Practical Network Defense

Master's degree in Cybersecurity 2024-25

Network traffic regulation with iptables

Angelo Spognardi

spognardi@di.uniroma1.it

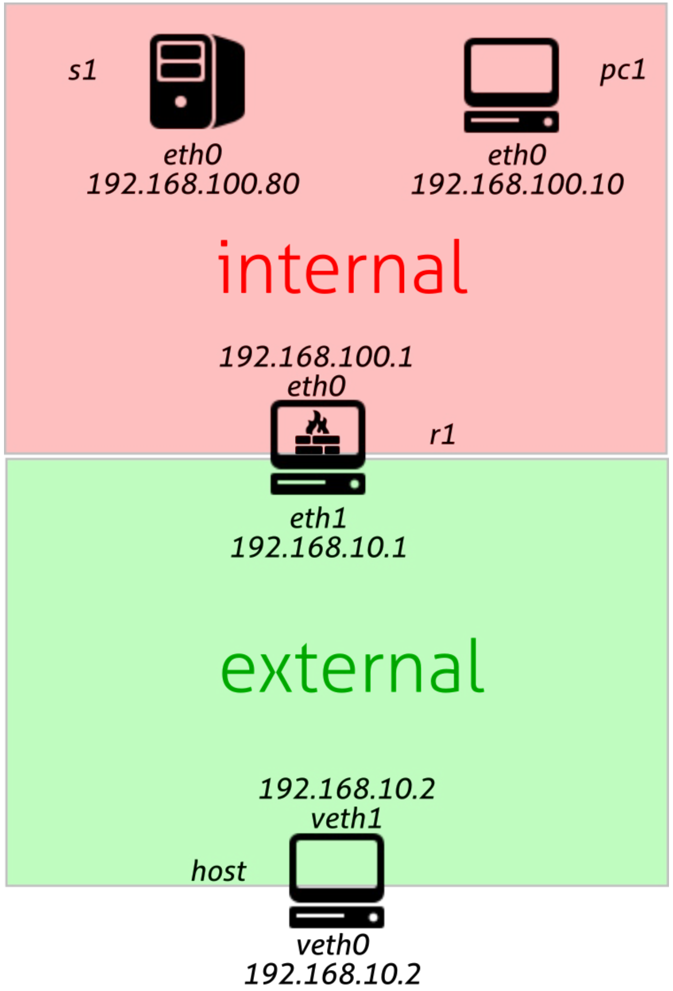
Dipartimento di Informatica

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/tcpdump
- 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/tcpdump
- 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.frozentux.net/iptables-tutorial/iptables-tutorial.html

Iptables fundamantals

- 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

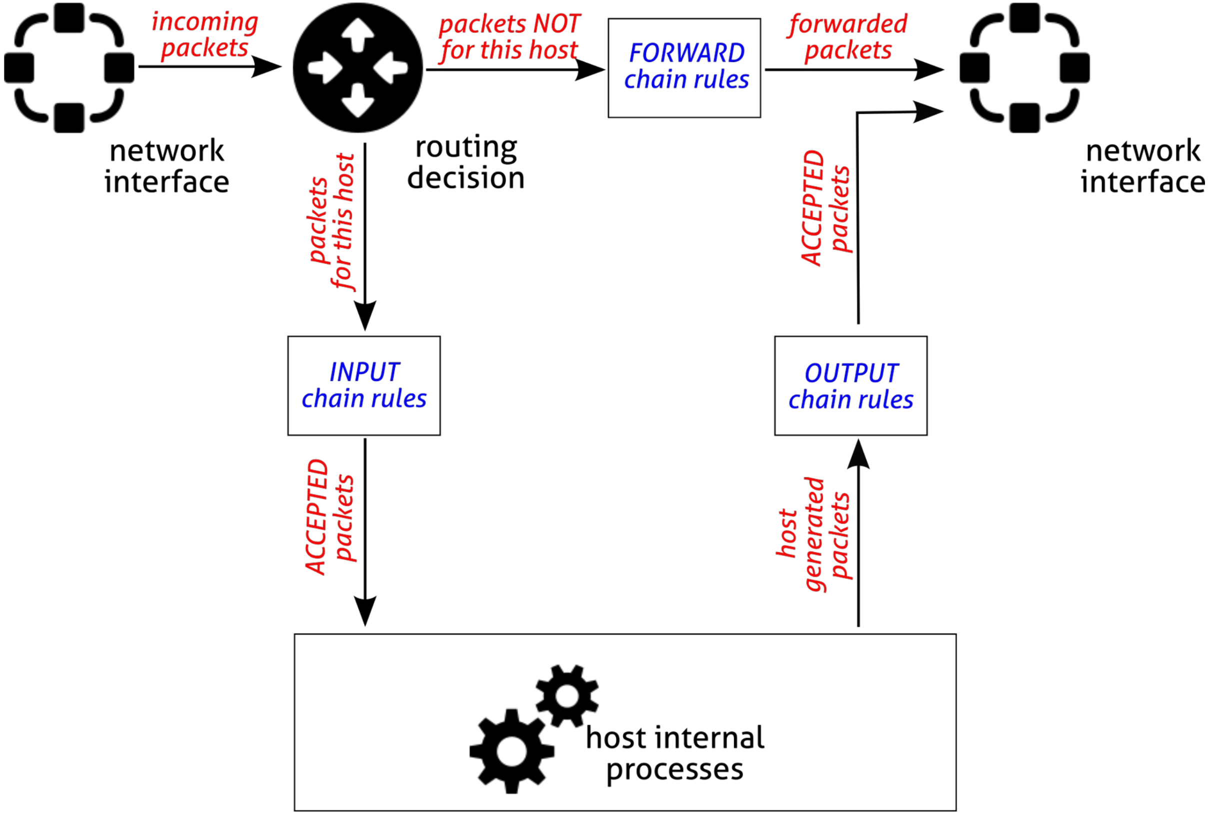
```
iptables -L
```

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 blocket, 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
--sport port	Match the **tcp/udp** source port (according to -p)
--dport port	Match the **tcp/udp** destination port (according to -p)
--icmp-type type	Match specific **icmp** packet types
-m *module*	Uses an **extension** module
-m state --state s	Enable connection tracking (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 starts a related connection (i.e., 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
```

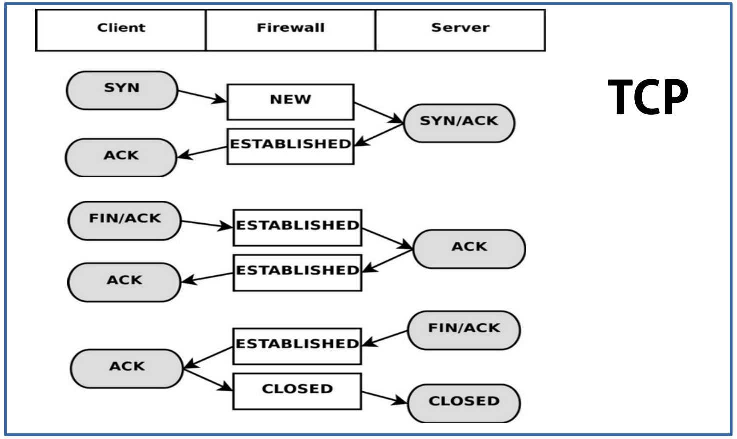
- The return traffic from web server is allowed, but only of sessions are established:

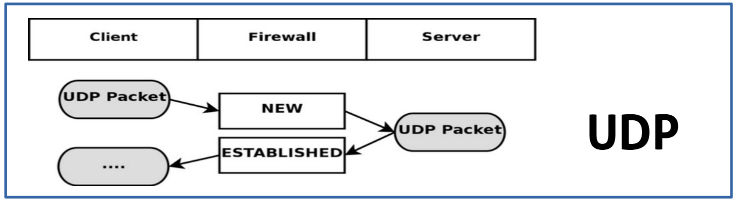
```
iptables -A FORWARD -d 0/0 -o eth0 -s 192.168.1.58 -i eth1 -p TCP \
             -m state --state ESTABLISHED -j ACCEPT
```

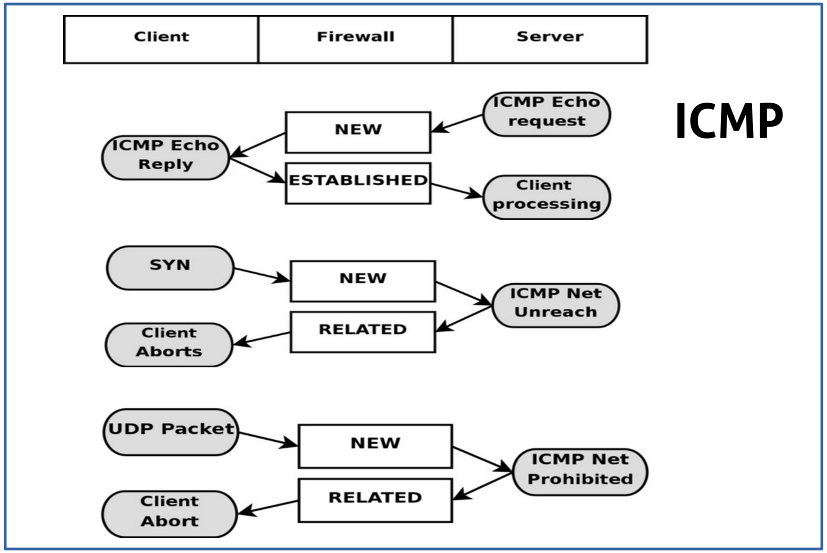
- If sessions are used, you can reduce an attack called half open
  - Half open is known to consume server all free sockets (tcp stack memory) and is senced as a denial of service attack, but it is not.
  - Sessions are usally waiting 3 minutes.

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_LKLD2OR6XH1ugPDP5LWP5/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 for the labs

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

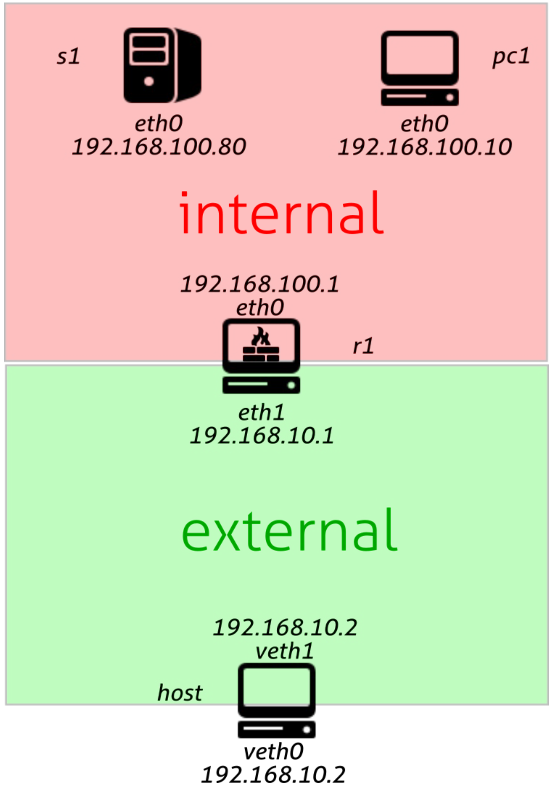
```
./connect-lab.sh 192.168.10.2/24 external
ip route add 192.168.100.0/24 via 192.168.10.1

ip addr add 2001:db8:cafe:2::2/64 dev veth0
ip route add 2001:db8:cafe:1::/64 via
2001:db8:cafe:2::1
```

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

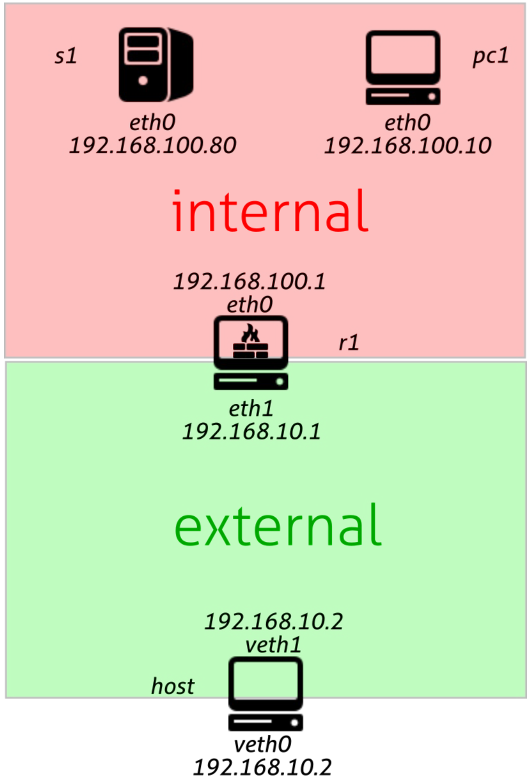
```
echo 2001:db8:cafe:1::80 s1 >>/etc/hosts
```

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



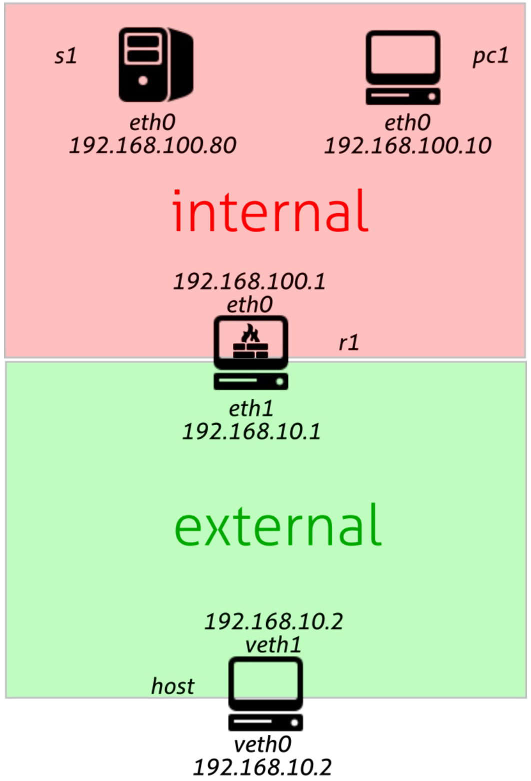
Exercise 1: pnd-labs/lab4/ex1

- 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

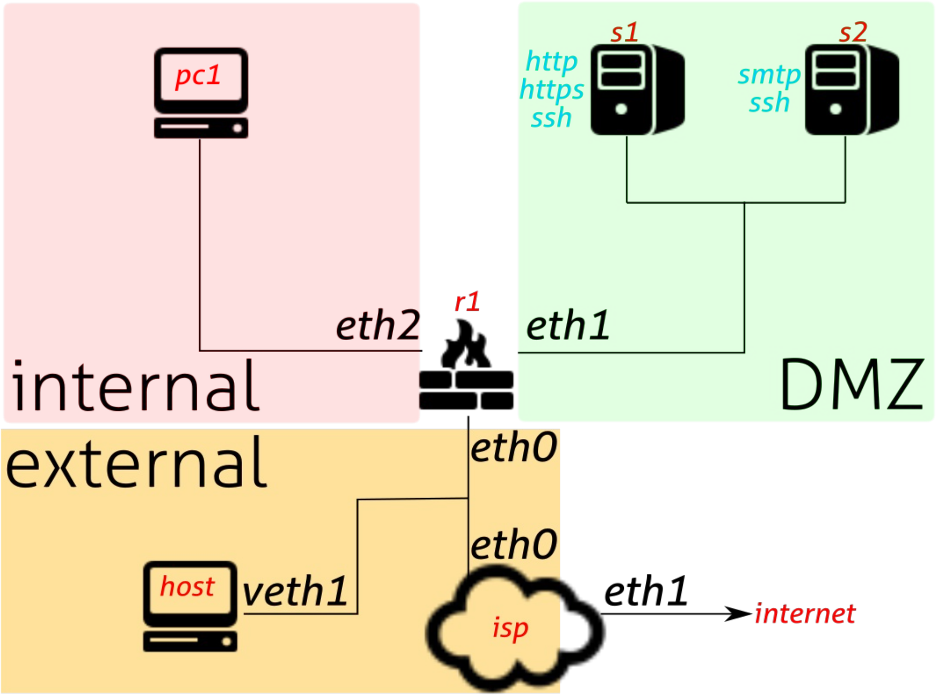
- Extend ex1 with IPv6
- Repeat the same exercise with the IPv6 addressing
- The internal network is 2001:db8:cafe:1::/64
- The external network is 2001:db8:cafe:2::/64
- You have to use ip6tables



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
- See the README file for the details

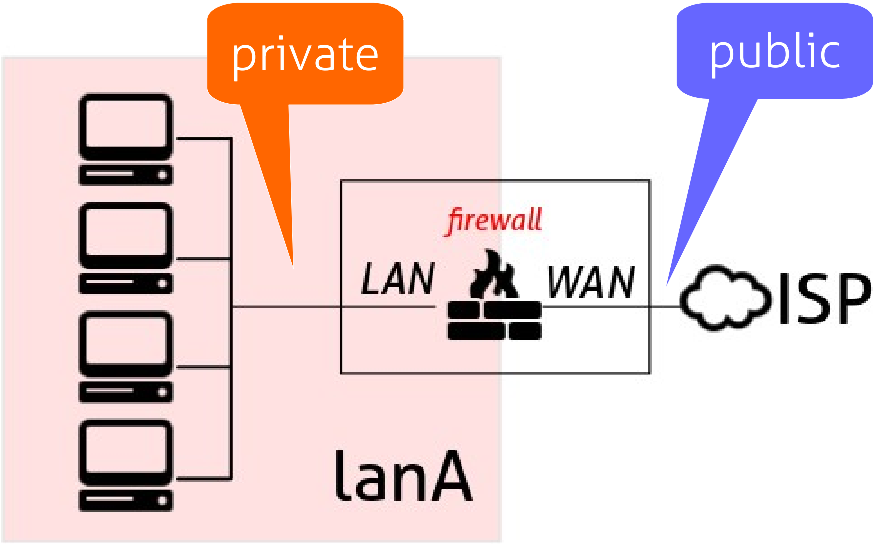


IPTABLES, tables beyond filtering

Network Address Translation (NAT)

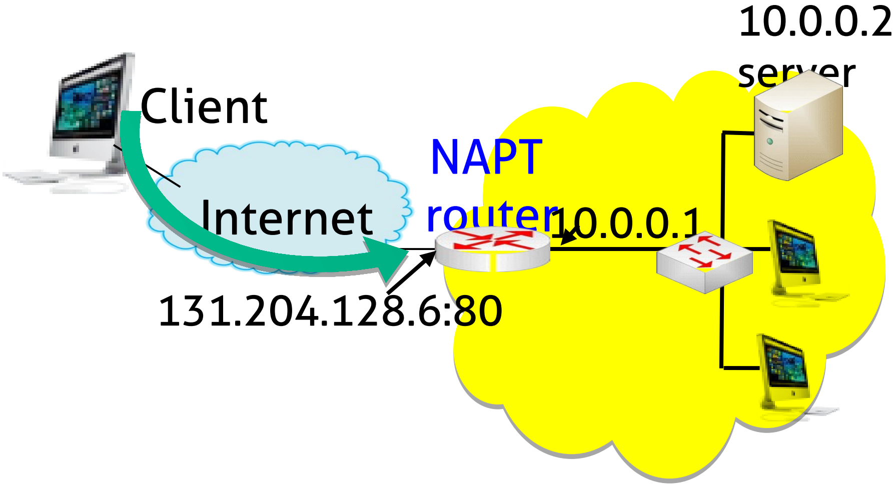
- Translate the address (f.e.: between incompatible IP addressing)
- Informally speaking, connecting to the Internet a LAN using un-routable in-house LAN addresses

- 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

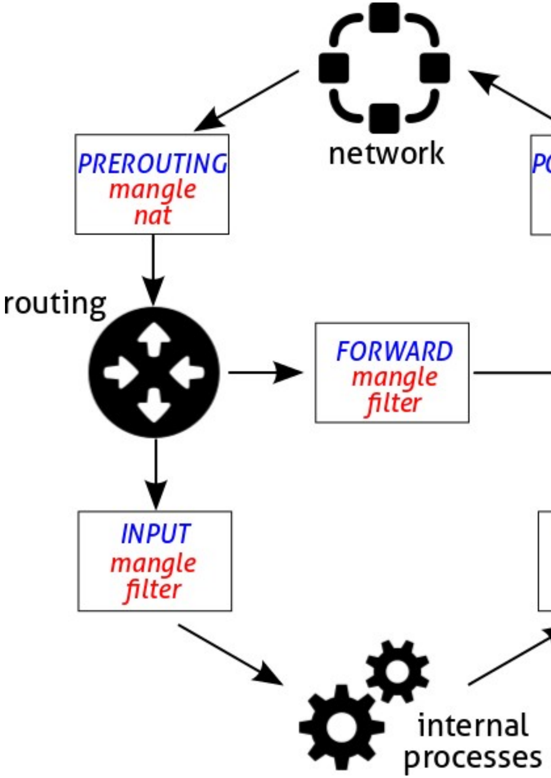
- NAPT router blocks all incoming ports by default
- Many applications have had problems with NAPT in the past in their handling of incoming requests
- Four major methods
  - Application Level Gateways (ALGs)
  - Static port forwarding
  - Universal Plug and Play (UPnP) Internet Gateway Device (IGD)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



iptables: four built-in tables

1. MANGLE: manipulate bits in TCP header
2. FILTER: packet filtering
3. NAT: network adress 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 form 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 fowarded 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
```

opnsense activity

OPNsense

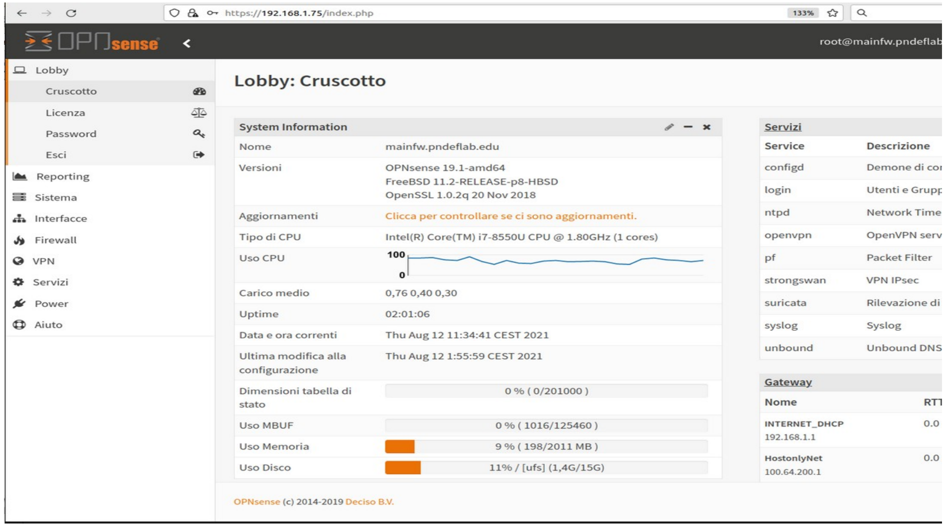
- OPNsense is an open-source router-firewall based on a particularly robust version of BSD
  - BSD is considered one of the most security-conscious Unix distributions
- OPNsense is very popular because it is also easy to use and install, besides having a free license
- It installs like a regular operating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



Lobby: Cruscotto

System Information

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
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)

Servizi

Service	Descrizione
configd	Demone di co
login	Utenti e Grupp
ntpd	Network Time
openvpn	OpenVPN serv
pf	Packet Filter
strongswan	VPN IPsec
suricata	Rilevazione di
syslog	Syslog
unbound	Unbound DNS

Gateway

Nome	RTT
INTERNET_DHCP 192.168.1.1	0.0
HostonlyNet 100.64.200.1	0.0

OPNsense (c) 2014-2019 Deciso B.V.

First access

- Use the VPN configuration received
- You will be able to connect to the internal ACME
- No connection will be possible if you do not explicitly allow incoming traffic (only ping should work, you can try…)
- To access the OPNsense control panel, you must use the client-ext1 host
- Open firefox and 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

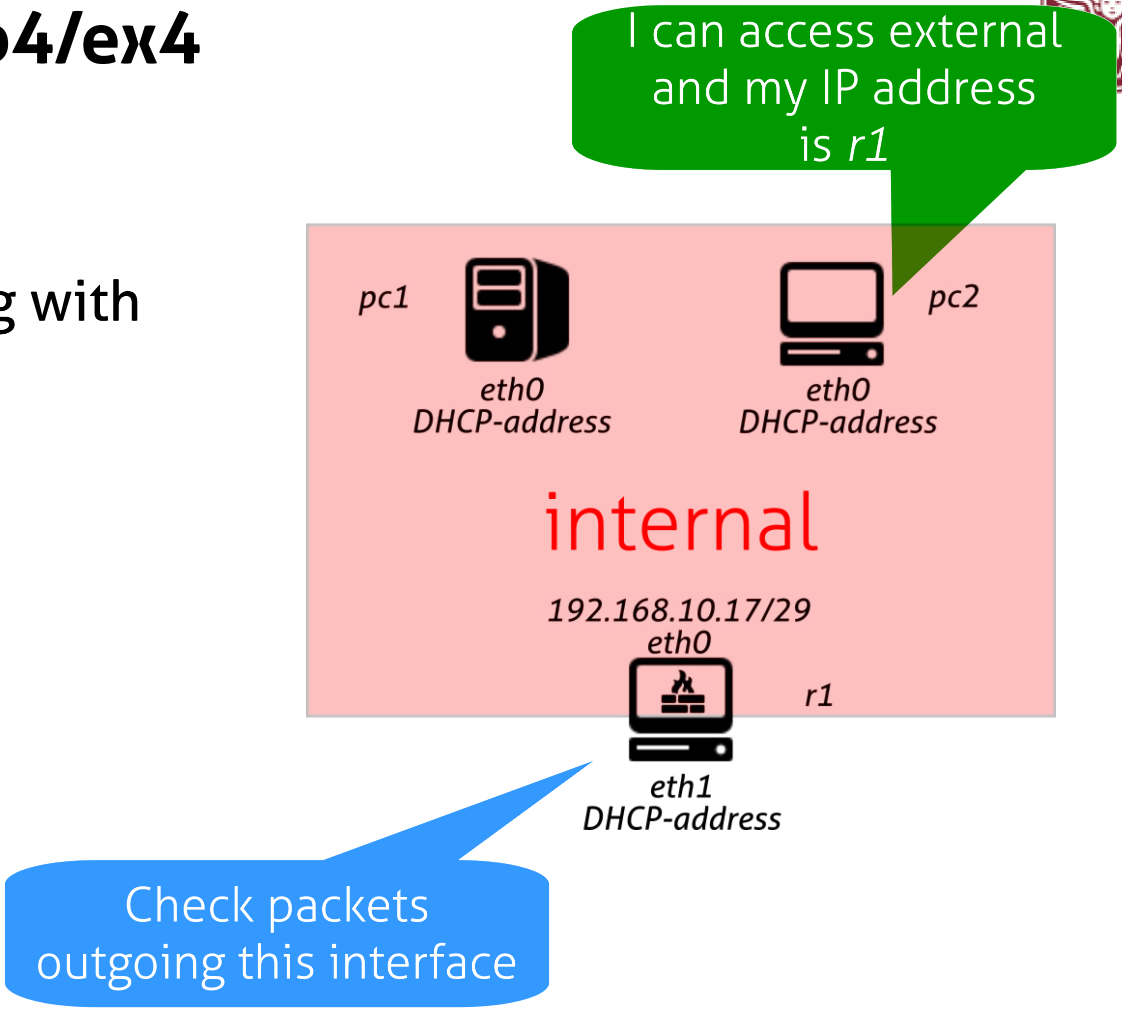
Activity

- Start a web browser in the webserver (100.100.6.2)
  - python http.server 80
- Try to connect to it from your host
- Make possible the connection
  - Interfaces[WAN] menu:
    - untick Block private networks
    - untick Block bogon networks
  - Firewall→Rules→[WAN] menu:
    - add a PASS rule for the 100.101.0.0/24 network
- Try again
- Make possible only the access to the DMZ and SSH in the other networks

Lab activity: ex4

Exercise 4: pnd-labs/lab4/ex4

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


I can access external and my IP address is *r1*
pc1
eth0
DHCP-address
pc2
eth0
DHCP-address
internal
192.168.10.17/29
eth0
r1
eth1
DHCP-address
Check packets outgoing this interface

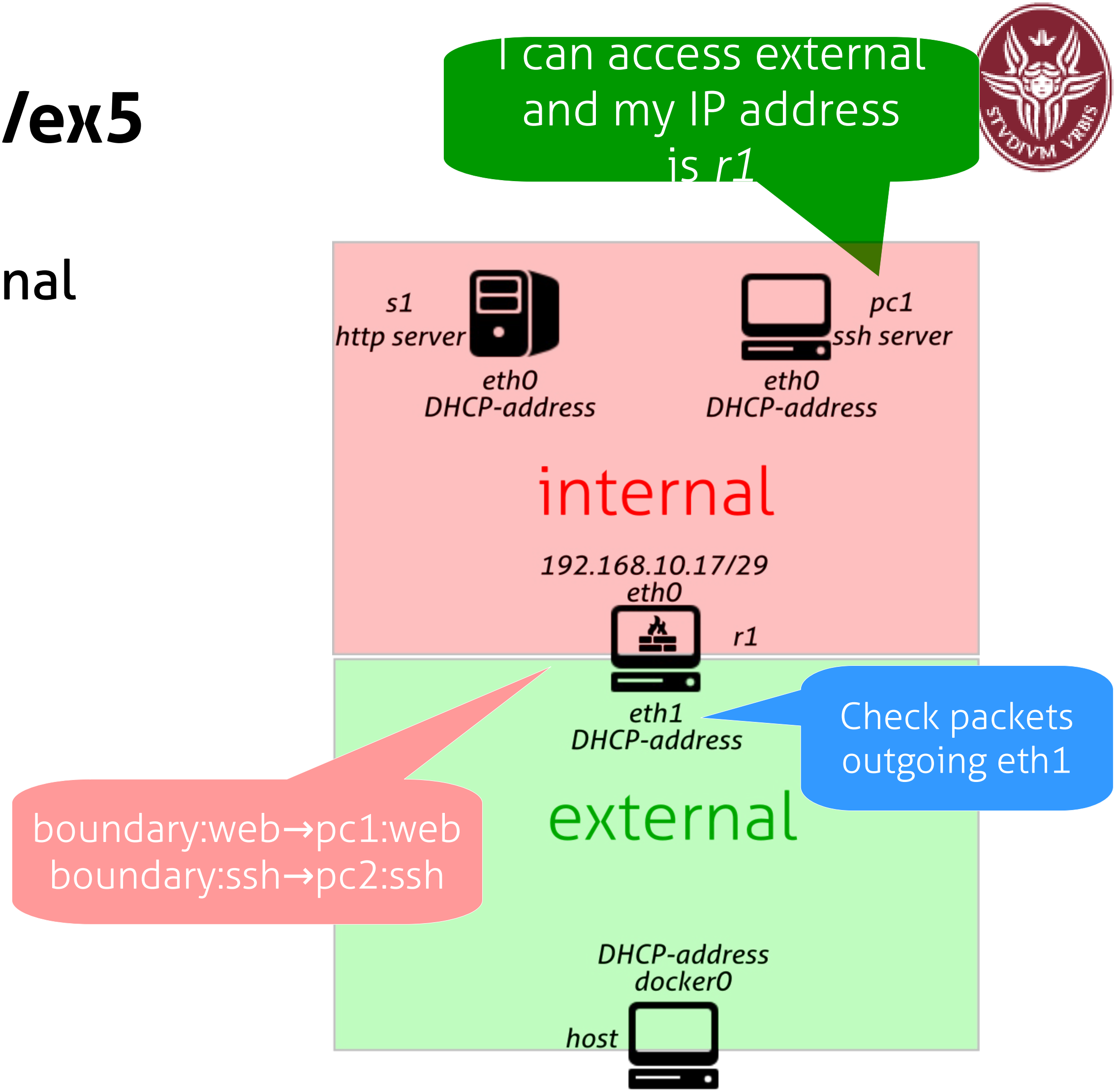
Exercise 4: Policy to protect r1

- 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

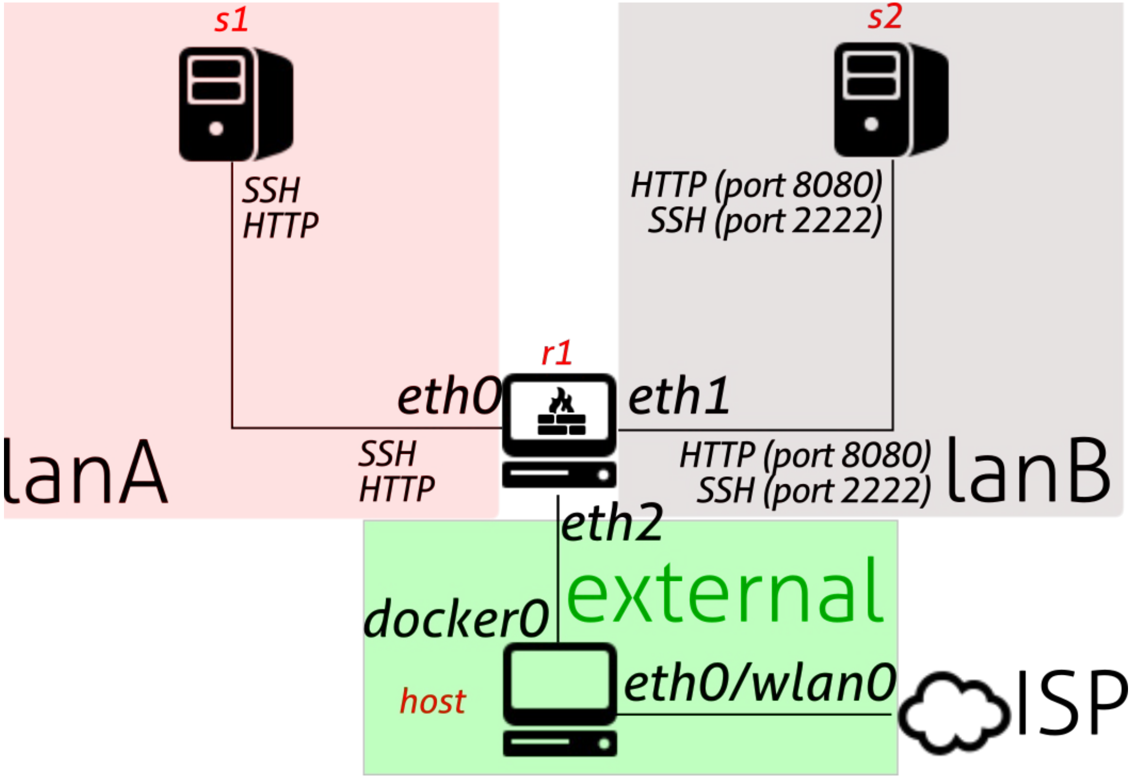
Exercise 5: pnd-labs/lab4/ex5

- 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 6: pnd-labs/lab4/ex6
NAT with 2 networks and services

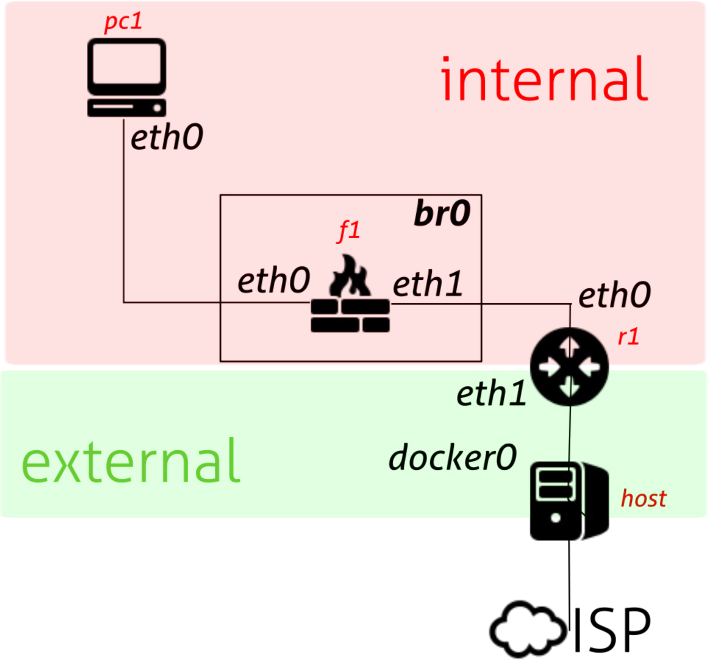


Exercise 6: policy to implement

- 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 7: 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 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