Practical Network Defense

Master's degree in Cybersecurity 2020-21

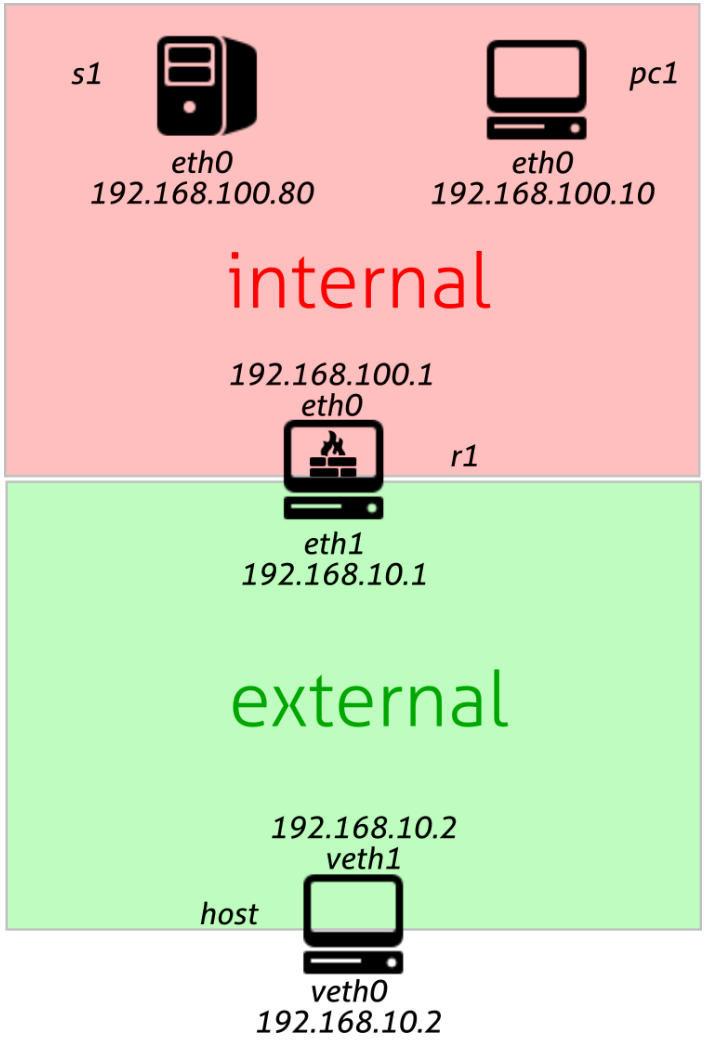
Network traffic regulation with iptables

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
- 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.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

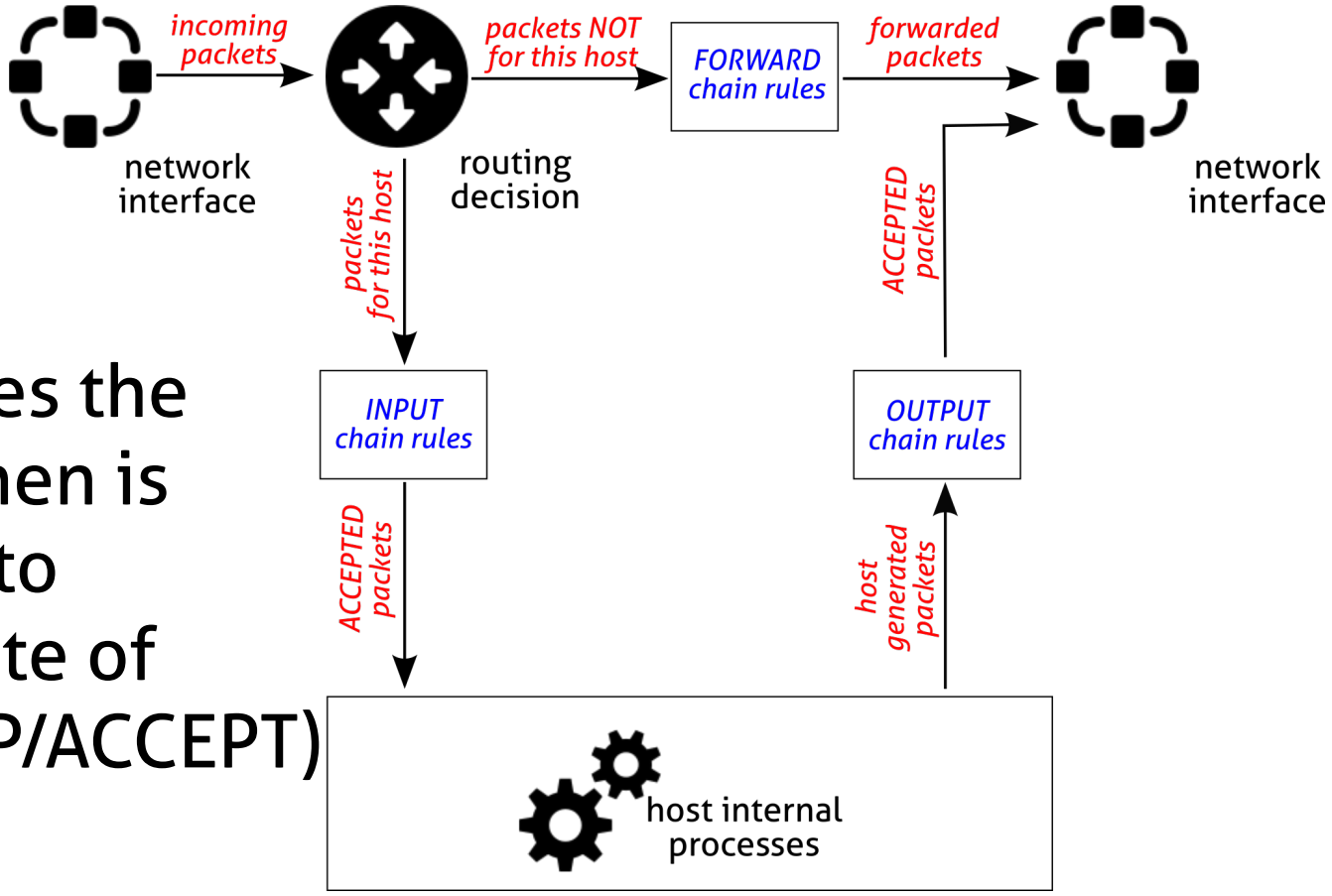
```
iptables -L
```

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
-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 *module*	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 webbserver 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
```

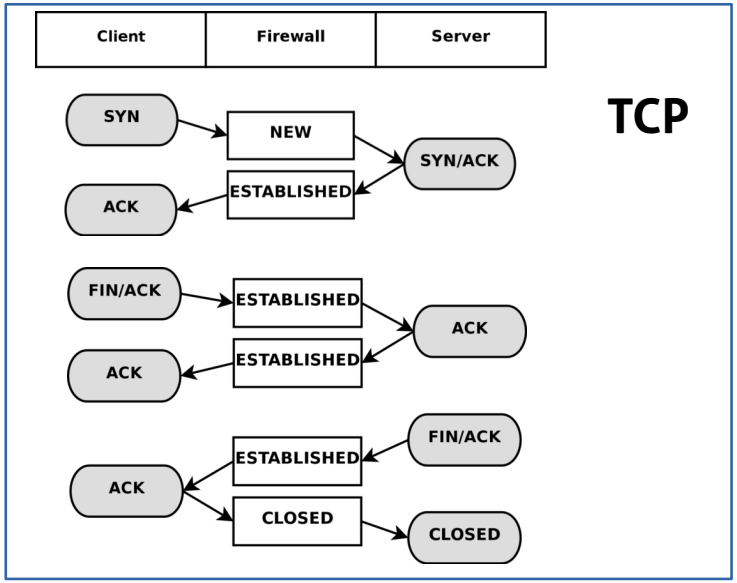
- If sessions are used, you can reduce an attack called half open

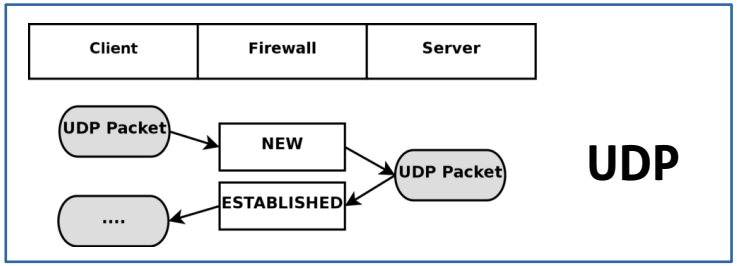
  Half open is known to consume server all free sockets (tcp stack memory) and is senced as a denial of service attack, but it is not.

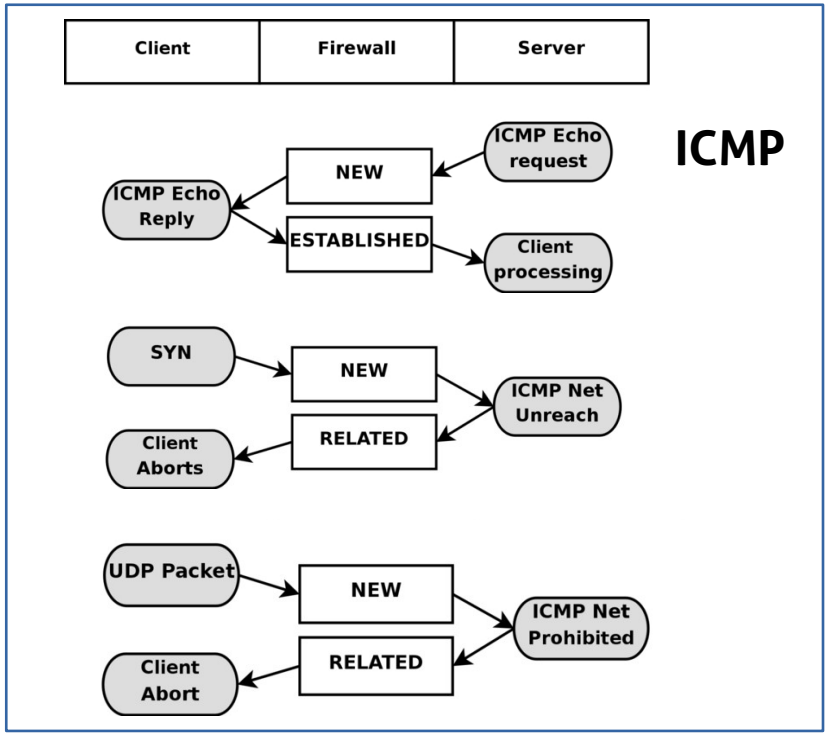
  Sessions are usally waiting 3 minutes.

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_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

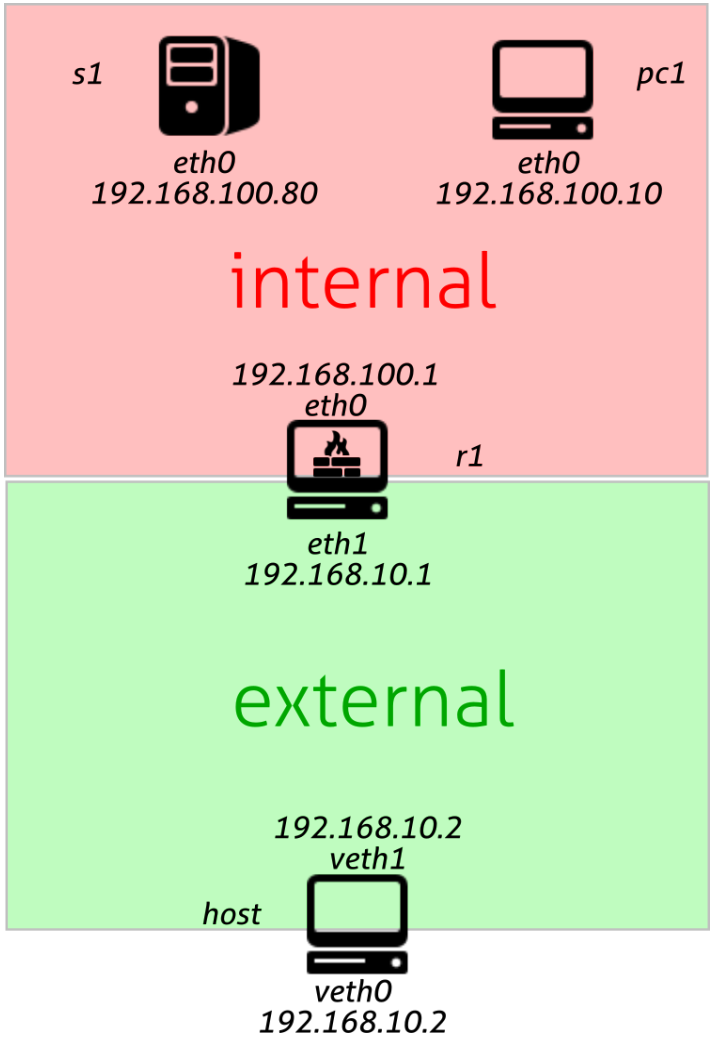
The kathara VM

- 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

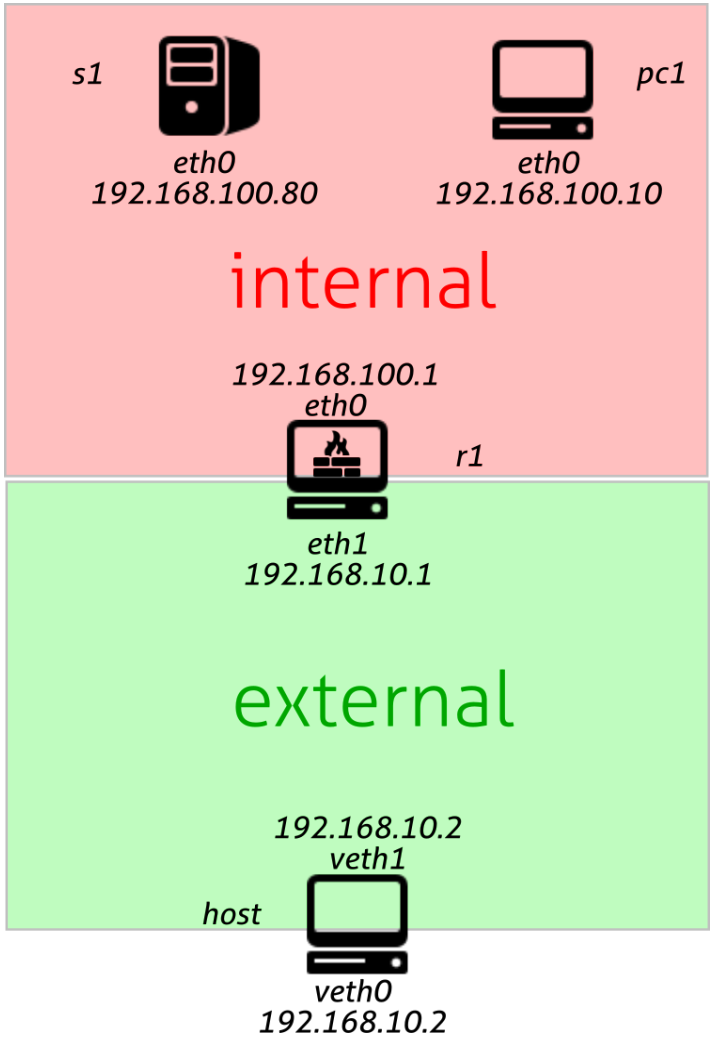
Exercise 1: pnd-labs/lab4/ex1

- 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: ssh, http on different ports



Exercise 2: pnd-labs/lab4/ex2

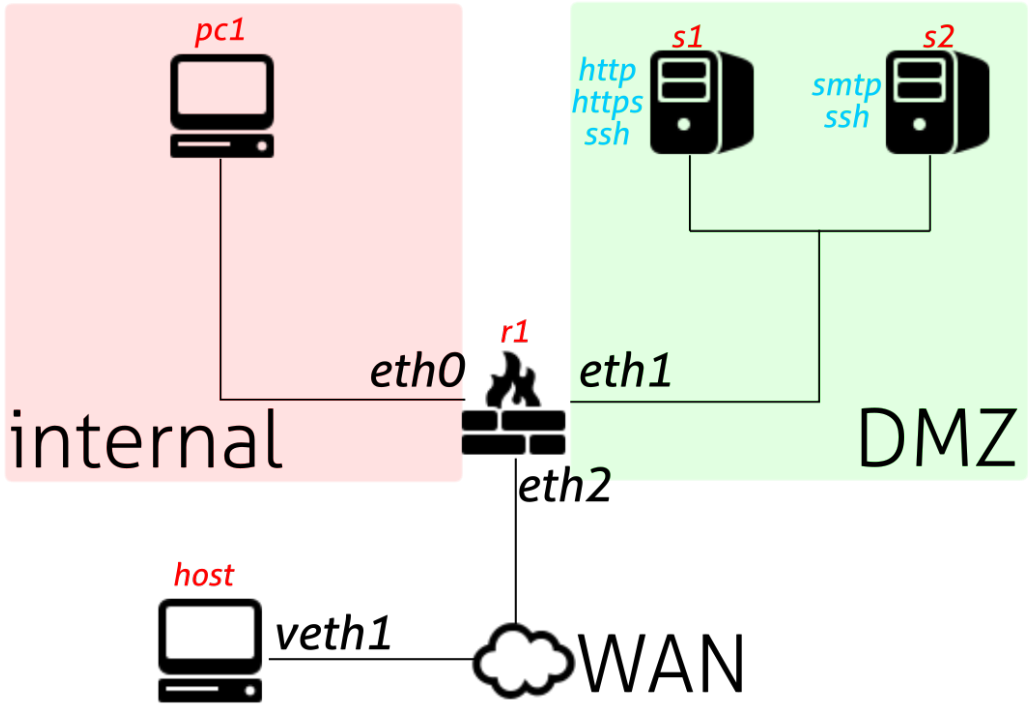
- Extend ex1 with IPv6
- Repeat the same exercise with the IPv6 addressing
- The internal network is 2001:db8:cafe:1::/64
- The external network is 2001:db8:cafe:2::/64
- You have to use ip6tables



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