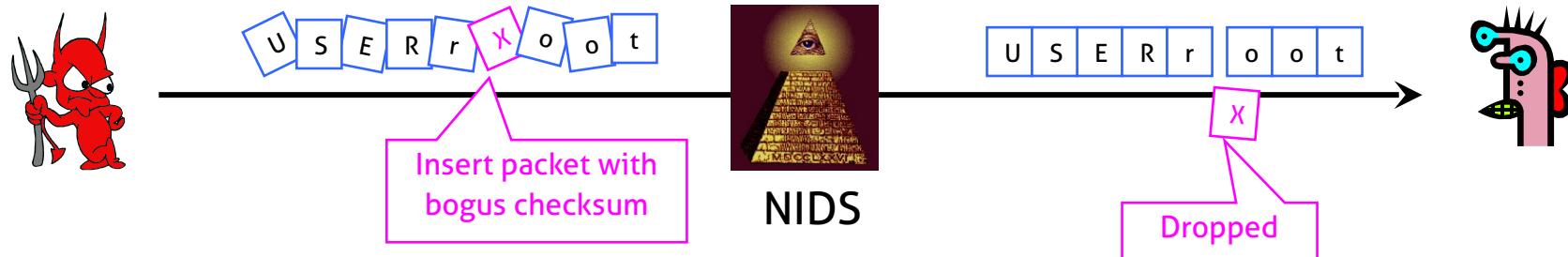
Example of IDS rule evasion

- Want to detect “USER root” in packet stream
- Scanning for it in every packet is not enough
 - Attacker can split attack string into several packets; this will defeat stateless NIDS
- Recording previous packet’s text is not enough
 - Attacker can send packets out of order
- Full reassembly of TCP state is not enough
 - Attacker can use TCP tricks so that certain packets are seen by NIDS but dropped by the receiving application
 - Manipulate checksums, TTL (time-to-live), fragmentation

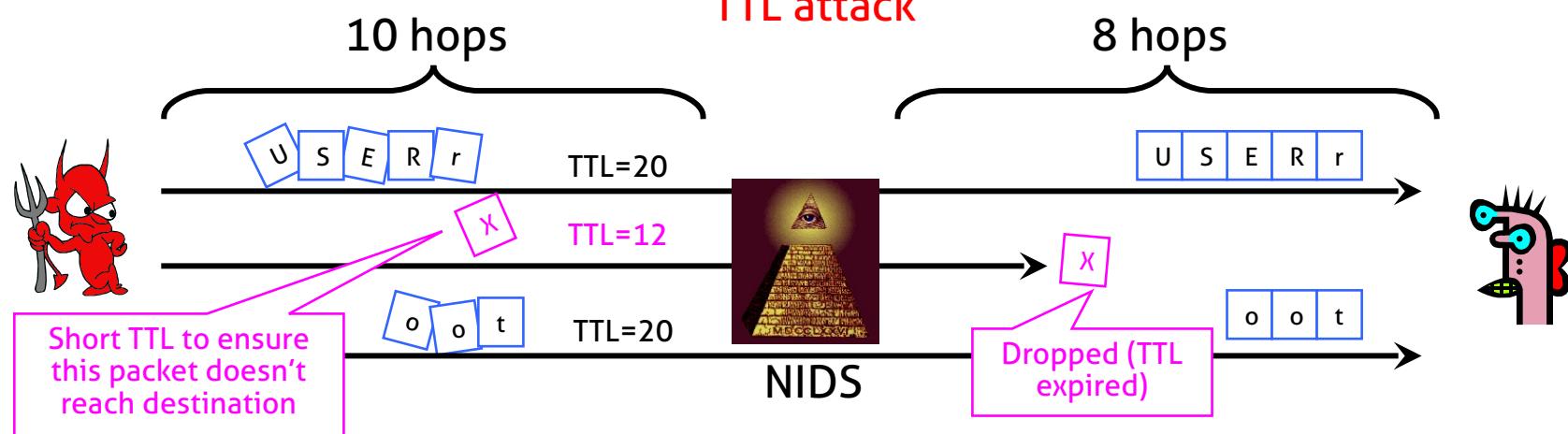


TCP Attacks on NIDS

Insertion attack



TTL attack





Behavior-based detection techniques: protocol anomaly

- Layers 2-7 inspection by rules
- A protocol or service for a non-standard purpose or on a non-standard port
 - e.g., modified protocols for tunneling through firewalls (e.g., P2P on port 80)
 - Port scan
- Full IP defragmentation, and TCP reassembly
- Deep packet inspection
 - IP fragmentation overlaps and suspicious IP options
 - Unusual TCP segmentation overlaps and illegal TCP options and usage
 - Deep application protocol parsing/decoding
 - Illegal field values and combinations
 - Illegal command usage
 - Unusually long or short field lengths, which can indicate buffer overflow
 - Very long argument to a string function
 - Unusual number of occurrences of particular fields/commands
 - Application semantics
 - Type of encoding is legal for a given field
 - Applications can be embedded within it
 - Application level anomaly: shellcode in unexpected fields



Behavior-based detection techniques: statistical anomaly

- Statistical measures are used to capture network traffic behavior
 - A stable balance among different types of TCP packets in the absence of attacks
 - 3-way handshake, 4-way close, and data transfer
- This balance can be learned and compared against short-term observations that will be affected by attack events
 - Profiles based on statistical measures of time-of-the-day, day-of-the-week variations in traffic volume
 - Profiles for traffic rate distributions on a multi-week scale normal network environments
 - Profiles based on statistical measures could raise DDoS anomalies based on rare events of the difference between the long- and short-term distributions or based on a rare occurrence of long bursts of high-rate traffic
- Sensitivity level: can be adjusted to different levels of profile deviation
 - e.g., low sensitivity: high traffic profile deviation can trigger a DDoS alarm



Signature vs. Behavior

- Signature-based detection
 - Clearly indicates the detected attack method
- Behavioral-based alerts
 - Indicate:
 - The attack type
 - The behavioral rule that was violated, such as port scan
 - The statistical profile that was violated
- Behavioral protection can not identify the specific attack or exploit that was blocked
 - Require security administrator to investigate clues given by the behavioral rule to determine which attack was actually attempted
 - Acceptable for new and unknown attacks
 - But established threats and known exploits should be easily identifiable
- Deciphering information about each attack reported by a behavioral-only system becomes unmanageable for a large number of hosts



Combine behavior and signature

- False negatives: attack is not detected
 - Problem in signature-based misuse detection
- False positives: harmless behavior is classified as an attack
 - Problem in statistical anomaly detection
- Signature “can not” detect:
 - DoS/DDoS
 - Zero-day exploits
 - Protocol/application anomaly
- Best solution is to combine both signatures and behavioral rules
 - No false positives: Do not ever block legitimate traffic under any circumstances
 - No false negatives: Do not miss attacks, even when the attacker intentionally tries to evade detection



Combined Behavior and Signature Detection

- Detection Correlation:
 - Signature, Anomaly, and Denial of Service detection functionality
 - Interdependence and cross-checking of suspicious traffic
 - Behavioral protection can block zero-day attacks without updates to the system
- Once an exploit has been recognized using behavior-based technique, a stateful signature can be created to provide accurate detection and save manpower
 - Lower the false positive rates
 - Reduce the response time to attacks
- Most anti-virus, anti-spyware, firewall products are integrated with both Behavior and Signature intrusion detection



IDS vs. IPS

- IDS: out of band
 - An IDS false positive is an alert that did not result in an intrusion
 - The system under attack was not vulnerable to the attack
 - The detection mechanism may be faulty
 - IDS detected an anomaly that turned out to be benign
- An IDS false positive causes a security analyst to expend unnecessary effort
 - Minimize false positives
- No interference with traffic
- IPS: in band
 - When an IPS has a false positive, legitimate traffic will be blocked
 - IPS cannot have false positives
 - Better development for filters and thorough tests
- To match the line speed, IPS hardware requirement is higher
 - ASIC or FPGA



Observations

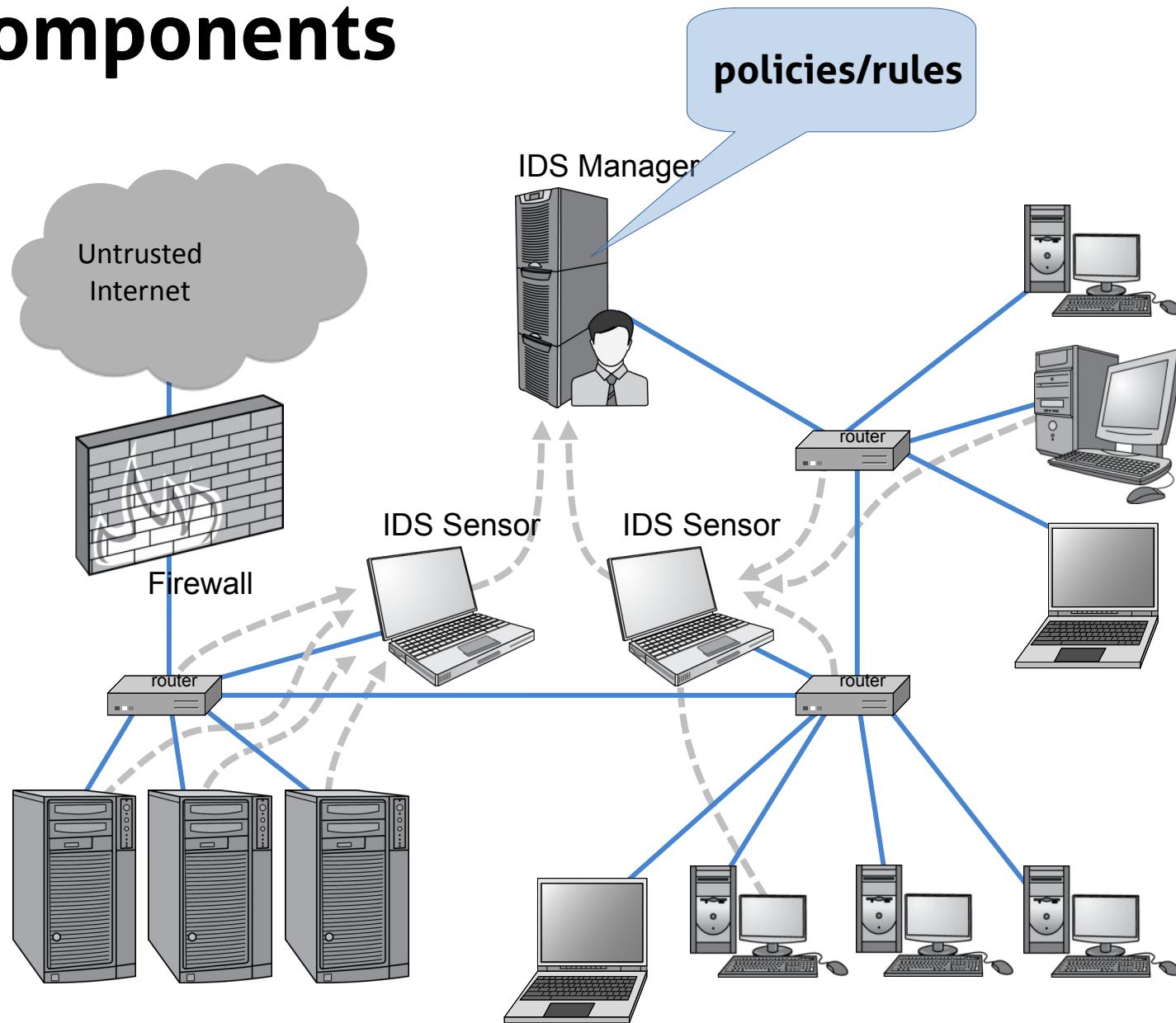
- Legitimate traffic in real networks contains anomalies
 - Protocol anomalies come from custom applications that use off-the-shelf protocol libraries, but use them in unexpected ways
 - Behavioral anomalies come from exceptional, but often critical, business processes
- IDS filters create leads on suspicious activity intended for an expert to follow
- IPS filters are used for automatic action such as blocking traffic or quarantining an endpoint
- Anomaly-based detection mechanisms (both protocol and statistical) are useful for IDS, but inappropriate for IPS
 - Anomaly filters can not be used for blocking, only for alerting



IDS architecture

- Usually several parts:
 - Network sensors: detect and send data to the system
 - Central monitoring system: a server that processes and analyzes data sent from sensors
 - Database and storage component: repository for event information

IDS Components





That's all for today

- **Questions?**
- See you next lecture!
- **References:**
 - [NIST Guide to Intrusion Detection and Prevention Systems \(IDPS\)](#)
 - Chapter 19 textbook
- **Other interesting readings:**
 - [A Framework for Constructing Features and Models for Intrusion Detection Systems](#)
 - [Specification-based anomaly detection: a new approach for detecting network intrusions](#)

Practical Network Defense

Master's degree in Cybersecurity 2021-22

Intrusion Detection Systems: Snort/Suricata, fail2ban

Angelo Spognardi
spognardi@di.uniroma1.it

*Dipartimento di Informatica
Sapienza Università di Roma*



Host IDS/IPS

- Many host security products have integrated HIPS, anti-malware and firewall
 - Protects mobile hosts from attack when attached outside the protected network
 - Protects against local attacks from a user, and codes/scripts from removable devices
 - Protects against attacks from the same subnet/VLAN
 - Protects against encrypted attacks where the encrypted data stream terminates at the host being protected
 - Inspect packet content after decrypting received VPN or SSL packets
 - Inspect packet together with anti-malware software by decrypting or emulating malware
- Con: if an attacker takes over a host, then one can tamper with IDS/agent binaries and modify audit logs
- Con: only local view of the attack
- Con: Host-based anomaly detection has high false alarm rate



Network IDS/IPS

- Deploying sensors at strategic locations with a central monitor
 - Inspecting network traffic
 - Watch for violations of protocols and unusual connection patterns
 - Protect network equipment, such as printers that do not have HIDS
 - Protect against network-oriented attacks
 - DDoS, bandwidth consumption
 - Independent of host OS
 - Monitoring user activities
 - Look into the data portions of the packets for malicious command sequences
- Con: may not detect encrypted traffic
 - Data portions and some header information can be encrypted
- Con: can not detect some attacks in the host
- Con: high requirement for computation capability of IDS/IPS



Then: combine host and network IDS/IPS

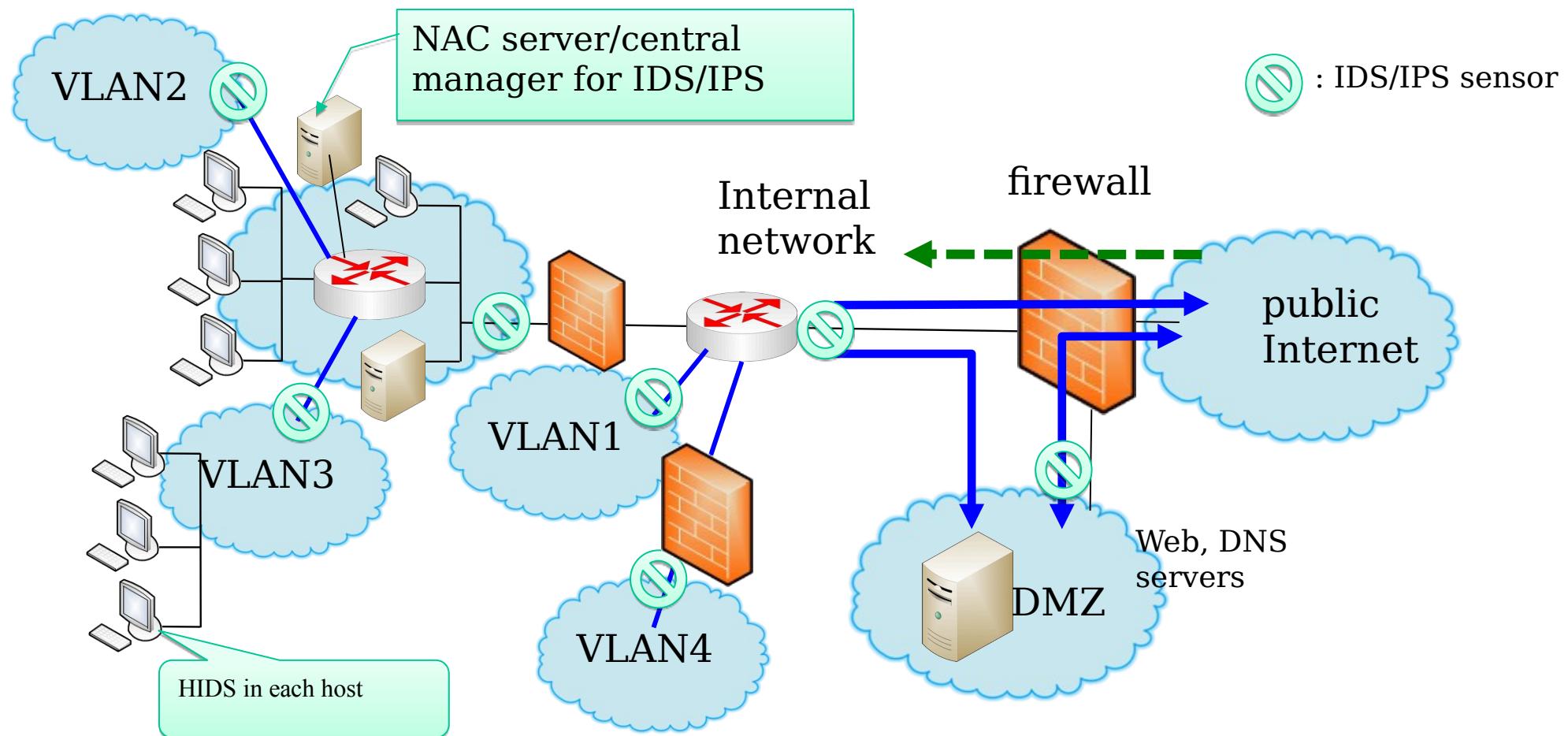
- Both HIDS and NIDS technologies are not equally adept at detecting and blocking certain attacks
- There are attacks that can only be detected by HIDS
 - E.g., local privilege escalation, metamorphic malware
- Attacks that can only be detected by NIDS
 - E.g., Routing advertisement injection
- Integrating the strengths of both architectures provides a solution whose sum is greater than its parts
- More accurate result for quarantining a host or block/filter traffic
- The basis for NAC (Network Access Control) products



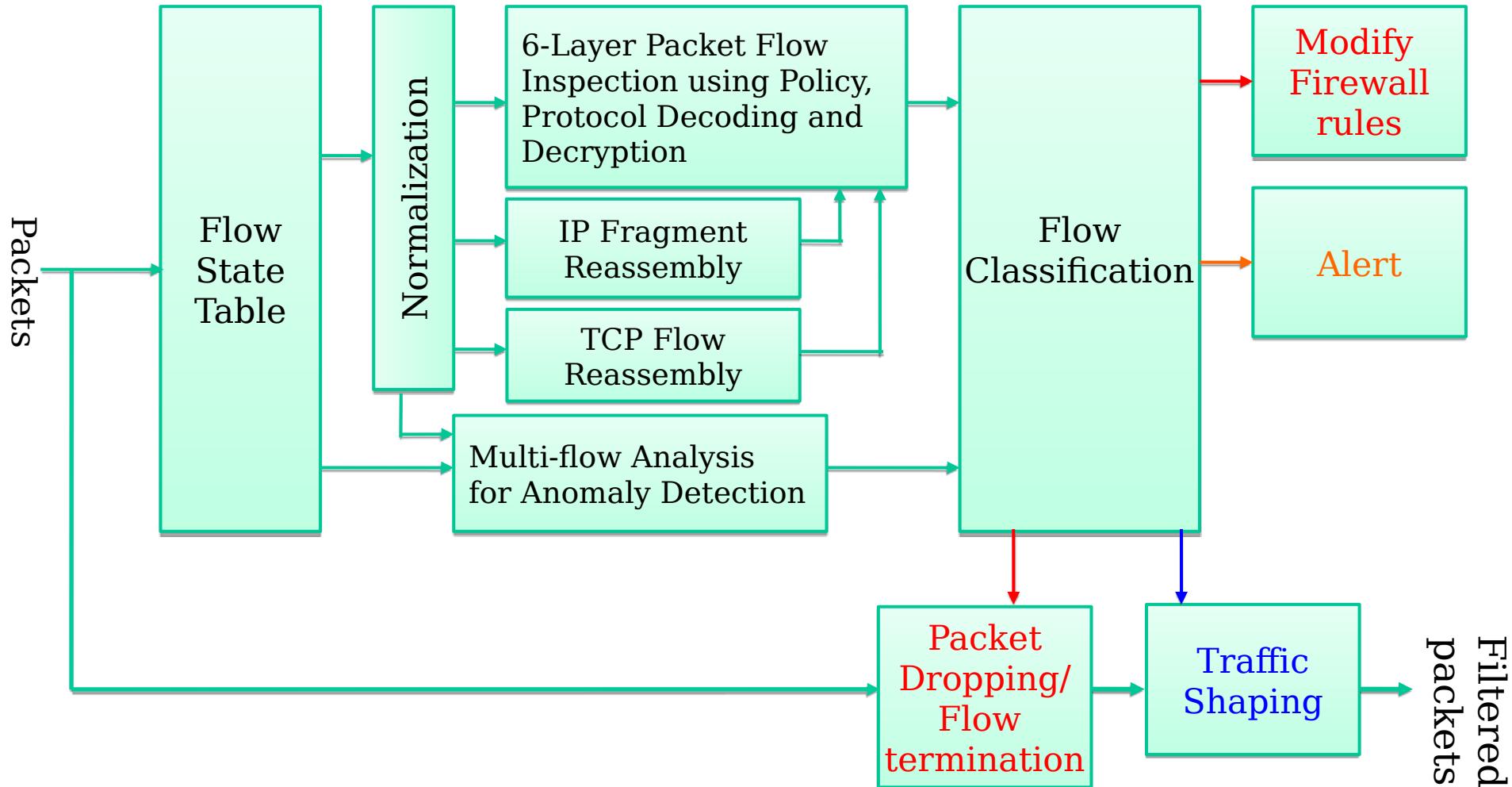
Distributed IDS

- Extend focus from single systems to information infrastructure
 - More effective defense has these working together to detect intrusions
 - Agent-based coordination between host and NAC server
- Monitoring and correlating public, internal VLANs, and DMZ segments of the IDS/IPS sensors and firewalls
- Correlation among these segments to yield an accurate picture of network attacks that were either blocked or made it into the internal network
- Correlate HIDS and NIDS for constant monitoring and blocking in NAC (central control)
- Exchange Format Working Group (IDWG) of the IETF
- Intrusion Detection Exchange Protocol (IDXP): RFC 4767
 - An application-level protocol for exchanging data between IDS's
 - IDXP supports mutual-authentication, integrity, and confidentiality over a connection-oriented protocol
 - The protocol provides for the exchange of the Intrusion Detection Message Exchange Format (IDMEF) messages in implementations of the data model in the Extensible Markup Language (XML)
 - The IDMEF message elements are described in RFC 4765, and developed by the Intrusion Detection

Distributed intrusion detection



Network-based IPS block diagram





State information and analysis

- State information for a session (flow):
 - Maintaining state information enables sensors to gain context for attack detection
 - Inspecting the entire content of the data packet
- State information is captured and updated in real time
- State information is the basis for Layer 2-7 detection
 - Utilize multiple token matches to capture attack signatures/behaviors that span packet boundaries or are out-of-order in a packet stream
 - Detect/block malware, Trojans, key loggers, P2P, botnets, worms
- Appropriate use of the state information is the key to detection
 - Accuracy depends on the selection of parameters and their transitions



Normalization

TCP normalization

- Inspect invalid or suspect conditions
 - E.g., a SYN sent to the client from the server or a SYNACK sent to the server from the client
- Block certain types of network attacks
 - E.g., insertion attacks and evasion attacks
 - Insertion attacks occur when the inspection module accepts a packet that the end system rejects
 - Evasion attacks occur when the inspection module rejects a packet while the end system accepts it
- Discards segments containing
 - Bad segment checksum
 - Bad TCP header or payload length
 - Suspect TCP flags (for example, NULL, SYN/FIN, or FIN/URG)
- To configure TCP normalization
 - Assemble various TCP commands into a parameter map for filtering as policy
 - E.g., parameter map contains ranges for MSS, # of SYN retries, # of out of order segments, control of timeout, random sequence number, Window scale factor, urgent flag, etc

IP normalization

- Inspect IP packets using configured parameter map for:
 - General security checks
 - ICMP security checks
 - Fragmentation security checks
 - IP fragment reassembly
 - IP fragmentation if a packet exceeds the outbound maximum transmission unit (MTU)
- Configure IP normalization parameter map
 - ToS
 - TTL
 - Unicast reverse path
 - Fragment reassembly
 - Maximum # of fragments
 - MTU



Actions from IPS

- Packet dropping
- Session termination
- Firewall rules modification for blocking suspicious hosts
- Traffic shaping for slowing down less critical traffic such as P2P, video
- Alerts generation
- Log generation



Snort and Suricata



Snort: open source NIDS/NIPS

- Written by Martin Roesch
 - Now developed by Sourcefire, of which Roesch is the founder and CTO
- The most widely deployed intrusion detection and prevention technology worldwide
- Using a rule-driven language
- Combining the benefits of signature, protocol and anomaly based inspection methods.
 - Snort can be combined with other software such as SnortSnarf, sguil, OSSIM, and the Basic Analysis and Security Engine (BASE) to provide a visual console
 - Emerging Threats: community maintained Snort rule sets are evolving
- Large rule sets for known vulnerabilities

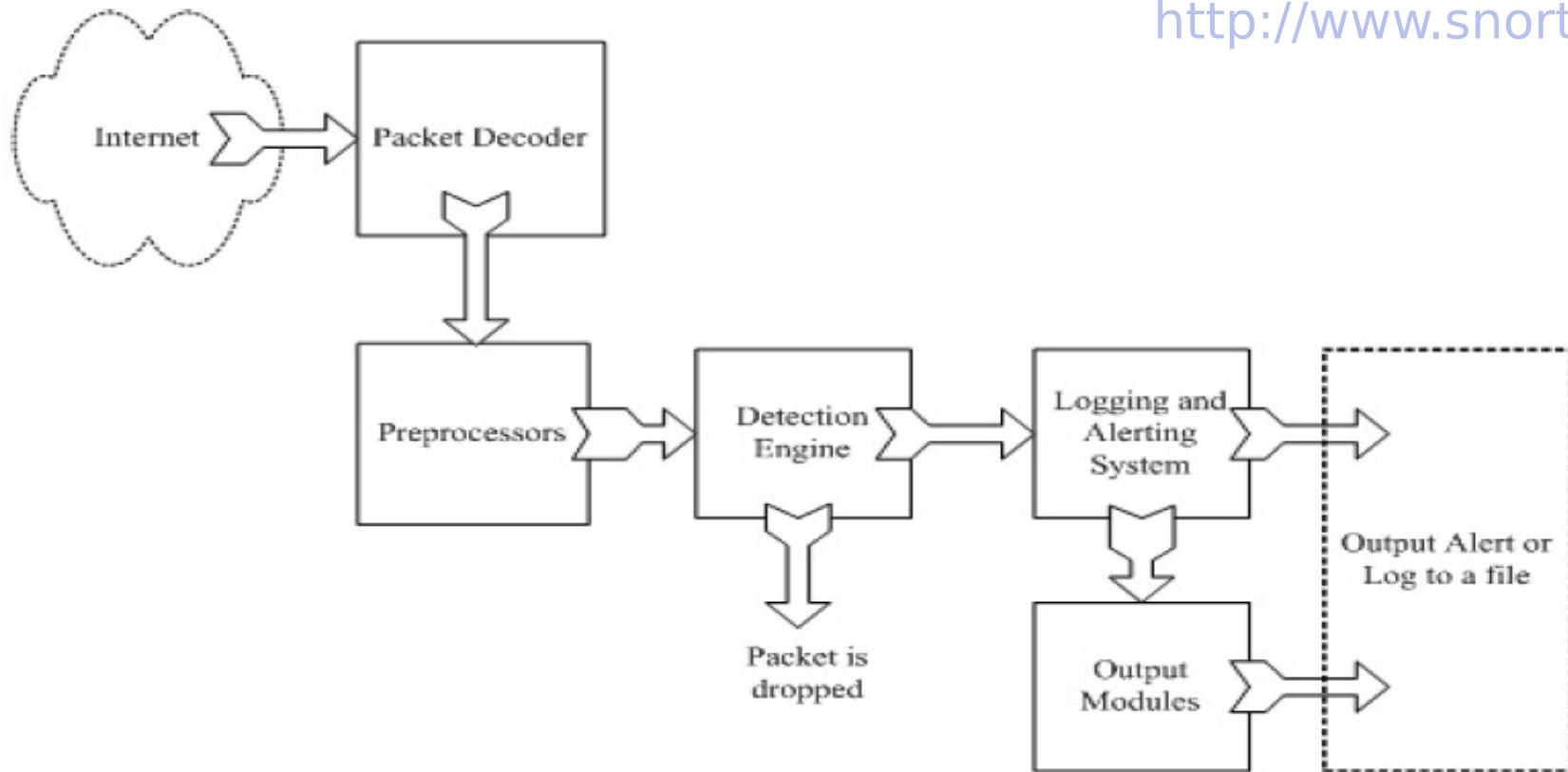




Snort



<http://www.snort.org/>



From: Rafeeq Ur Rehman, *Intrusion Detection Systems with Snort: Advanced IDS Techniques with Snort, Apache, MySQL, PHP, and ACID.*

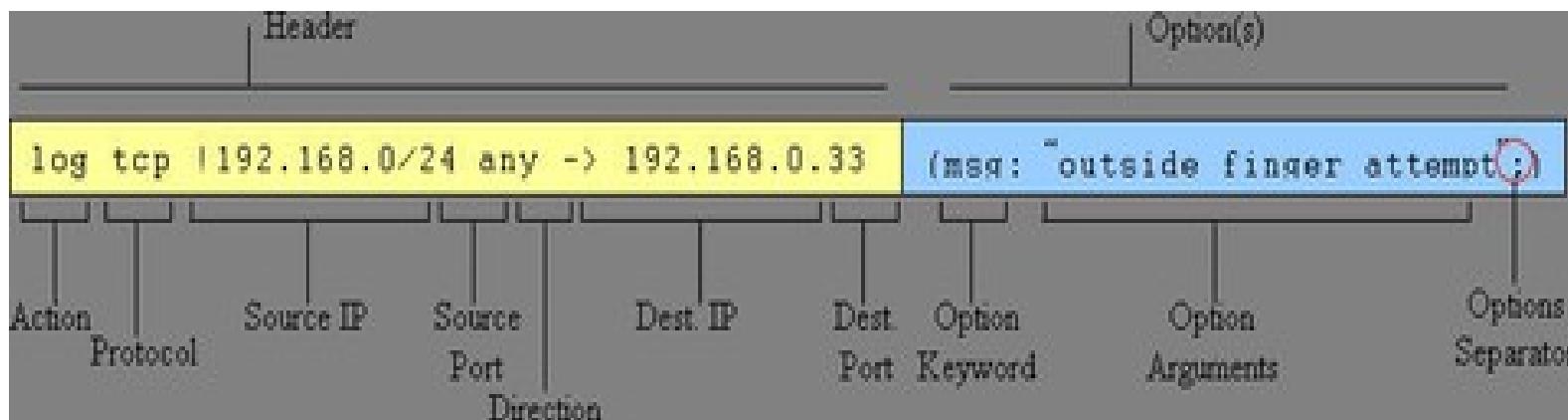
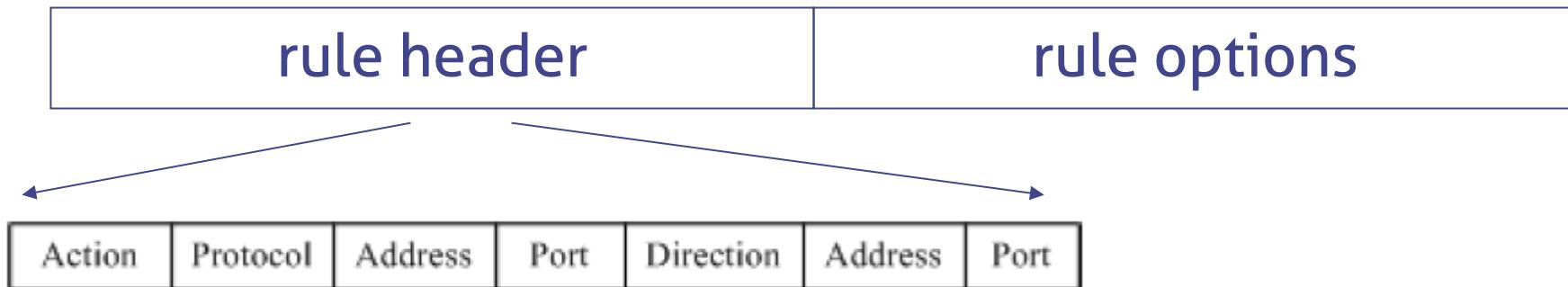


Snort components

- Packet Decoder
 - input from Ethernet, SLIP, PPP...
- Preprocessors
 - detect anomalies in packet headers
 - packet defragmentation
 - decode HTTP URI
 - reassemble TCP streams
- Detection Engine: applies rules to packets
- Logging and Alerting System
- Output Modules: alerts, log, other output

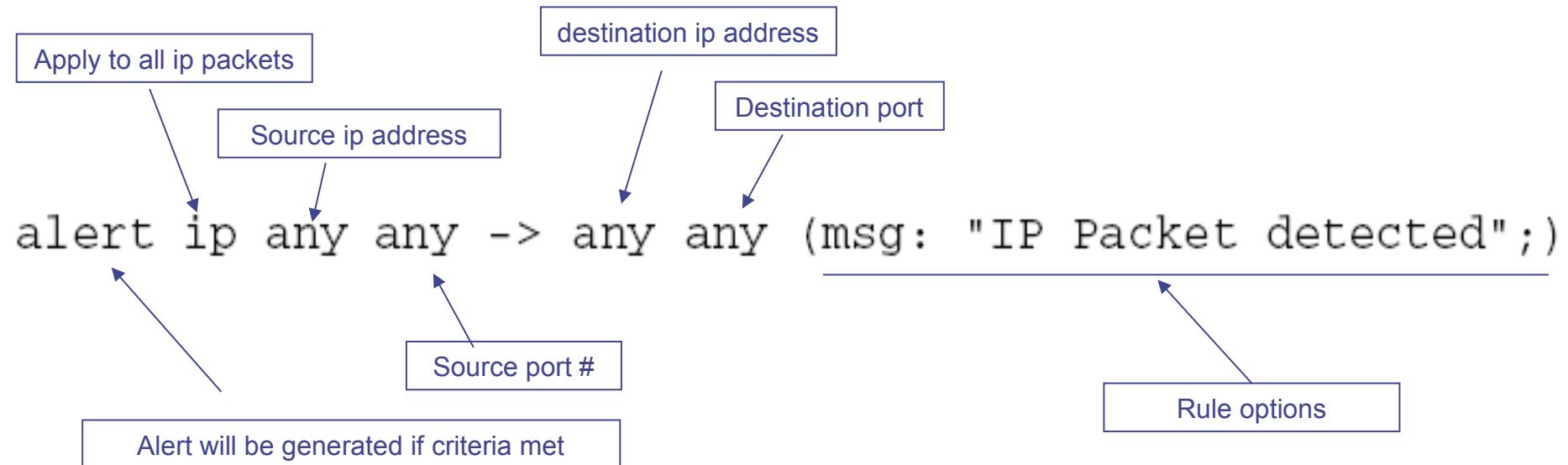


Snort detection rules





Example



```
alert tcp $TELNET_SERVERS 23 -> $EXTERNAL_NET any (msg:"TELNET  
Attempted SU from wrong group"; flow:  
from_server,established; content:"to su root"; nocase;  
classtype:attempted-admin; sid:715; rev:6;)
```



TCP/IP header rule options

ipopts: opt	Match specific IP option opt
flags: fff	Match settings of one or more TCP flags
seq: nnn	Match specific TCP sequence number
ack: nnn	Match specific TCP ack number
window: nnn	Match specific TCP window size
itype: nnn	Match ICMP type field value (or range)
icode: nnn	Match ICMP code field value (or range)
fragbits: mmm	Match IP fragmentation/reserved header bits
ip_proto: proto	Match IP Protocol field by number or name
id: nnn	Match value of IP ID header field
ttl: nnn	Match value of IP TTL header field
dsize: nnn	Match packet payload size (or size range)
flow: flowstat	Match flow direction/state
rpc: app,ver,proc	Match RPC application, version and procedure



Payload checking rule options

content: "xxx"	Match pattern "xxx" in packet payload
offset: nnn	Offset for start of search for content match
depth: nnn	Number of bytes to search for content match
distance: nnn	Offset for search relative to end of last match
within: nnn	No. of bytes to search rel. to end of last match
nocase	Ignore case when looking for matches
isdataat: nnn	Checks that data are present in byte onnn, possibly relative to previous match
pcre: "/regex/mmm"	Match pattern given by Perl regular expression regex with modifiers mmm
uricontent: "sss"	Match a onormalised URI, i.e. where hex codes, directory traversals etc. have been rationalised
byte_jump: rules	Gives rules for matching TLV-encoded protocols



Snort challenges

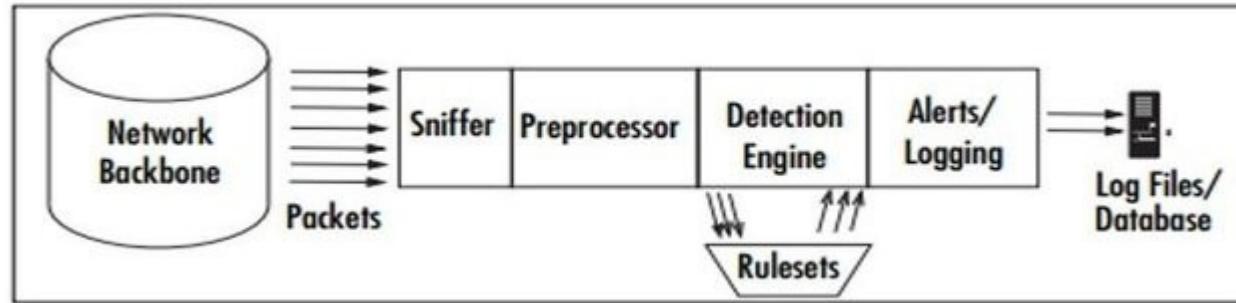
- Misuse detection – avoid known intrusions
 - Database size continues to grow
 - Today Snort has > 3000 rules
 - Snort spends 80% of time doing string match
- Anomaly detection – identify new attacks
 - Probability of detection is low (very low...)



Suricata

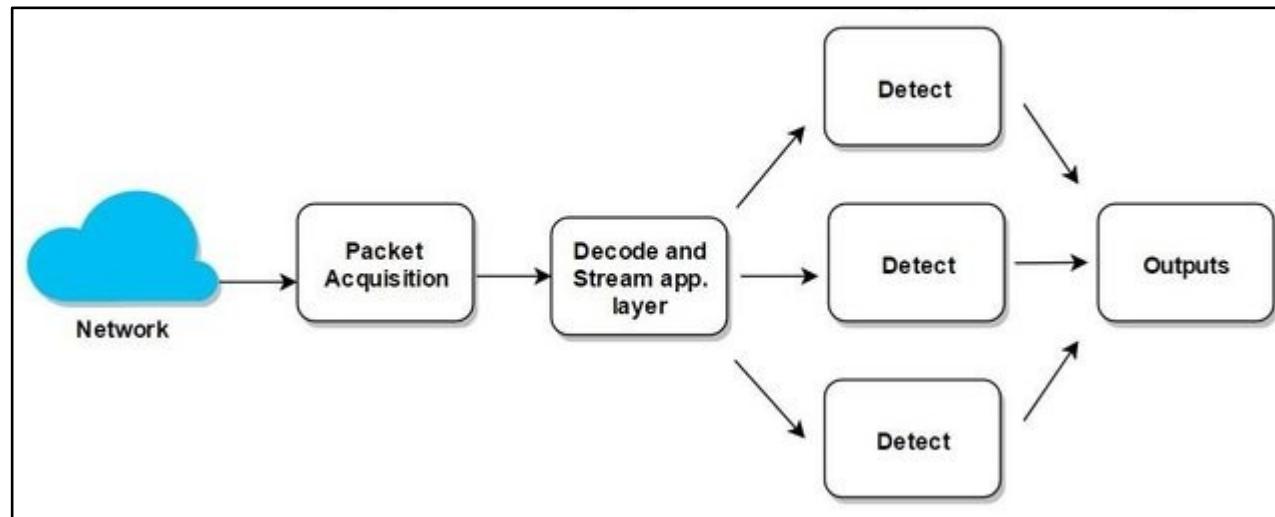
- High performance Network IDS, IPS and Network Security Monitoring engine
- Open source and owned by a community-run non-profit foundation, the Open Information Security Foundation (OISF)
- Real time intrusion detection (IDS), inline intrusion prevention (IPS), network security monitoring (NSM) and offline pcap processing
- Developed to overcome snort limitations
 - Financial help from the U.S. Department of Homeland Security

Suricata vs Snort architecture



Snort

Suricata





Suricata features

- Suricata's workload is distributed thanks to its multi-threading capabilities and GPU acceleration
- Support for packet decoding of: IPv4, IPv6, TCP, UDP, SCTP, ICMPv4, ICMPv6, GRE, Ethernet, PPP, PPPoE, Raw, SLL, VLAN, QINQ, MPLS, ERSPAN
- App layer decoding of: HTTP, SSL, TLS, SMB, DCERPC, SMTP, FTP, SSH, DNS, Modbus, ENIP/CIP, DNP3, NFS, NTP, DHCP, TFTP, KRB5, IKEv2
- IP reputation
- Integration with other solutions, such as SIEMs and databases using YAML and JSON files as inputs and outputs
- Incorporates the Lua scripting language to create rules that identify conditions that would be difficult or impossible with a legacy Snort Rule
- A lot more: <https://suricata-ids.org/features/all-features/>



Worth mentioning Zeek (old name: Bro)

- Older than snort (1998 vs. 1995)
- Support and attention following a grant from the National Science Foundation in 2010
- Zeek uses Bro Script (similar to C++) rather than a rules structure to define network traffic
- It can be used as IDS but also as a network monitor
 - However, the deep-packet inspection aspect of Zeek makes it far more resource intensive for basic alerts
 - Setup and maintenance of Zeek can be challenging at best for even experienced users



Bro script example: Matching URLs

- Report all Web requests for files called “passwd”

```
event http_request(c: connection,          # Connection.
                    method: string,        # HTTP method.
                    original_URI: string, # Requested URL.
                    unescaped_URI: string, # Decoded URL.
                    version: string)      # HTTP version.

{
    if ( method == "GET" && unescaped_URI == /.*passwd/ )
        NOTICE(...); # Alarm.
}
```



Bro script example: Scan Detection

- Count failed connection attempts per source address

```
global attempts: table[addr] of count &default=0;

event connection_rejected(c: connection)
{
    local source = c$id$orig_h;          # Get source address.

    local n = ++attempts[source];      # Increase counter.

    if ( n == SOME_THRESHOLD )        # Check for threshold.
        NOTICE(...);                 # Alarm.
}
```



fail2ban



A small break...



Student's Opinions Questionnaires (OPIS)

- Two options:
 - the infostud app (probably best option)
 - the infostud website
 - follow the following instructions
https://www.uniroma1.it/sites/default/files/field_file_allegati/vadevecum_opis_eng_27_11_2018_002_modalita_compatibilita.pdf
 - use this course code **CITY115P**
- Be positive!



Fail2ban

Fail2Ban



- Intrusion prevention software framework that protects computer servers from brute-force attacks
 - https://www.fail2ban.org/wiki/index.php/Main_Page
- Fail2ban scans log files and bans IP addresses of hosts that have too many failures within a specified time window
- Think of it as a dynamic firewall: it detects incoming connection failures, and dynamically adds a firewall rule to block that host after too many failures



Fail2ban features

- client/server
- multi-threaded
- autodetection of datetime format
- lots of predefined support
 - services – sshd, apache, qmail, proftpd, sasl, asterisk, squid, vsftpd, aspp, etc
 - actions – iptables, tcp-wrapper, shorewall, sendmail, ipfw, etc



Fail2ban limitations

- Reaction time – fail2ban is a **log parser**, so it cannot do anything before something is written to the log file.
- Syslog daemons normally buffer output, so you may want to disable buffering in your syslog daemon
- fail2ban waits 1 second before checking log files for changes, so it is possible to get more failures than specified by `maxretry`
- A local user could initiate a DoS attack by forging syslog entries with the `logger(1)` command



Terminology

- fail2ban
 - Software that bans & unbans IP addresses after a defined number of failures
- (un)ban
 - (Remove)/Add a firewall rule to (un)block an IP address
- jail
 - A jail is the definition of one fail2ban-server thread that watches one or more log file(s), using one filter and can perform one or more actions
- filter
 - Regular expression(s) applied to entries in the jail's log file(s) trying to find pattern matches identifying brute-force break-in attempts
- action
 - One or more commands executed when the outcome of the filter process is true AND the criteria in the fail2ban and jail configuration files are satisfied to perform a ban



Fail2ban components

- Content of /etc/fail2ban/ directory
- fail2ban-server
 - The core of the IPS
- fail2ban-client
 - The cli interface with the server
- fail2ban-regex
 - A cli utility to test regular expressions and filters



fail2ban activity



Activity 1

- In the topology of ACME, configure the webserver to ban/unban hosts bruteforcing SSH access using fail2ban



Activity 2

- Configure fail2ban in the syslog server to ban/unban hosts that are bruteforcing other hosts
- It has to interact with the two firewalls to add the banned hosts in an alias list (like fail2ban)
 - HINT: you can use shell scripts and SSH keys, with the pfctl command
 - <https://www.openbsd.org/faq/pf/tables.html>
 - Beware that pfctl needs administrator rights, then be sure to adequately protect your script!



That's all for today

- **Questions?**
- See you next lecture!
- **References:**
 - [NIST Guide to Intrusion Detection and Prevention Systems \(IDPS\)](#)
 - Chapter 19 textbook
- **Other interesting readings:**
 - [A Framework for Constructing Features and Models for Intrusion Detection Systems](#)
 - [Specification-based anomaly detection: a new approach for detecting network intrusions](#)

Network Access Control with OpenSource tools

802.1x with PacketFence

Emanuele Gabrielli & Piergiorgio Moretti

2022 May 26



Main Goal

Increase the security level of a network by strictly verifying user (or device) identity and checking/classifying devices on the basis of (estimated) reliability.

All that maintaining a reasonable level of manageability (i.e. easy to use, support)



Starting scenario

- Static IP assignment to users (not Devices!)
- Dynamic (and random) IP assignment with DHCP
- Dynamic IP assignment with DHCP (with *Ethernet address <-> IP* association)



Starting scenario

Static IP assignment to users

Pros:

- Easy to implement on the server side. Almost no service needed
- Clear attribution of responsibilities to user

Cons:

- Manual association user <-> IP
- User need to manage the inventory of its IPs
- Enforcement on usage of the right IP (usually) is performed by Ethernet Address
 - ◆ is a weak constraint
 - ◆ user needs to ask for an IP for each device (or adapter!)



Starting scenario

Dynamic (and random) IP assignment with DHCP

Pros:

- Easy to implement on the server side. Almost no service needed
- Clear attribution of responsibilities to user

Cons:

- Manual tracking of user <-> IP association
- User need to manage the inventory of its IPs
- Enforcement on usage of the right IP (usually) is performed by Ethernet Address
 - ◆ is a weak constraint
 - ◆ user needs to ask for an IP for each device (or adapter!)



Starting scenario

Dynamic IP assignment with DHCP (with *Ethernet address <-> IP* association)

Pros:

- User can simply acquire IP address configuring DHCP options (no need to remember, nor to manually configure IP)
- Clear attribution of responsibility of IP addresses

Cons:

- On the server side (the network management services and stuff) must be configured association among user <-> ethernet address <-> IP address
- Association on the server side must be defined in advance => we need to pre-allocate all IPs for requesting Users X Device (this is not an optimal allocation of IPs)
- Enforcement on usage of the right IP (usually) is performed by Ethernet Address
 - ◆ is a weak constraint
 - ◆ user needs to ask for an IP for each device (or adapter!)



Desiderata

- User connects its device to the network and authenticating himself using institutional account (es: email address or windows domain) he automatically (i.e. DHCP) receive an IP address
- Allow the user to request (i.e. self registration with email, social login) a new account at the first access
- User can simultaneously connect two or more devices (each one having its own IP address)
- User device can save credentials so to automatically reply to authentication challenge
- Define policies to separate devices on different networks (i.e. VLAN) with levels of trust and/or services
- Avoid to use hardware vendor solutions which imposes to have all required hardware of the same vendor
- Use OpenSource (better if free) software solutions with the possibility of a commercial (and community) support



An Integrated solution

Network Access Control (NAC) can provide a reasonable and integrated solution to manage the access to the network providing a good level of security.

“Network access control is a computer networking solution that uses a set of protocols to define and implement a policy that describes how to secure access to network nodes by devices when they initially attempt to access the network.”(*)

(*): Matias, Jon; Garay, Jokin; Mendiola, Alaitz; Toledo, Nerea; Jacob, Eduardo (2014). "FlowNAC: Flow-based Network Access Control". 2014 Third European Workshop on Software Defined Networks: 79–84.
doi:10.1109/EWSDN.2014.39.



Which NAC ?

Main requirements: OpenSource and free

Available choices:

1. PacketFence: <https://www.packetfence.org/>
 - o well done web site, documentation and installation procedures
 - o broad community support
 - o commercial support available
2. OpenNAC: <https://sourceforge.net/projects/opennac/>
 - o latest update 7 years ago
 - o only the enterprise version seems to be available
3. FreeNAC: <https://freenac.net>
 - o the homepage is not accessible from some browser



Selected solution

PacketFence:

“PacketFence is a **fully supported**, trusted, Free and Open Source network access control (NAC) solution. Boasting an impressive feature set including a **captive-portal for registration** and remediation, centralized wired, wireless and VPN management, industry-leading **BYOD capabilities**, **802.1X and RBAC support**, **integrated network anomaly detection with layer-2 isolation of problematic devices**; PacketFence can be used to effectively secure small to very large heterogeneous networks.

Released under the GPL, PacketFence is built using trusted open source components that allows it to offer an impressive amount of features.”



PacketFence selected Features

Flexible VLAN Management and Role-Based Access Control

Guest Access - Bring Your Own Device (BYOD)

More Built-in Violation Types

Automatic Registration

Flexible Authentication

Routed Networks

Pass-Through

Standards-Based

Extensible / Easily Customizable



PacketFence: Automatic Registration

▼ Automatic Registration

Because most networks in production are already very large and complex, PacketFence provides several means to automatically register a client or device.

- > By network device

A network device (Switch, AP, Wireless Controller) can be set to automatically register all the MAC addresses that request access to the network. Very helpful for a transition into production.

- > By DHCP fingerprinting

DHCP fingerprinting can be used to automatically register specific device types (eg. VoIP phones, printers).

- > By MAC address Vendor

The vendor portion of a MAC address can be used to automatically register devices from a vendor. For example, all Apple products could be automatically registered using such a rule.

- > and more

Snort, Nessus, OpenVAS, Browser User-Agent and even more techniques could also be used to automatically register devices.



PacketFence: Flexible VLAN management

▼ **Flexible VLAN Management and Role-Based Access Control**

The solution is built around the concept of network isolation through VLAN assignment. For more details on how this work see the [Technical Introduction](#) page. Because of its long experience and several deployments, the VLAN management of PacketFence grew to be very flexible over the years. Your VLAN topology can be kept as it is and only two new VLAN will need to be added throughout your network: registration VLAN and isolation VLAN. Moreover, PacketFence can also make use of roles support from many equipment vendors.

VLAN and roles can be assigned using the various means:

- Per switch (default for VLAN)
- Per client category (default for roles)
- Per client
- Using any arbitrary decision (if you use our perl extension points)

Also, the per-switch method can be combined with the others. For example, with a default PacketFence setup, a VLAN or a role can be assigned to your printers and your PCs (if categorized properly) based on what equipment they are connected to. This implies that you can easily have per-building per-device type VLANs.



PacketFence: Guest Access (BYOD)

▼ **Flexible VLAN Management and Role-Based Access Control**

The solution is built around the concept of network isolation through VLAN assignment. For more details on how this work see the [Technical Introduction](#) page. Because of its long experience and several deployments, the VLAN management of PacketFence grew to be very flexible over the years. Your VLAN topology can be kept as it is and only two new VLAN will need to be added throughout your network: registration VLAN and isolation VLAN. Moreover, PacketFence can also make use of roles support from many equipment vendors.

VLAN and roles can be assigned using the various means:

- Per switch (default for VLAN)
- Per client category (default for roles)
- Per client
- Using any arbitrary decision (if you use our perl extension points)

Also, the per-switch method can be combined with the others. For example, with a default PacketFence setup, a VLAN or a role can be assigned to your printers and your PCs (if categorized properly) based on what equipment they are connected to. This implies that you can easily have per-building per-device type VLANs.



PacketFence: block unwanted devices

▼ More Built-in Violation Types

Looking at automatically blocking particular devices on your network? PacketFence is for you. In addition to using Windows Management Instrumentation (WMI), Snort, Suricata, OpenVAS or Nessus as a source of information, PacketFence can combine the following detection mechanisms to effectively block network access from those unwanted devices :

> DHCP Fingerprint

PacketFence can block devices based on their DHCP fingerprint. Nearly every operating systems out there have an unique DHCP fingerprint. PacketFence can make use of this information and block network access from those devices. Based on DHCP fingerprints, you could automatically block, for example:

- Sony PlayStation devices or any other game consoles
- Wireless access points (WAPs)
- VoIP phones

> User-Agent

PacketFence can block devices based on the provided User-Agent when those particular devices perform network activity using their embedded Web browser. Using this, you could automatically block, for example:

- Apple iPod or iPhone devices
- Everyone using an old Microsoft Internet Explorer (IE) release

> MAC addresses

PacketFence can block network access to devices having a specific MAC address pattern. Using this, you could automatically block, for examples, all devices from a specific network vendor.



PacketFence: Flexible Authentication

▼ Flexible Authentication

PacketFence can authenticate your users using several protocols/standards. This allows you to integrate PacketFence in your environment without requiring your users to remember yet another username and password. Supported authentication sources are:

LDAP

- Microsoft Active Directory
 - Novell eDirectory
 - OpenLDAP
 - ... or any LDAP-compliant server
-

RADIUS

- Cisco ACS
 - RADIUS (FreeRADIUS, Radiator, etc.)
 - Microsoft NPS
 - ... or any RADIUS-compliant server
-

Local user file (Apache htpasswd format)

OAuth2

- Facebook
 - Google
 - GitHub
 - LinkedIn
 - Microsoft Live
 - Twitter
-

SAML

Moreover, PacketFence can also use its internal SQL database to authenticate locally-created users.



PacketFence: Routed Networks

▼ **Routed Networks**

PacketFence's architecture allows it to work over routed networks. The server can be located in your datacenter and can still effectively secure branch offices.



PacketFence: Pass-Through

▼ **Pass-Through**

PacketFence can be configured to allow access to specified resources even when the node is in isolation. This allows you to give access to specific tools or patches through the captive portal.



PacketFence: Standards-Based

▼ Standards-Based

PacketFence is built using open standards to avoid vendor lock-in. Among the standards we support and use, there are:

- 802.1X
- Simple Network Management Protocol (SNMP)
- Standard SNMP management information base (MIB) like BRIDGE-MIB, Q-BRIDGE-MIB, IF-MIB, IEEE8021-PAE-MIB
- RADIUS
- Netflow / IPFIX
- Wireless ISP Roaming (WISPR)



PacketFence: Customization

▼ Extensible / Easily Customizable

PacketFence has a couple of extension points where you can override PacketFence's default behavior with a little bit of Perl code. The API has been designed to be easy to understand with only a couple of high-level entry points. Several examples are already there in the source code but commented. Also, when upgrading, PacketFence doesn't replace the files in the extensions points, this way you keep your modified behavior on upgrades.

The captive portal templates are also easily customizable with HTML and CSS knowledge. They are built using Perl's [Template Toolkit](#).



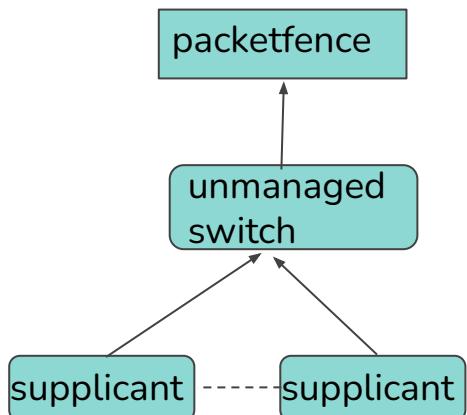
PacketFence: Enforcement types

Inline: In many other NAC solutions, it is not possible to support unmanageable devices such as entry-level consumer switches or access-points. Using PacketFence, with the new inline mode, it can be used in-band for those devices. So in other words, PacketFence would become the gateway of that inline network, and NAT or route the traffic using IPTables/IPSet to the Internet (or to another section of the network).

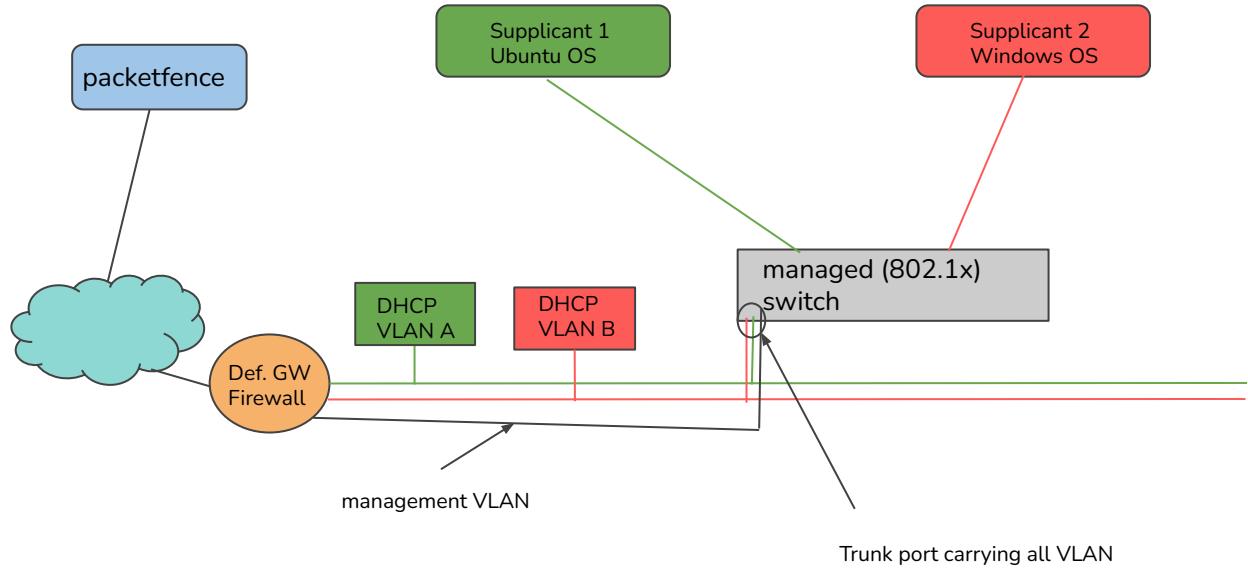
Out-of-Band using SNMP or RADIUS (or VLAN Enforcement): VLAN assignment is currently performed using several different techniques. These techniques are compatible one to another, but not on the same switch port. This means that you can use the more secure and modern techniques for your latest switches and another technique on the old switches that doesn't support latest techniques. As its name implies, VLAN assignment means that PacketFence is the server that assigns the VLAN to a device. This VLAN can be one of your VLANs or it can be a special VLAN where PacketFence presents the captive portal for authentication or remediation. VLAN assignment effectively isolates your hosts at the OSI Layer2 meaning that it is the trickiest method to bypass and is the one which adapts best to your environment since it glues into your current VLAN assignment methodology.

Inline vs VLAN enforcement

Inline



VLAN Enforcement





Applied enforcement: VLAN enforcement with 802.1x

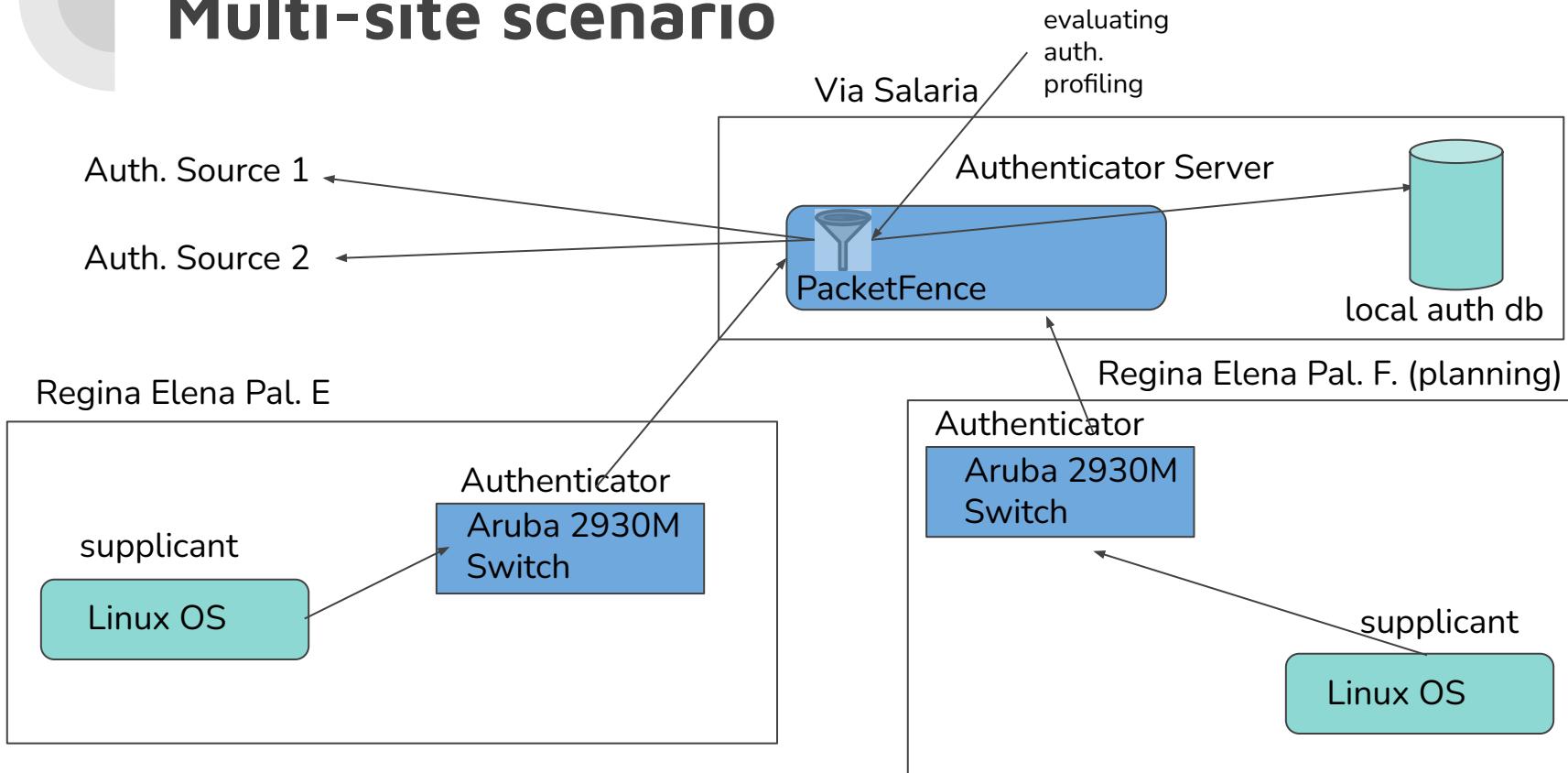
Wired: 802.1X + MAC Authentication

802.1X provides port-based authentication, which involves communications between a supplicant, authenticator (known as NAS), and authentication server (known as AAA). The supplicant is often software on a client device, such as a laptop, the authenticator is a wired Ethernet switch or wireless access point, and the authentication server is generally a RADIUS server.

The supplicant (i.e., client device) is not allowed access through the authenticator to the network until the supplicant's identity is authorized. With 802.1X port-based authentication, the supplicant provides credentials, such as user name / password or digital certificate, to the authenticator, and the authenticator forwards the credentials to the authentication server for verification. If the credentials are valid (in the authentication server database), the supplicant (client device) is allowed to access the network. The protocol for authentication is called Extensible Authentication Protocol (EAP) which have many variants. Both supplicant and authentication servers need to speak the same EAP protocol. Most popular EAP variant is PEAP-MsCHAPv2 (supported by Windows / Mac OSX / Linux for authentication against AD).



Multi-site scenario





PacketFence: VLAN enforcement with 802.1x

In this context, PacketFence runs the authentication server (a FreeRADIUS instance) and will return the appropriate VLAN to the switch. A module that integrates in FreeRADIUS does a remote call to the PacketFence server to obtain that information. More and more devices have 802.1X supplicant which makes this approach more and more popular.

MAC Authentication is a new mechanism introduced by some switch vendor to handle the cases where a 802.1X supplicant does not exist. Different vendors have different names for it. Cisco calls it MAC Authentication Bypass (MAB), Juniper calls it MAC RADIUS, Extreme Networks calls it Netlogin, etc. After a timeout period, the switch will stop trying to perform 802.1X and will fallback to MAC Authentication. It has the advantage of using the same approach as 802.1X except that the MAC address is sent instead of the user name and there is no end-to-end EAP conversation (no strong authentication). Using MAC Authentication, devices like network printer or non-802.1X capable IP Phones can still gain access to the network and the right VLAN.



PacketFence: VLAN enforcement

In order to build a VLAN isolation setup you need :

- a supported switch (please consult the list of supported switch vendors and types in the *Network Devices Configuration Guide* including information on uplinks)
- a normal, registration and isolation VLAN (VLAN numbers and subnets)
- a switch port for the PacketFence (PacketFence) server which needs to be configured as a dot1q trunk (several VLANs on the port)

Throughout this configuration example we use the following assumptions for our network infrastructure:

- VLAN 20 is the management VLAN
- VLAN 102 is the registration VLAN (unregistered devices will be put in this VLAN)
- VLAN 103 is the isolation VLAN (isolated devices will be put in this VLAN)
- VLAN 104 is the normal VLAN (registered devices will be put in this VLAN)



Isolation and Registration VLAN

You need to create a registration VLAN (with a DHCP server, but no routing to other VLANs) in which PacketFence will put unregistered devices. If you want to isolate computers which have open security event in a separate VLAN, an isolation VLAN needs also to be created.

Suplicants in the Isolation or Registration VLAN have the PacketFence as default Gateway!

=> We need to manage *PacketFence Routed Network!*

Why Isolate a device/suplicant?

The screenshot shows a network management interface with a sidebar and a main content area.

Left Sidebar:

- Status
- Reports
- Auditing
- Nodes
- Users
- Configuration** (selected)
- Policies and Access Control**
- Compliance**
- Fingerprint Profiling
- General Settings
- Device change detection
- Combinations
- Devices
- DHCP Fingerprints
- DHCP Vendors
- DHCPv6 Fingerprints
- DHCPv6 Enterprises
- MAC Vendors
- User Agents
- Network Anomaly Detection
- Scans
- Scan Engines
- WMI Rules
- Security Events** (selected)
- Integration
- Advanced Access Configuration
- Network Configuration
- System Configuration

Main Content Area:

Security Events

New Security Event

Status	ID	Description	Priority	Template	Action
OFF	defaults		4	generic	<button>Delete</button> <button>Clone</button> <button>Preview</button>
ON	1100001	Nessus Scan	4	failed_scan	<button>Delete</button> <button>Clone</button> <button>Preview</button>
ON	1100002	OpenVAS scan	4	failed_scan	<button>Delete</button> <button>Clone</button> <button>Preview</button>
OFF	1100003	MAC Vendor isolation example	4	banned_devices	<button>Delete</button> <button>Clone</button> <button>Preview</button>
OFF	1100004	Ancient OS isolation example	4	banned_os	<button>Delete</button> <button>Clone</button> <button>Preview</button>
OFF	1100006	P2P Isolation (snort example)	4	p2p	<button>Delete</button> <button>Clone</button> <button>Preview</button>
OFF	1100007	Auto-register Device example	1	generic	<button>Delete</button> <button>Clone</button> <button>Preview</button>
OFF	1100008	Disable NATing Routers and APs	4	nat	<button>Delete</button> <button>Clone</button> <button>Preview</button>
OFF	1100009	MAC isolation example	4	banned_devices	<button>Delete</button> <button>Clone</button> <button>Preview</button>
ON	1100010	Rogue DHCP	4	roguedhcp	<button>Delete</button> <button>Clone</button> <button>Preview</button>
OFF	1100011	Bandwidth Limit example (20GB/month)	4	bandwidth_limit	<button>Delete</button> <button>Clone</button> <button>Preview</button>
OFF	1100012	Device based Bandwidth Limit example (Android, 1GB/month)	4	bandwidth_limit	<button>Delete</button> <button>Clone</button> <button>Preview</button>
ON	1100013	Hostname change	4	generic	<button>Delete</button> <button>Clone</button> <button>Preview</button>
ON	1100014	Connection transport change	4	generic	<button>Delete</button> <button>Clone</button> <button>Preview</button>
ON	1100020	Wireless IPS	4	generic	<button>Delete</button> <button>Clone</button> <button>Preview</button>
ON	1200001	System Scan	9	system_scan	<button>Delete</button> <button>Clone</button> <button>Preview</button>



Isolate by scanning

DIPARTIMENTO
DI INFORMATICA



SAPIENZA
UNIVERSITÀ DI ROMA

Windows Patches Are Not Up-to-date

Due to the threat this poses for other systems on the network, network connectivity has been disabled until corrective action is taken. Instructions for disinfection are below.

Updating your system

Open Windows Update and apply all available updates.

help: provide info

IP 1.2.3.4

MAC 00:11:22:33:44:55



Improvements

- Develop an extension module to support ***private VLANs***
- Develop an extension module to apply **Two-Factory-Authentication**
- Evaluate to simplify the deployment avoiding to use Routed Network