IN5290 Ethical Hacking

Lecture 3: Network reconnaissance, port scanning

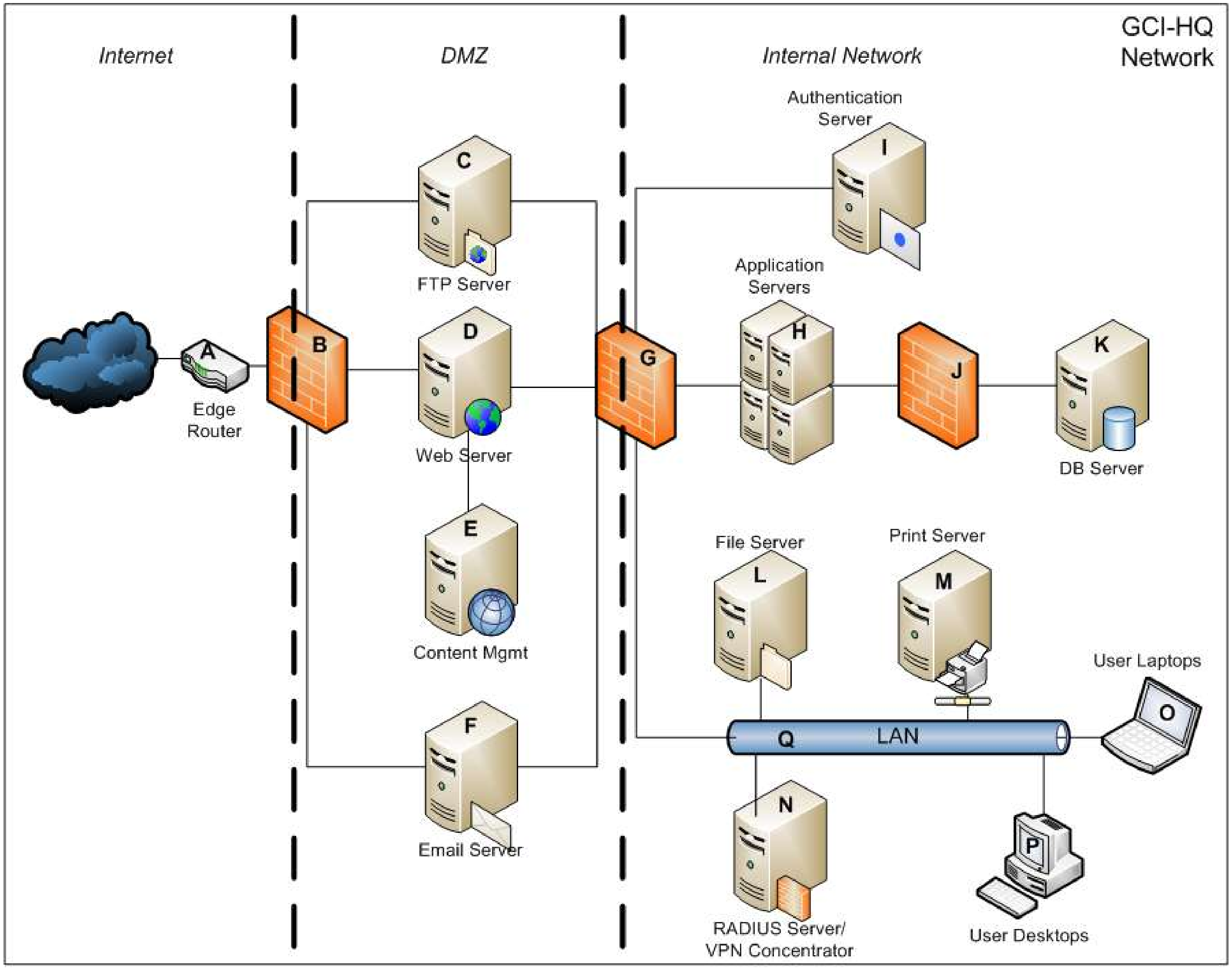
Universitetet i Oslo

Laszlo Erdödi

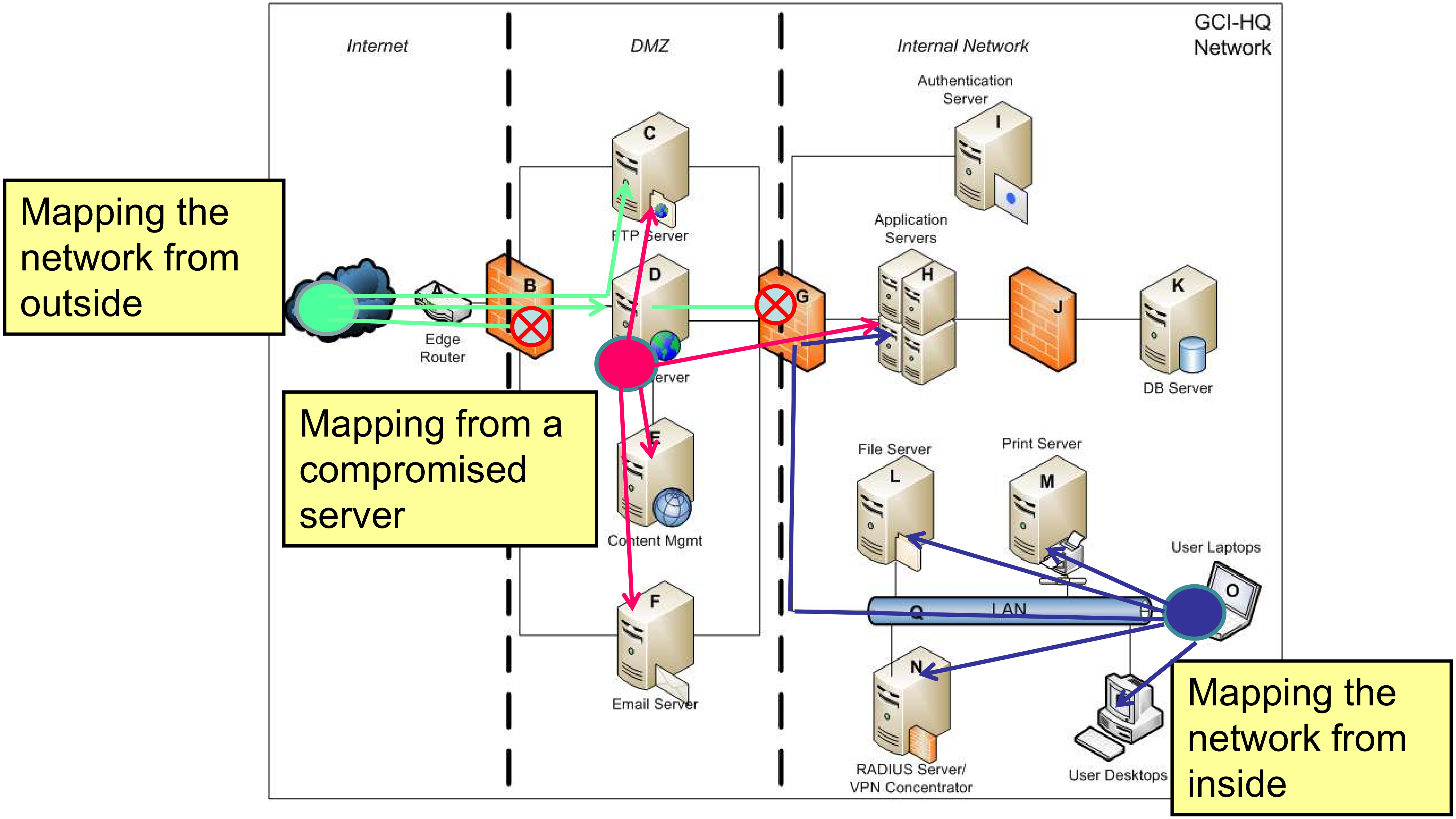
Lecture Overview

- Identifying hosts in a network
- Identifying services on a host
- What are the typical services
- Ordinary and special port scanning methods

Network layout example



Network scanning positions



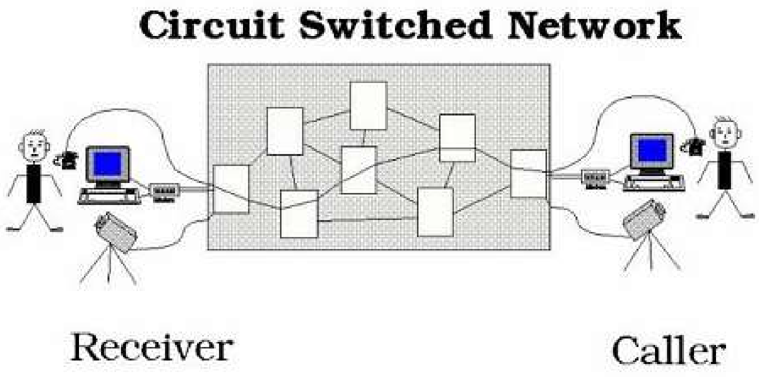
Mapping the network from outside

Mapping from a compromised server

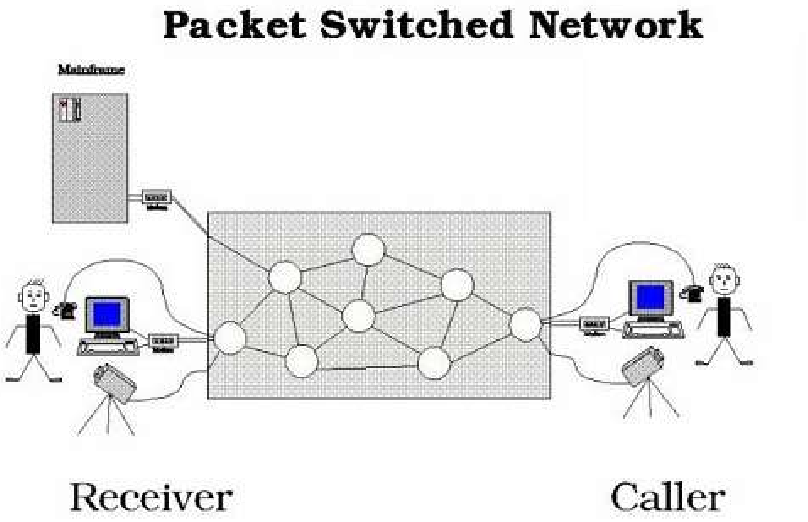
Mapping the network from inside

Circuit switched vs Packet switched networks

In circuit switched networks a virtual line is allocated between the communicating parties. The line is busy until the communication ends.

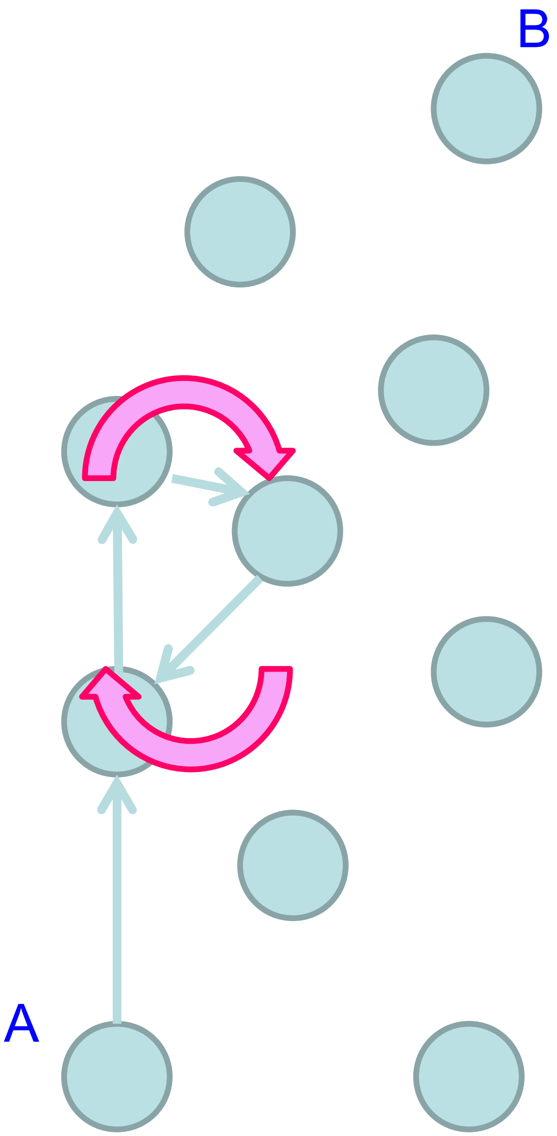


In packet switched networks the caller sends packets to the direction of the receiver. There’s no planned route, each network device chooses the most appropriate device as next considering routing tables and traffic.

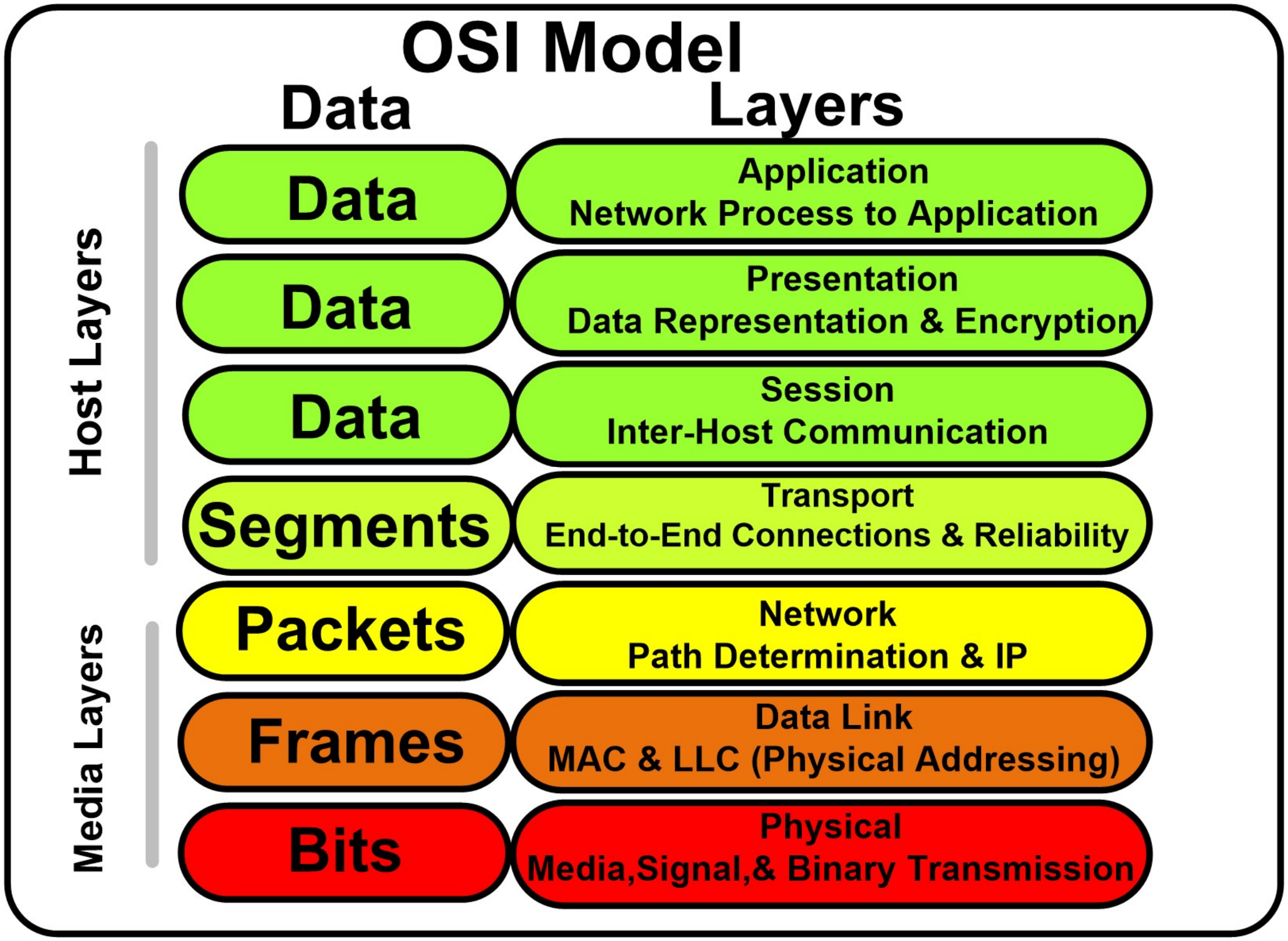


Packet switched networks – avoiding infinite loops

- As there's no planned route between the sender and the receiver it can happen that a packet gets stuck in the network following an infinite loop
- Messages are placed in network packets according to the *OSI* model
- Every packet should contain a *ttl* value (*Time to Live*) that is decreasing when arriving to the next network device (network hop)
- When *ttl* is 1 the packet has to be dropped



The OSI modell

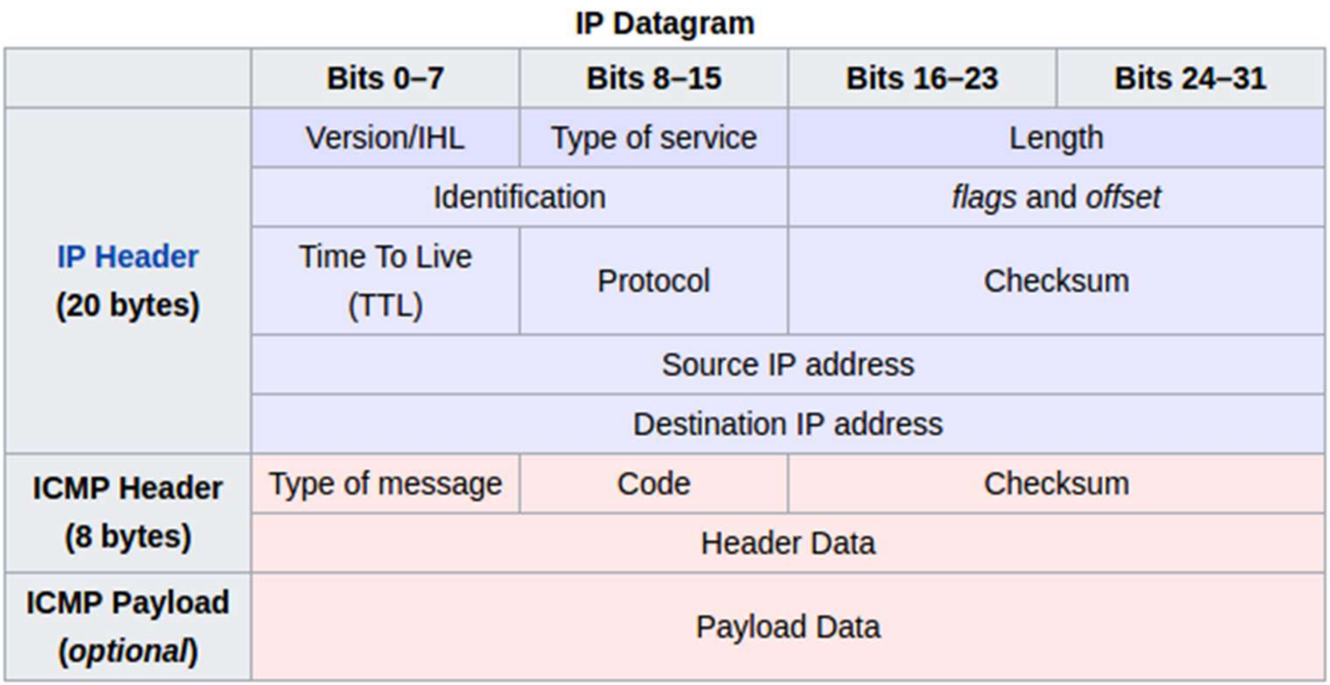


OSI Model

	Data	Layers
Host Layers	Data	Application Network Process to Application
	Data	Presentation Data Representation & Encryption
	Data	Session Inter-Host Communication
	Segments	Transport End-to-End Connections & Reliability
Media Layers	Packets	Network Path Determination & IP
	Frames	Data Link MAC & LLC (Physical Addressing)
	Bits	Physical Media,Signal,& Binary Transmission

http://electricala2z.com/cloud-computing/osi-model-layers-7-layers-osi-model/

Layer 3 – Internet Control Message Protocol (ICMP)



IP Datagram

	Bits 0–7	**Bits 8–15**	**Bits 16–23**	**Bits 24–31**
IP Header (20 bytes)	Version/IHL	Type of service	Length	
	Identification		*flags* and *offset*	
	Time To Live (TTL)	Protocol	Checksum	
	Source IP address			
	Destination IP address			
ICMP Header (8 bytes)	Type of message	Code	Checksum	
	Header Data			
ICMP Payload (*optional*)	Payload Data			

- To check if a host is responding
- *Echo request* – *Echo reply* to make sure a host is turned on

Network mapping - answer options

- **Positive answer**

  In case of *icmp* we get an echo reply for our echo request

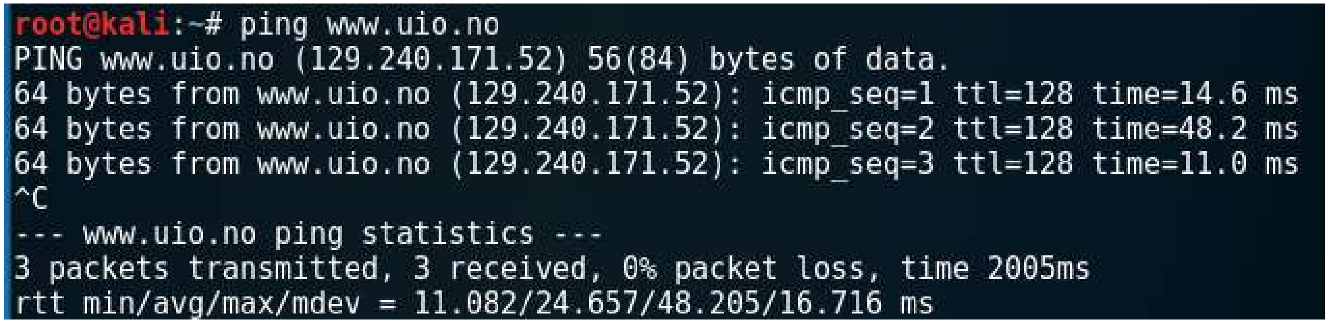
- **Negative answer**

  In case of *icmp* we get destination unreachable / host unreachable message

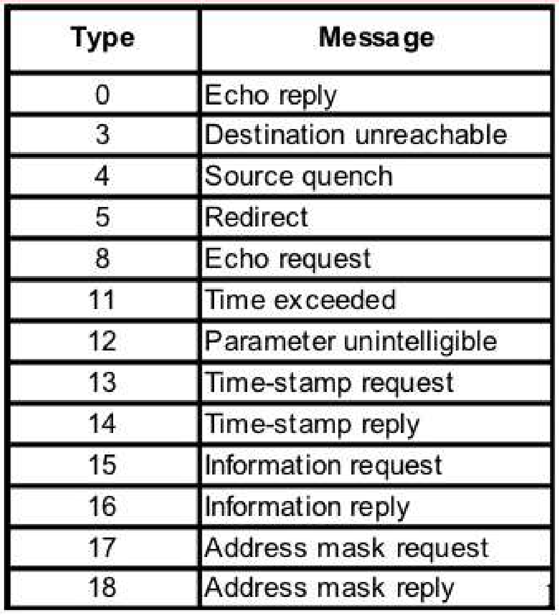
- **No answer**

  In case of *icmp*, we have no response from the host that was addressed by the echo request

Internet Control Message Protocol (ICMP) examples - ping



```
root@kali:~# ping www.uio.no
PING www.uio.no (129.240.171.52) 56(84) bytes of data.
64 bytes from www.uio.no (129.240.171.52): icmp_seq=1 ttl=128 time=14.6 ms
64 bytes from www.uio.no (129.240.171.52): icmp_seq=2 ttl=128 time=48.2 ms
64 bytes from www.uio.no (129.240.171.52): icmp_seq=3 ttl=128 time=11.0 ms
^C
--- www.uio.no ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2005ms
rtt min/avg/max/mdev = 11.082/24.657/48.205/16.716 ms
```



Type	Message
0	Echo reply
3	Destination unreachable
4	Source quench
5	Redirect
8	Echo request
11	Time exceeded
12	Parameter unintelligible
13	Time-stamp request
14	Time-stamp reply
15	Information request
16	Information reply
17	Address mask request
18	Address mask reply

https://www.slideshare.net/asimnawaz54/internet-control-message-protocol

Layer 3 – Internet Control Message Protocol (ICMP)

Since ICMP contains the *ttl* value, it is possible to guess the receiver host's operating system by its *ttl*.

Initial *ttl* values:

- Windows: 128 since Windows2000
- Linux: 64 for 2.0.x kernel
- Solaris: 255

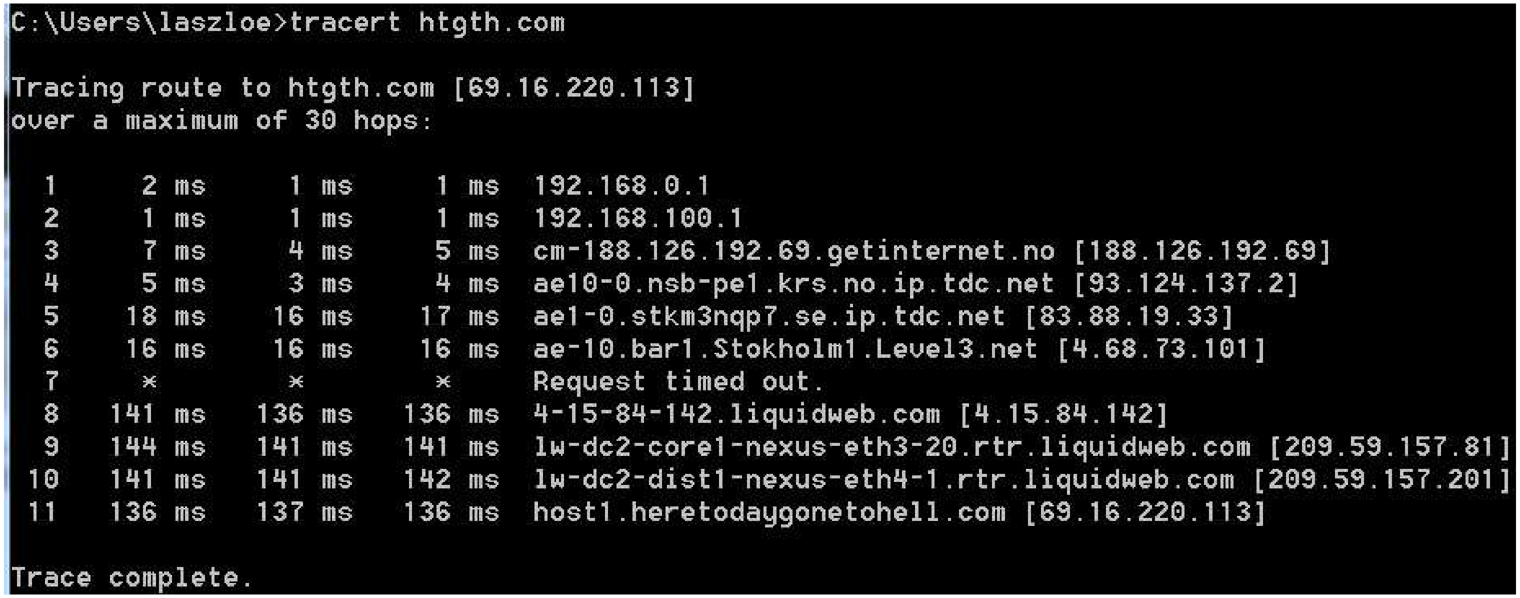
Detailed list at *Subin's Blog*: https://subinsb.com/default-device-ttl-values/

ICMP practice examples:

- Find a host with 64 as initial *ttl*
- Find a host with 128 as initial *ttl*

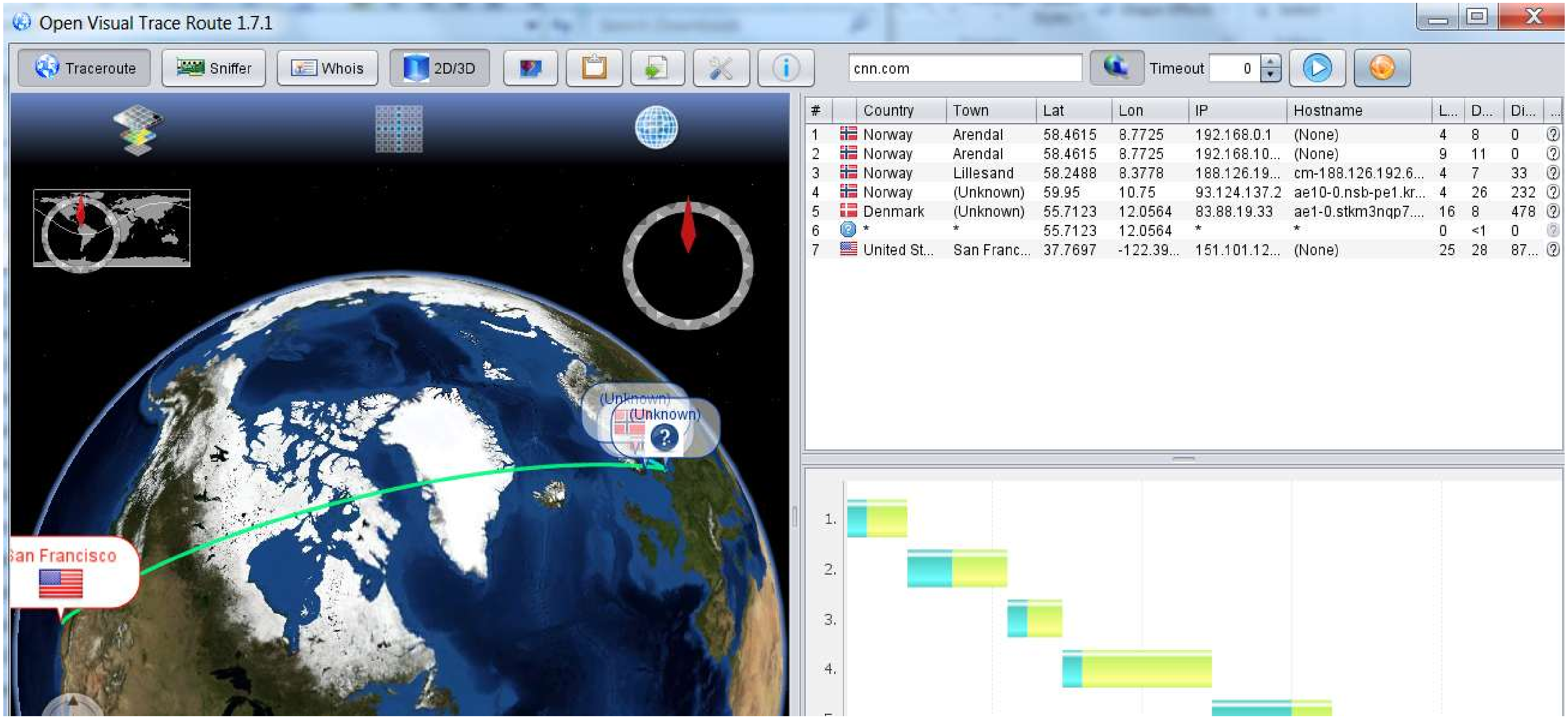
Internet Control Message Protocol (ICMP) examples - traceroute

Since all devices have to drop the packets with *ttl*=1, it is possible to map the route of a packet by repeating the ping with increasing *ttl* values. First, the initial *ttl* is 2, so after the first hop the device sends a time exceeded message. With *ttl*=3 the time exceed message is coming from the device at the second hop, etc.



```
C:\Users\laszloe>tracert htgth.com

Tracing route to htgth.com [69.16.220.113]
over a maximum of 30 hops:

  1     2 ms     1 ms     1 ms  192.168.0.1
  2     1 ms     1 ms     1 ms  192.168.100.1
  3     7 ms     4 ms     5 ms  cm-188.126.192.69.getinternet.no [188.126.192.69]
  4     5 ms     3 ms     4 ms  ae10-0.nsb-pe1.krs.no.ip.tdc.net [93.124.137.2]
  5    18 ms    16 ms    17 ms  ae1-0.stkm3nqp7.se.ip.tdc.net [83.88.19.33]
  6    16 ms    16 ms    16 ms  ae-10.bar1.Stokholm1.Level3.net [4.68.73.101]
  7     *        *        *     Request timed out.
  8   141 ms   136 ms   136 ms  4-15-84-142.liquidweb.com [4.15.84.142]
  9   144 ms   141 ms   141 ms  lw-dc2-core1-nexus-eth3-20.rtr.liquidweb.com [209.59.157.81]
 10   141 ms   141 ms   142 ms  lw-dc2-dist1-nexus-eth4-1.rtr.liquidweb.com [209.59.157.201]
 11   136 ms   137 ms   136 ms  host1.heretodaygonetohell.com [69.16.220.113]

Trace complete.
```

Internet Control Message Protocol (ICMP) examples – visual traceroute



cnn.com

Timeout 0

#		Country	Town	Lat	Lon	IP	Hostname	L...	D...	Di...	...
1		Norway	Arendal	58.4615	8.7725	192.168.0.1	(None)	4	8	0	?
2		Norway	Arendal	58.4615	8.7725	192.168.10...	(None)	9	11	0	?
3		Norway	Lillesand	58.2488	8.3778	188.126.19...	cm-188.126.192.6...	4	7	33	?
4		Norway	(Unknown)	59.95	10.75	93.124.137.2	ae10-0.nsb-pe1.kr...	4	26	232	?
5		Denmark	(Unknown)	55.7123	12.0564	83.88.19.33	ae1-0.stkm3nqp7....	16	8	478	?
6	?	*	*	55.7123	12.0564	*	*	0	<1	0	?
7		United St...	San Franc...	37.7697	-122.39...	151.101.12...	(None)	25	28	87...	?

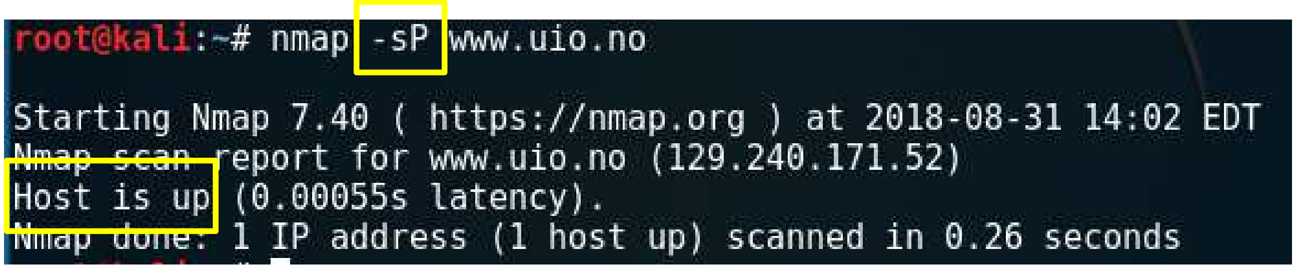
Nmap basic usage

Nmap is an universal port scanner

It is able to carry out ordinary and specific host and service discoveries

Nmap has a scripting engine which makes it capable of carrying out complex scanning as well as vulnerability discovery, fuzzing, etc. tasks

For one simple ping the following command has to be used:



```
root@kali:~# nmap -sP www.uio.no

Starting Nmap 7.40 ( https://nmap.org ) at 2018-08-31 14:02 EDT
Nmap scan report for www.uio.no (129.240.171.52)
Host is up (0.00055s latency).
Nmap done: 1 IP address (1 host up) scanned in 0.26 seconds
```

Nmap basic usage

Host(s) to be scanned can be set in multiple ways:

With domain: www.uio.no

With *ip*: 129.240.171.52

With *ip* range (CIDR): 129.240.171.0/24

With *ip* range (from-to) 129.240.171.2-6, 129.240.170-175.1

With list: 129.240.171.1,129.240.171.2

The main parameter is the scanning type that can be set with the –*s* switch, e.g. -*sP*: ping scan

Example task: How many hosts are alive in our current local network range? E.g. *nmap –sP 192.168.0.0/24*

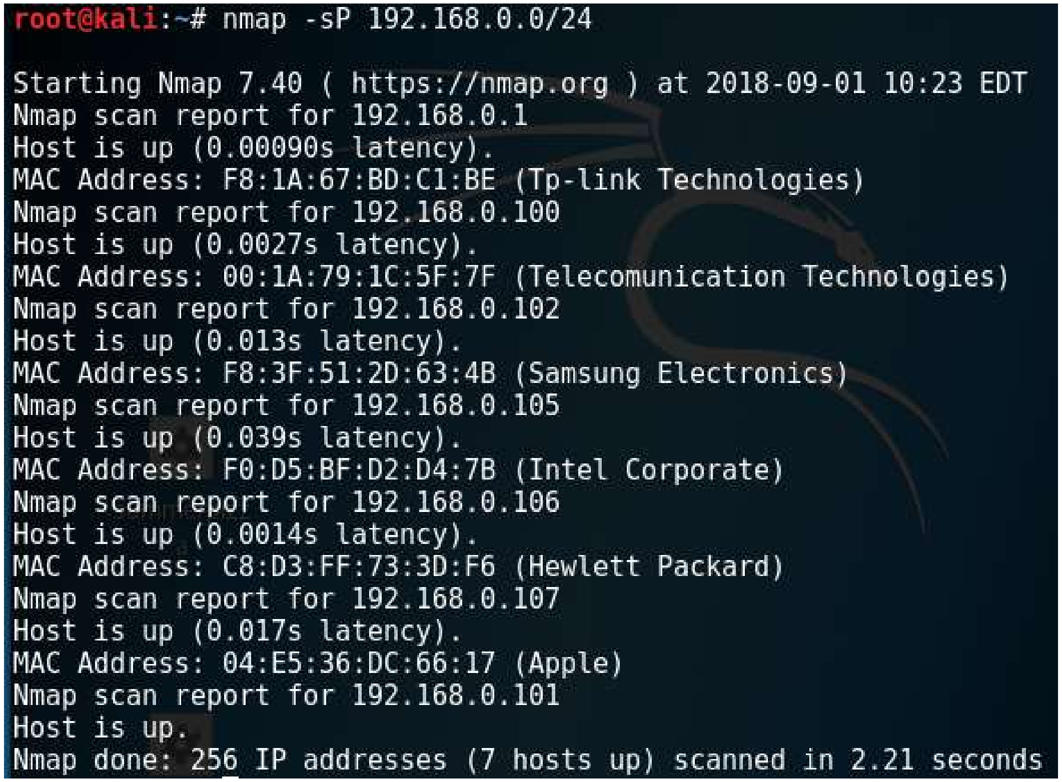
Nmap basic usage

With *nmap* it can be set:

- Type of scan (see detailed list later)
- Additional tests (e.g. version detection)
- Timing option (how many tries, how many parallel requests, max retries, scan delay, etc.)
- Hosts / host input
- Output result format (flat file, *xml*, etc.)
- Filtering (e.g. show only open ports)
- Scripts to run

Nmap - ping scan

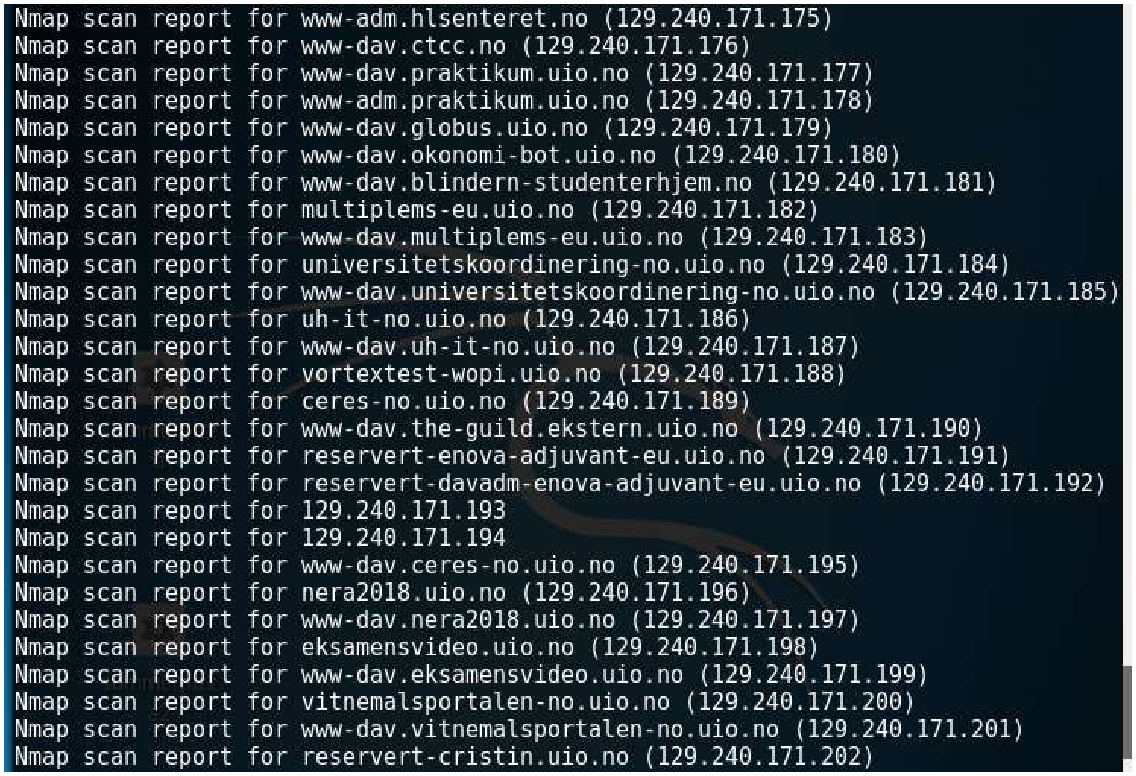
- With the –*sP* switch
- *Nmap* pings all the specified hosts
- The available hosts are listed with their *MAC* address
- *ICMP* messages are not always allowed in a network



```
root@kali:~# nmap -sP 192.168.0.0/24

Starting Nmap 7.40 ( https://nmap.org ) at 2018-09-01 10:23 EDT
Nmap scan report for 192.168.0.1
Host is up (0.00090s latency).
MAC Address: F8:1A:67:BD:C1:BE (Tp-link Technologies)
Nmap scan report for 192.168.0.100
Host is up (0.0027s latency).
MAC Address: 00:1A:79:1C:5F:7F (Telecomunication Technologies)
Nmap scan report for 192.168.0.102
Host is up (0.013s latency).
MAC Address: F8:3F:51:2D:63:4B (Samsung Electronics)
Nmap scan report for 192.168.0.105
Host is up (0.039s latency).
MAC Address: F0:D5:BF:D2:D4:7B (Intel Corporate)
Nmap scan report for 192.168.0.106
Host is up (0.0014s latency).
MAC Address: C8:D3:FF:73:3D:F6 (Hewlett Packard)
Nmap scan report for 192.168.0.107
Host is up (0.017s latency).
MAC Address: 04:E5:36:DC:66:17 (Apple)
Nmap scan report for 192.168.0.101
Host is up.
Nmap done: 256 IP addresses (7 hosts up) scanned in 2.21 seconds
```

Nmap - List scan

- With the –*sL* switch
- Has no connection with the hosts
- The *DNS* server is asked if a specific domain is registered in its database



```
Nmap scan report for www-adm.hlsenteret.no (129.240.171.175)
Nmap scan report for www-dav.ctcc.no (129.240.171.176)
Nmap scan report for www-dav.praktikum.uio.no (129.240.171.177)
Nmap scan report for www-adm.praktikum.uio.no (129.240.171.178)
Nmap scan report for www-dav.globus.uio.no (129.240.171.179)
Nmap scan report for www-dav.okonomi-bot.uio.no (129.240.171.180)
Nmap scan report for www-dav.blindern-studenterhjem.no (129.240.171.181)
Nmap scan report for multiplems-eu.uio.no (129.240.171.182)
Nmap scan report for www-dav.multiplems-eu.uio.no (129.240.171.183)
Nmap scan report for universitetskoordinering-no.uio.no (129.240.171.184)
Nmap scan report for www-dav.universitetskoordinering-no.uio.no (129.240.171.185)
Nmap scan report for uh-it-no.uio.no (129.240.171.186)
Nmap scan report for www-dav.uh-it-no.uio.no (129.240.171.187)
Nmap scan report for vortextest-wopi.uio.no (129.240.171.188)
Nmap scan report for ceres-no.uio.no (129.240.171.189)
Nmap scan report for www-dav.the-guild.ekstern.uio.no (129.240.171.190)
Nmap scan report for reservert-enova-adjuvant-eu.uio.no (129.240.171.191)
Nmap scan report for reservert-davadm-enova-adjuvant-eu.uio.no (129.240.171.192)
Nmap scan report for 129.240.171.193
Nmap scan report for 129.240.171.194
Nmap scan report for www-dav.ceres-no.uio.no (129.240.171.195)
Nmap scan report for nera2018.uio.no (129.240.171.196)
Nmap scan report for www-dav.nera2018.uio.no (129.240.171.197)
Nmap scan report for eksamensvideo.uio.no (129.240.171.198)
Nmap scan report for www-dav.eksamensvideo.uio.no (129.240.171.199)
Nmap scan report for vitnemalsportalen-no.uio.no (129.240.171.200)
Nmap scan report for www-dav.vitnemalsportalen-no.uio.no (129.240.171.201)
Nmap scan report for reservert-cristin.uio.no (129.240.171.202)
```

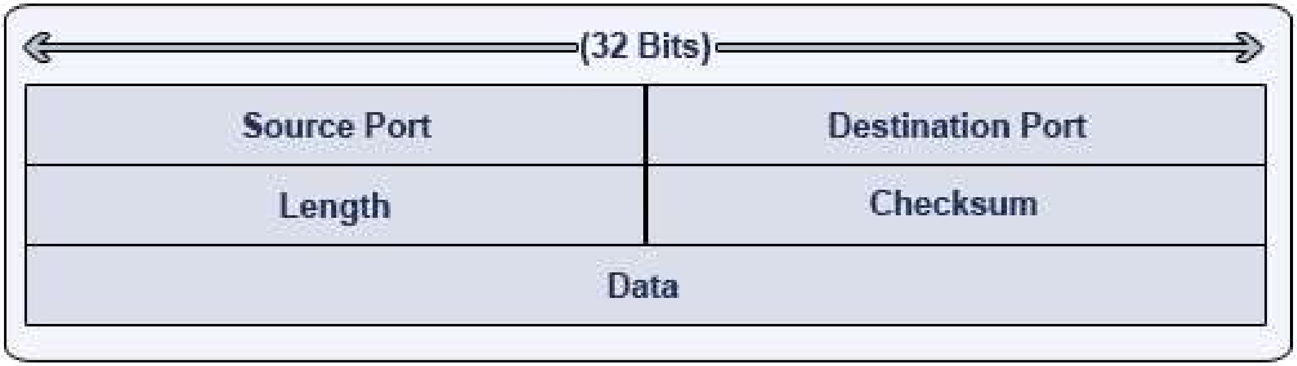
Layer 4 – Data transmission

Apart from sending short simple messages, bigger data blocks can be transmitted between the hosts. The data transfer is carried out in the 4th layer by using 2 different approaches:

- *UDP*: streaming the data (no guarantee that all data will arrive, but fast)
- *TCP*: the arrival of all data is guaranteed in the right order (trustworthy transmission, slower than *UDP*)

In addition, the data transmission is carried out using port numbers. One host can send and receive data in multiple channels using different port numbers for different services.

Layer 4 – UDP protocol



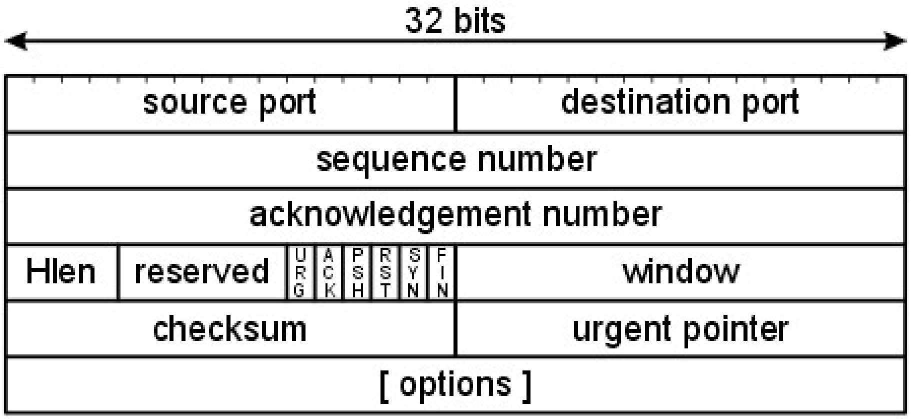
(32 Bits)

Source Port	Destination Port
Length	Checksum
Data	

The port number is a 2-byte value, it can be between 0-65535(=2^{32})
Typical *UDP* ports with services:

- *UDP* 53 *DNS*
- *UDP* 111 *RPC* (Remote Procedure Call)
- *UDP* 123 *NTP* (Network Time Protocol)

Layer 4 – TCP protocol



In order to ensure that the packages arrived in the right order the sequence number and the acknowledgement number are used.

TCP flags are for maintaining the connection status (*urg, ack, psh, rst, syn, fin*).

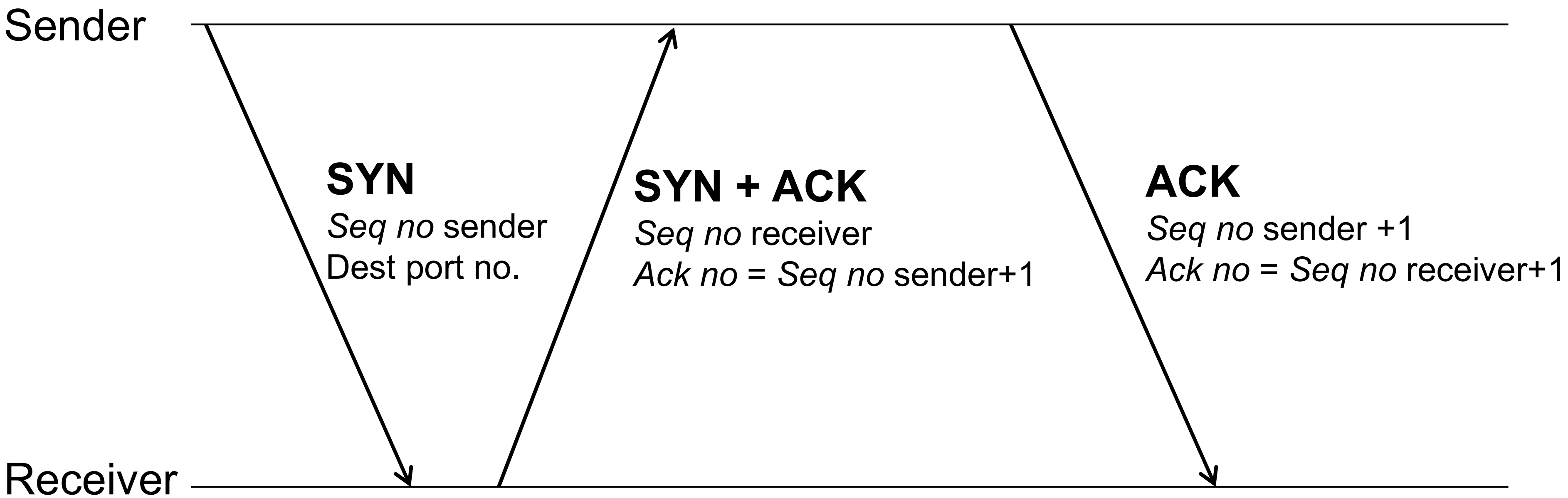
Layer 4 – TCP typical services

- *TCP* 80: *web http*
- *TCP* 443: *web https*
- *TCP* 20,21: *ftp*
- *TCP* 22: *ssh*
- *TCP* 25: *smtp*
- *TCP* 137,139,445: *netbios*
- *TCP* 3306: *mysql*
- *TCP* 3389: *remote desktop*
- *TCP* 5900: *VNC*

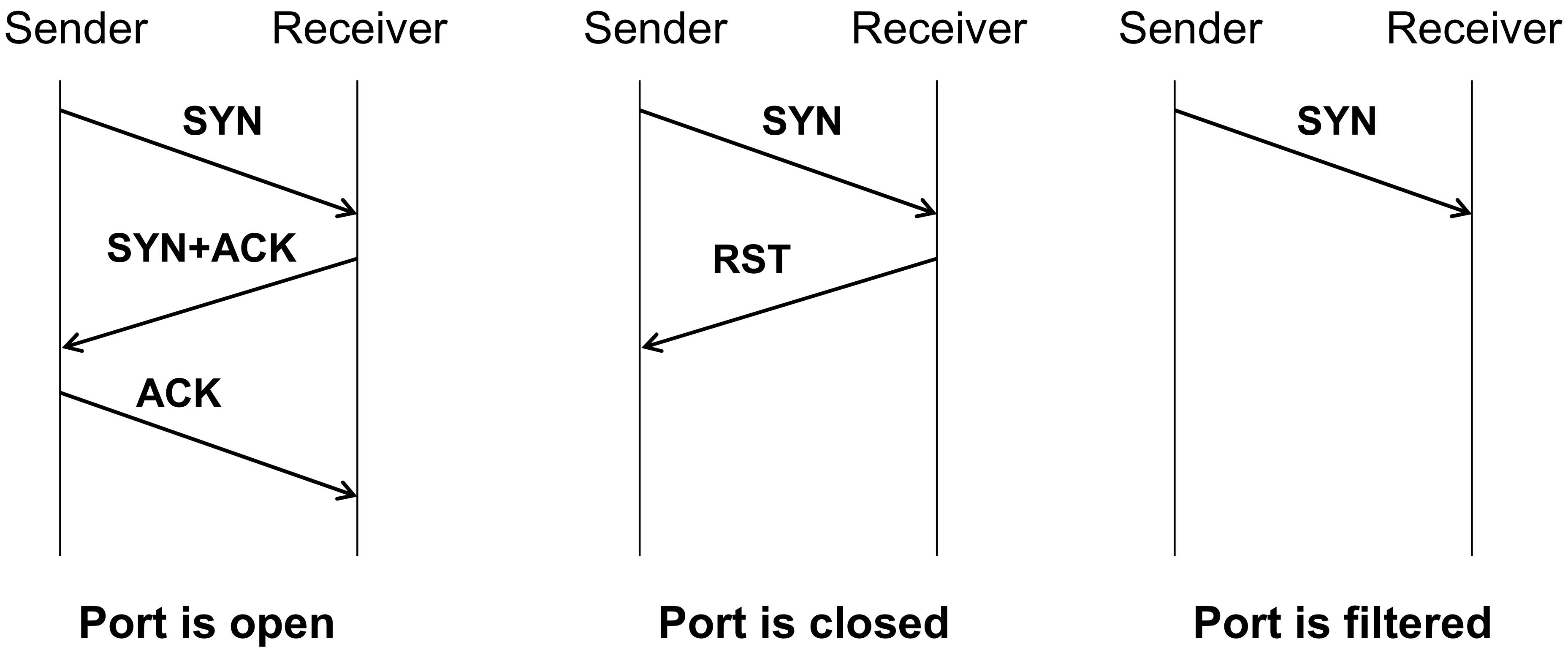
Remember that any service can be used in any port, these are only recommendations

Layer 4 – TCP 3-way handshake

TCP handshake is the process when a connection is about to be established in a specific port.



Tcp scan (full tcp scan)



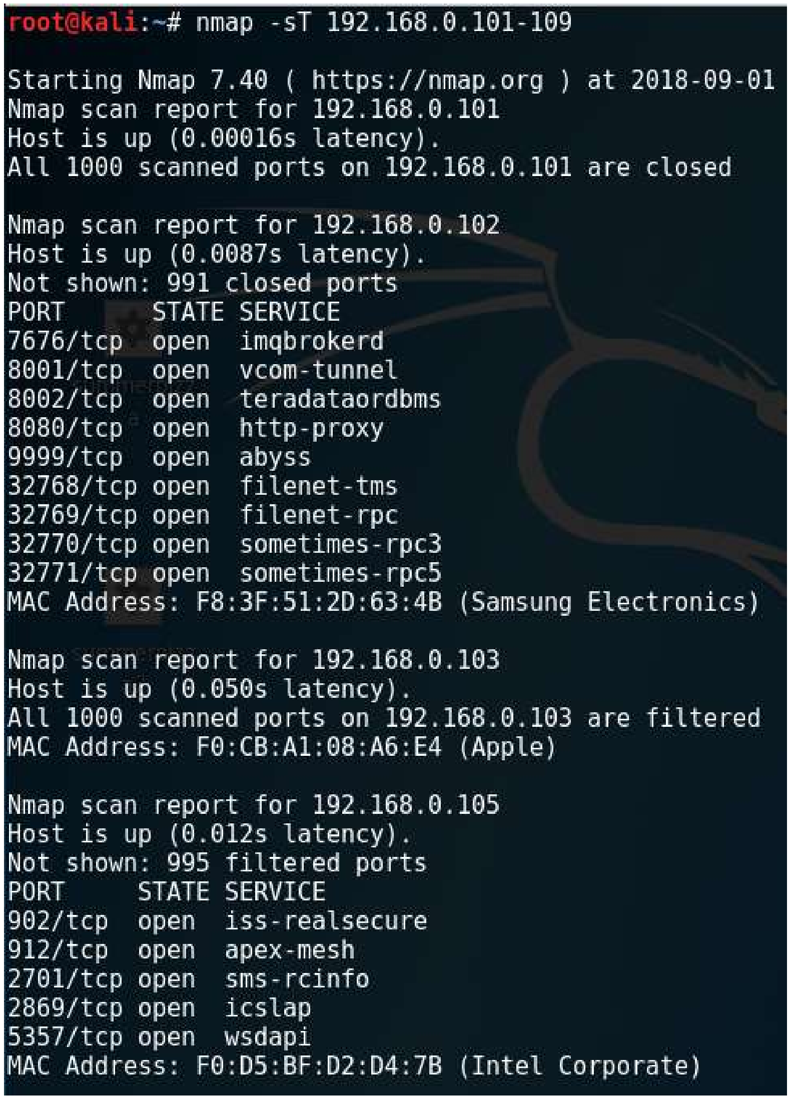
Nmap carries out *tcp* scan with the *–sT* switch
Port numbers can be specified optionally
Example: *nmap –sT –p80,43 host*

Tcp scan (full tcp scan)

The number of possible ports is 65535, scanning all ports requires too much time (and too noisy).

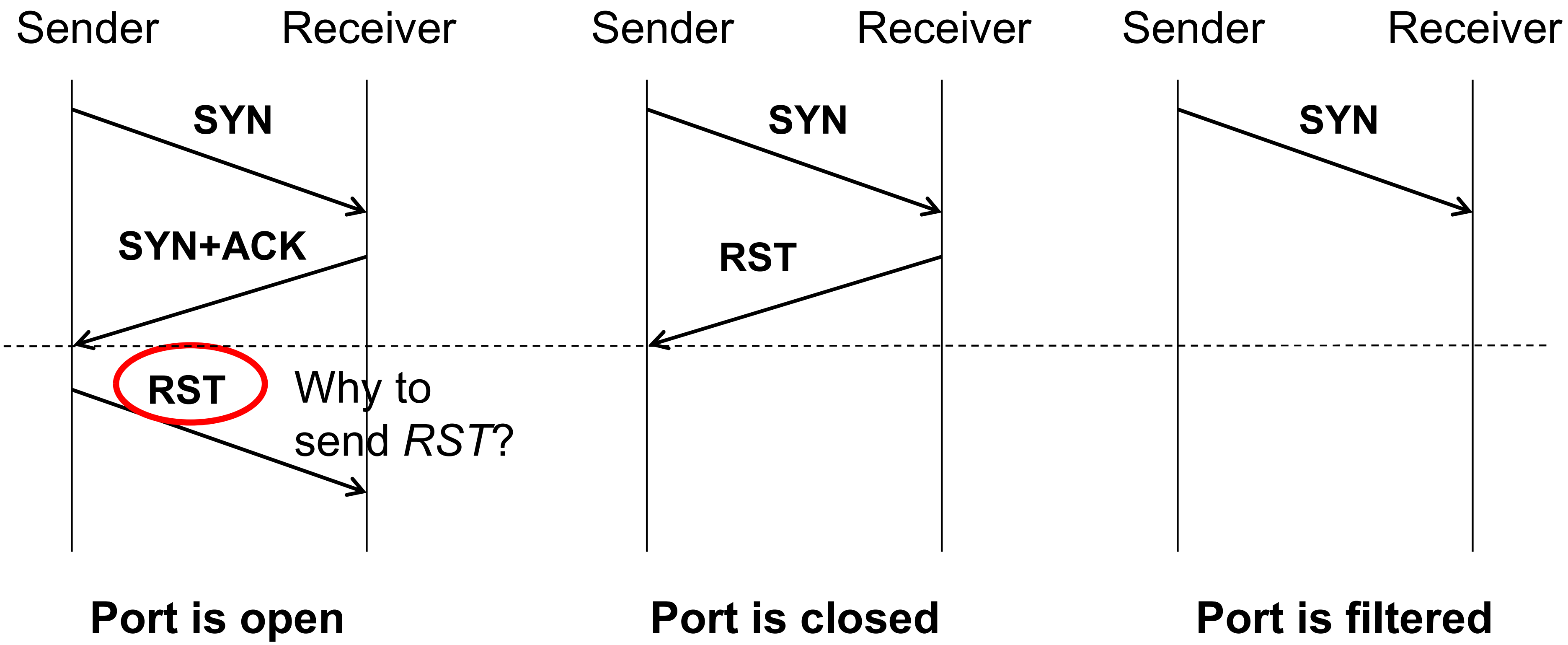
We can reduce the port numbers by specifying them with the *–p* switch.

Without *–p nmap* will scan the 1024 most popular ports.



```
root@kali:~# nmap -sT 192.168.0.101-109

Starting Nmap 7.40 ( https://nmap.org ) at 2018-09-01
Nmap scan report for 192.168.0.101
Host is up (0.00016s latency).
All 1000 scanned ports on 192.168.0.101 are closed

Nmap scan report for 192.168.0.102
Host is up (0.0087s latency).
Not shown: 991 closed ports
PORT      STATE SERVICE
7676/tcp  open  imqbrokerd
8001/tcp  open  vcom-tunnel
8002/tcp  open  teradataordbms
8080/tcp  open  http-proxy
9999/tcp  open  abyss
32768/tcp open  filenet-tms
32769/tcp open  filenet-rpc
32770/tcp open  sometimes-rpc3
32771/tcp open  sometimes-rpc5
MAC Address: F8:3F:51:2D:63:4B (Samsung Electronics)

Nmap scan report for 192.168.0.103
Host is up (0.050s latency).
All 1000 scanned ports on 192.168.0.103 are filtered
MAC Address: F0:CB:A1:08:A6:E4 (Apple)

Nmap scan report for 192.168.0.105
Host is up (0.012s latency).
Not shown: 995 filtered ports
PORT     STATE SERVICE
902/tcp  open  iss-realsecure
912/tcp  open  apex-mesh
2701/tcp open  sms-rcinfo
2869/tcp open  icslap
5357/tcp open  wsdapi
MAC Address: F0:D5:BF:D2:D4:7B (Intel Corporate)
```

SYN scan (half open scan)



Port is open

Port is closed

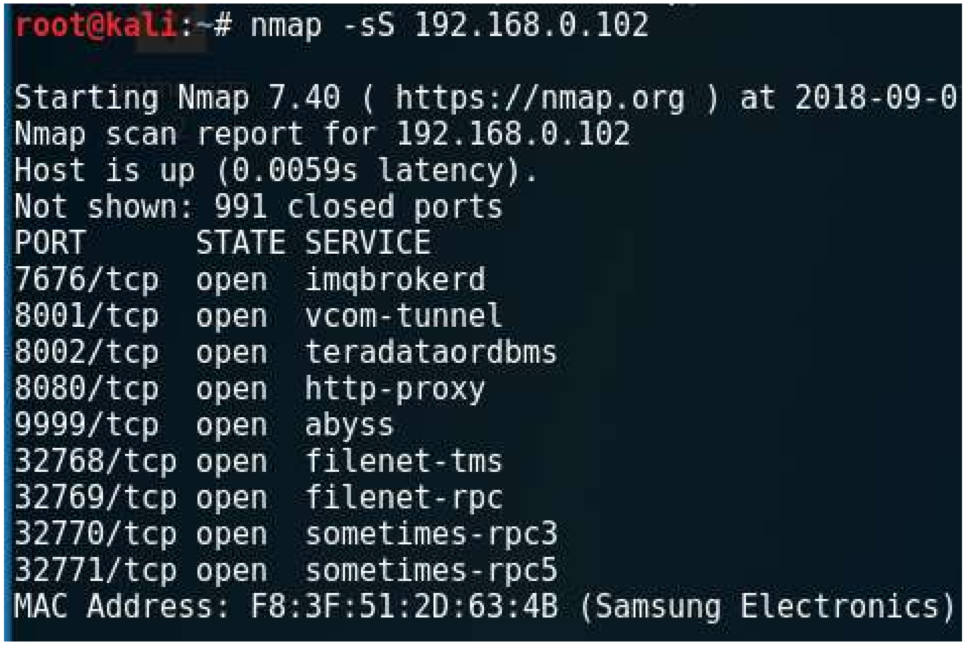
Port is filtered

Nmap carries out *syn* scan with the *–sS* switch.
Port numbers can be specified optionally.
Example: *nmap –sS –p80,43 host*

SYN scan (half open scan)

Why to use *syn* scan instead of *tcp* scan? Does it have different result?

The main difference is that in case of *tcp* scan the *tcp* connection is established for every open ports. Firewalls usually log only the established connections.



```
root@kali:~# nmap -sS 192.168.0.102

Starting Nmap 7.40 ( https://nmap.org ) at 2018-09-0
Nmap scan report for 192.168.0.102
Host is up (0.0059s latency).
Not shown: 991 closed ports
PORT      STATE SERVICE
7676/tcp  open  imqbrokerd
8001/tcp  open  vcom-tunnel
8002/tcp  open  teradataordbms
8080/tcp  open  http-proxy
9999/tcp  open  abyss
32768/tcp open  filenet-tms
32769/tcp open  filenet-rpc
32770/tcp open  sometimes-rpc3
32771/tcp open  sometimes-rpc5
MAC Address: F8:3F:51:2D:63:4B (Samsung Electronics)
```

Reverse scans

In case of reverse scanning, *Nmap* looks for closed ports. The result of a reverse scan can be either *open/filtered* or *closed*. It cannot be determined if a port is filtered or open.

According to *TCP* if a port is closed the receiver sends *rst* answer no matter which status flag is set:

-*sN* Null scan (no flags)

-*sF* Fin scan (only *fin* flag is set)

-*sX* Xmas scan (*push*, *fin* and *rst* flags are set)

-*sM* Maimon scan (*fin* and *ack* are set)

With *hping* we can set any flag (more reverse scan options, see later)

Ack scan

Ack scan is to determine if a firewall is stateful or stateless.

- The stateless firewall examines a packet as it is independent of the previous packets.
- The stateful firewall can follow packet streams considering previous packets.

For a stateless firewall an *ack* package seems like the third step of the handshake. For the stateful firewall it is pointless (no *syn* and *syn+ack* before).

nmap -sA

Decoy scan – hide ourselves

If a *TCP* connection is established it will be logged by the firewalls – this is noisy (in a network with huge internet traffic there are several port scans by robots).

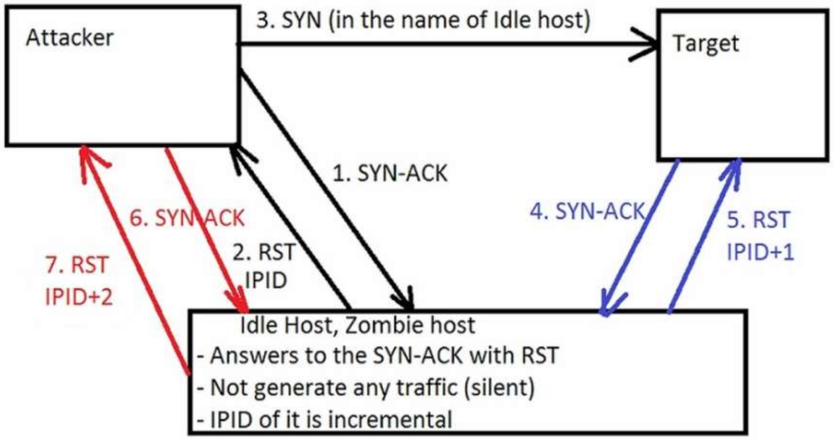
Decoy scan uses the «needle in the haystack» theory: it sends out each request in multiple copies with different source *ip*.

Questions: Can we modify our source *ip* in the packet?
If so, why don't we modify it all the time?

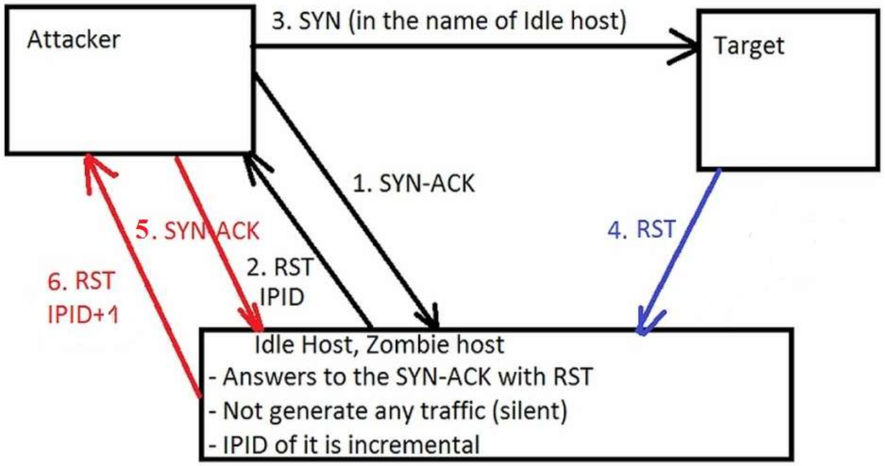
Decoy scan example: *nmap –sT –p80 –D5.44.65.150,195.88.55.16, 194.61.183.124 www.uio.no*

Idle scan, ftp bounce – hide ourselves

There are more sophisticated ways of hiding ourselves:


Attacker
3. SYN (in the name of Idle host)
Target
1. SYN-ACK
4. SYN-ACK
5. RST
IPID+1
6. SYN-ACK
2. RST
IPID
7. RST
IPID+2
Idle Host, Zombie host
- Answers to the SYN-ACK with RST
- Not generate any traffic (silent)
- IPID of it is incremental

Port is open


Attacker
3. SYN (in the name of Idle host)
Target
1. SYN-ACK
4. RST
5. SYN-ACK
2. RST
IPID
6. RST
IPID+1
Idle Host, Zombie host
- Answers to the SYN-ACK with RST
- Not generate any traffic (silent)
- IPID of it is incremental

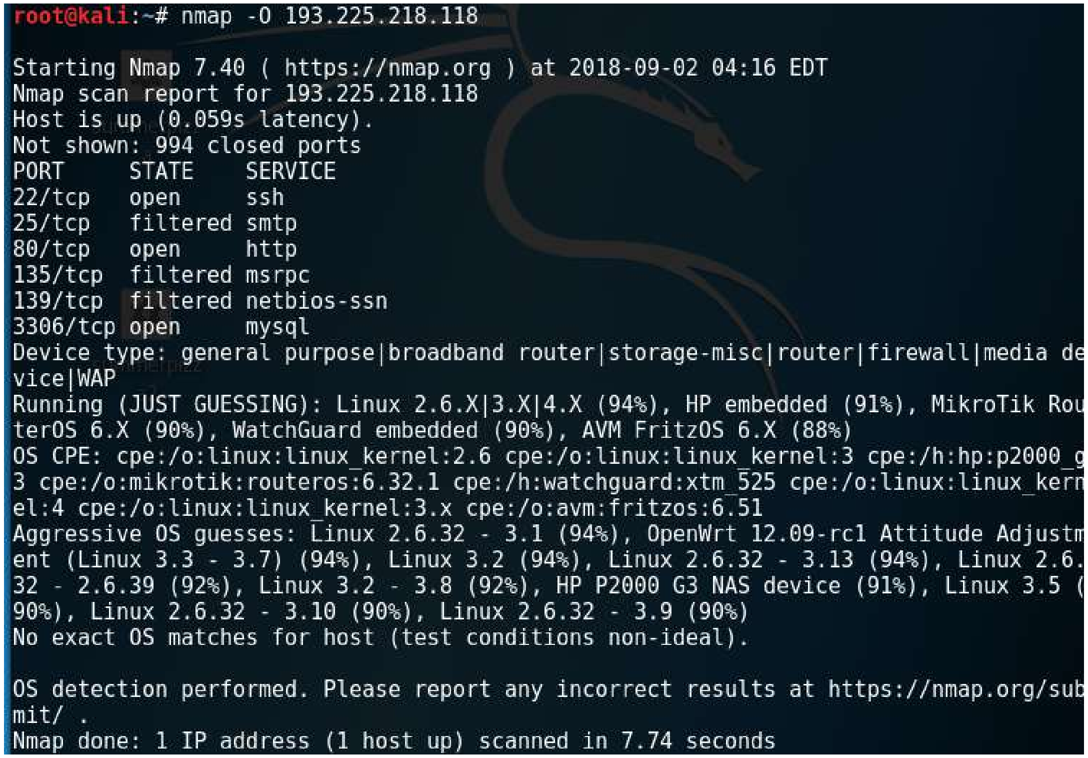
Port is closed
Port is filtered (without step 4.)

Example idle scan: *nmap –sI zombie.somewhere.com* www.uio.no
Example ftp bounce: *nmap -b user@FTP-Address Target-Address*

Operating System detection

Nmap's remote *OS* detection uses *TCP/IP* stack fingerprinting. Nmap sends a series of *TCP* and *UDP* packets to the remote host and examines practically every bit in the responses.

After performing dozens of tests such as *TCP ISN* sampling, *TCP* options support and ordering, *IP ID* sampling, and the initial window size check, *Nmap* compares the results to its *nmap-os-db* database of more than 2,600 known *OS* fingerprints and prints out the *OS* details if there is a match.

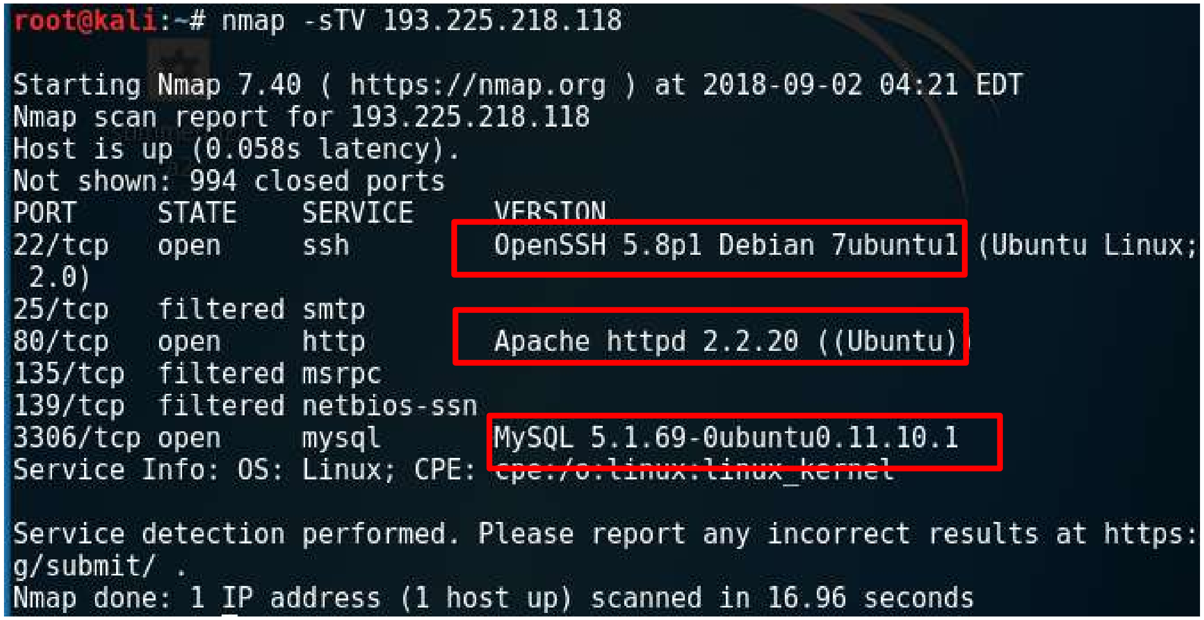


```
root@kali:~# nmap -O 193.225.218.118

Starting Nmap 7.40 ( https://nmap.org ) at 2018-09-02 04:16 EDT
Nmap scan report for 193.225.218.118
Host is up (0.059s latency).
Not shown: 994 closed ports
PORT     STATE    SERVICE
22/tcp   open     ssh
25/tcp   filtered smtp
80/tcp   open     http
135/tcp  filtered msrpc
139/tcp  filtered netbios-ssn
3306/tcp open     mysql
Device type: general purpose|broadband router|storage-misc|router|firewall|media de
vice|WAP
Running (JUST GUESSING): Linux 2.6.X|3.X|4.X (94%), HP embedded (91%), MikroTik Rou
terOS 6.X (90%), WatchGuard embedded (90%), AVM FritzOS 6.X (88%)
OS CPE: cpe:/o:linux:linux_kernel:2.6 cpe:/o:linux:linux_kernel:3 cpe:/h:hp:p2000_g
3 cpe:/o:mikrotik:routeros:6.32.1 cpe:/h:watchguard:xtm_525 cpe:/o:linux:linux_kern
el:4 cpe:/o:linux:linux_kernel:3.x cpe:/o:avm:fritzos:6.51
Aggressive OS guesses: Linux 2.6.32 - 3.1 (94%), OpenWrt 12.09-rc1 Attitude Adjustm
ent (Linux 3.3 - 3.7) (94%), Linux 3.2 (94%), Linux 2.6.32 - 3.13 (94%), Linux 2.6.
32 - 2.6.39 (92%), Linux 3.2 - 3.8 (92%), HP P2000 G3 NAS device (91%), Linux 3.5 (
90%), Linux 2.6.32 - 3.10 (90%), Linux 2.6.32 - 3.9 (90%)
No exact OS matches for host (test conditions non-ideal).

OS detection performed. Please report any incorrect results at https://nmap.org/sub
mit/ .
Nmap done: 1 IP address (1 host up) scanned in 7.74 seconds
```

Service version detection

Version detection interrogates the ports to determine more about what is actually running. The *nmap-service-probes* database contains probes for querying various services and match expressions to recognize and parse responses.

Nmap tries to determine the service protocol, the version number, hostname, device, the *OS* family. With *banner grabbing* completely exact version numbers can be retrieved (*Banner* info can be modified).



```
root@kali:~# nmap -sTV 193.225.218.118

Starting Nmap 7.40 ( https://nmap.org ) at 2018-09-02 04:21 EDT
Nmap scan report for 193.225.218.118
Host is up (0.058s latency).
Not shown: 994 closed ports
PORT     STATE    SERVICE     VERSION
22/tcp   open     ssh         OpenSSH 5.8p1 Debian 7ubuntu1 (Ubuntu Linux;
 2.0)
25/tcp   filtered smtp
80/tcp   open     http        Apache httpd 2.2.20 ((Ubuntu))
135/tcp  filtered msrpc
139/tcp  filtered netbios-ssn
3306/tcp open     mysql      MySQL 5.1.69-0ubuntu0.11.10.1
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https:
g/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 16.96 seconds
```

Hping2, hping3

Besides *nmap* there are other port scanners like the *hping* family.

- Firewall testing
- Advanced port scanning
- Network testing, using different protocols, *TOS*, fragmentation
- Manual path *MTU* discovery
- Advanced traceroute, under all the supported protocols
- Remote *OS* fingerprinting
- Remote uptime guessing
- TCP/IP stacks auditing

Hping2, hping3

Examples:

Fin scan: *hping3 -c 1 -V -p 80 -s 5050 -F 0daysecurity.com*

Smurf attack: *hping3 -1 --flood -a VICTIM_IP BROADCAST_ADDRESS*

Land attack (DOS): *hping3 -V -c 1000000 -d 120 -S -w 64 -p 445 -s 445 --flood*

- *--flood*: sent packets as fast as possible. Don't show replies.
- *-V* <-- Verbose
- *-c --count*: packet count
- *-d --data*: data size
- *-S --syn*: set SYN flag
- *-w --win*: winsize (default 64)
- *-p* --destport [+][+]<port> destination port(default 0) ctrl+z inc/dec
- *-s* --baseport: base source port (default random)

See detailed examples here:

https://www.golinuxcloud.com/hping3-command-in-linux/

Nmap scripting engine

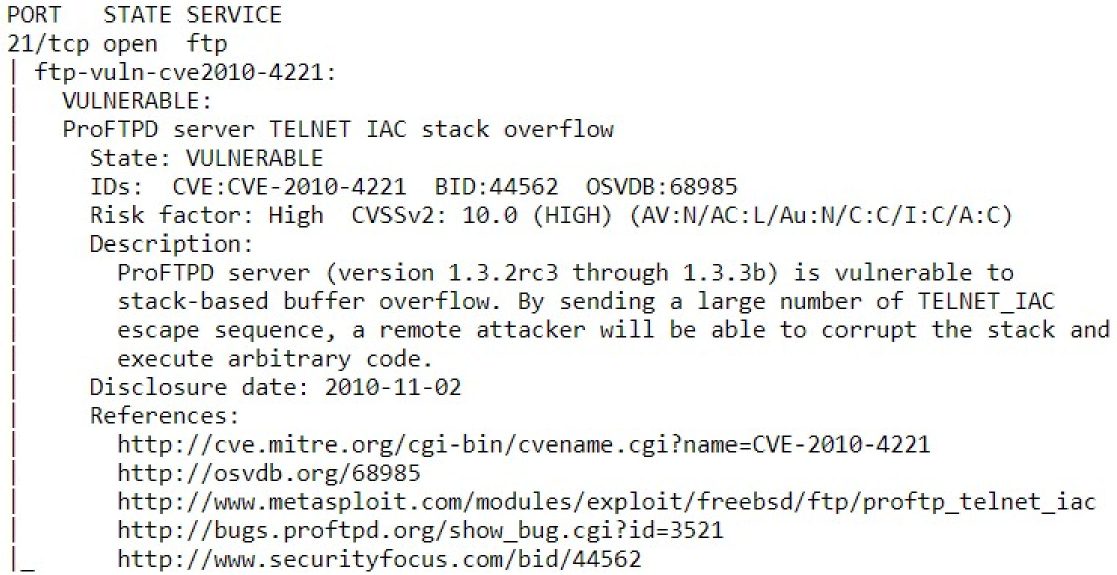
Nmap is not only a port scanner, but a lightweight vulnerability discovery tool as well. With the scripting capabilities we can specify special requests using the *lua* language. The *Nmap* database contains prewritten scripts that are put into categories:

- Auth
- Broadcast
- Brute
- Default
- Discovery
- DOS
- Exploit
- External
- Fuzzer
- Intrusive
- Malware
- Safe
- Version
- Vuln

Nmap scripting engine

Example: *nmap –sT –p21 –script==ftp-vuln-cve2010-4221 target*

Script output:



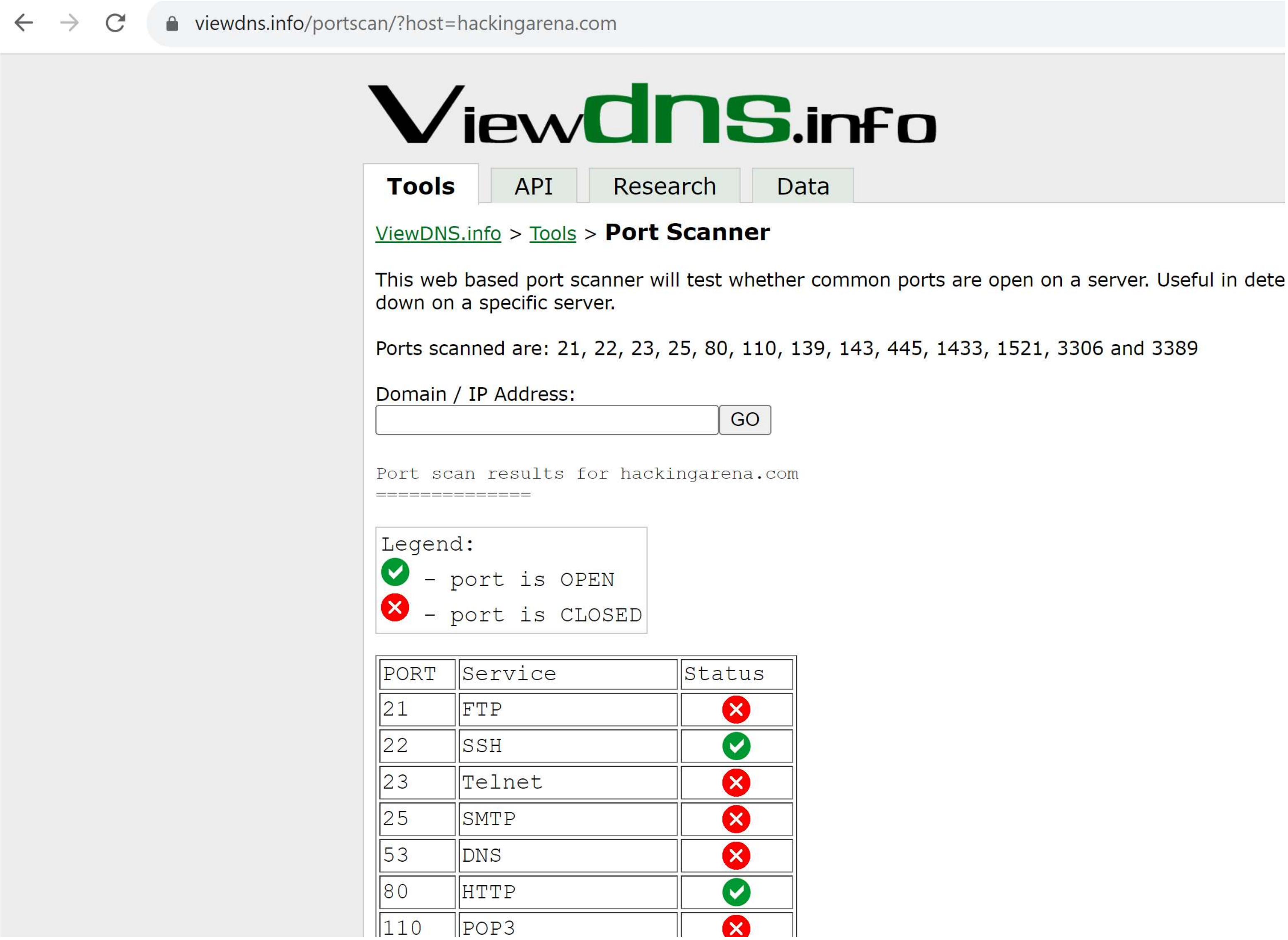
```
PORT   STATE SERVICE
21/tcp open  ftp
| ftp-vuln-cve2010-4221:
|   VULNERABLE:
|   ProFTPD server TELNET IAC stack overflow
|     State: VULNERABLE
|     IDs:  CVE:CVE-2010-4221  BID:44562  OSVDB:68985
|     Risk factor: High  CVSSv2: 10.0 (HIGH) (AV:N/AC:L/Au:N/C:C/I:C/A:C)
|     Description:
|       ProFTPD server (version 1.3.2rc3 through 1.3.3b) is vulnerable to
|       stack-based buffer overflow. By sending a large number of TELNET_IAC
|       escape sequence, a remote attacker will be able to corrupt the stack and
|       execute arbitrary code.
|     Disclosure date: 2010-11-02
|     References:
|       http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2010-4221
|       http://osvdb.org/68985
|       http://www.metasploit.com/modules/exploit/freebsd/ftp/proftp_telnet_iac
|       http://bugs.proftpd.org/show_bug.cgi?id=3521
|_      http://www.securityfocus.com/bid/44562
```

Other examples:

All scripts from a category: *nmap –sT –p21 –script==vuln target*

All scripts (carpet bombing!): *nmap –sT –p21 –script==all target*

Online port scanning (viewdns.info)



ViewDNS.info > Tools > **Port Scanner**

This web based port scanner will test whether common ports are open on a server. Useful in dete down on a specific server.

Ports scanned are: 21, 22, 23, 25, 80, 110, 139, 143, 445, 1433, 1521, 3306 and 3389

Domain / IP Address:

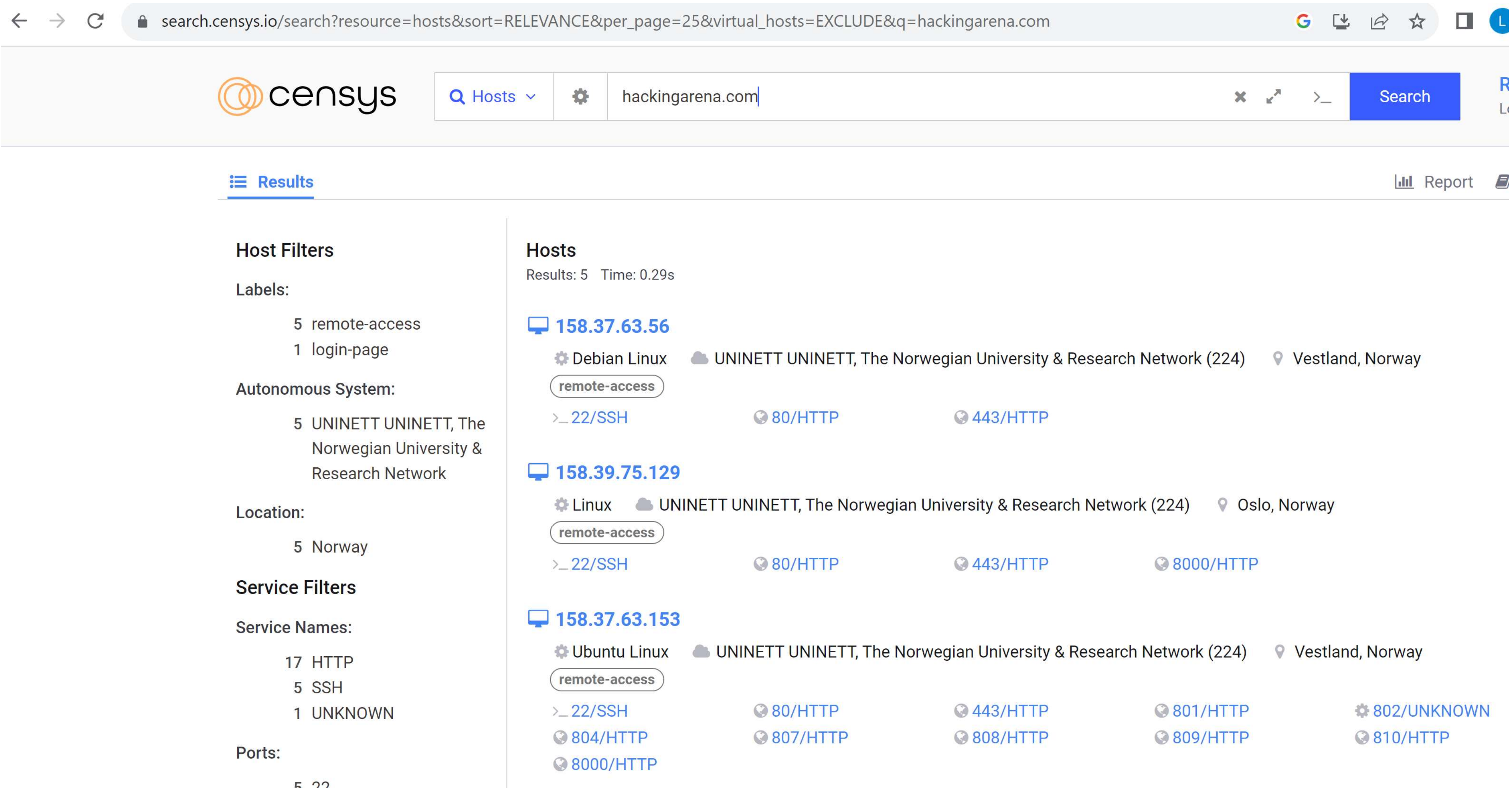
GO

```
Port scan results for hackingarena.com
==============
```

Legend:
✓ - port is OPEN
✗ - port is CLOSED

PORT	Service	Status
21	FTP	✗
22	SSH	✓
23	Telnet	✗
25	SMTP	✗
53	DNS	✗
80	HTTP	✓
110	POP3	✗

Online port scanning (censys.io)



search.censys.io/search?resource=hosts&sort=RELEVANCE&per_page=25&virtual_hosts=EXCLUDE&q=hackingarena.com

hackingarena.com

Host Filters

Labels:

5 remote-access
1 login-page

Autonomous System:

5 UNINETT UNINETT, The Norwegian University & Research Network

Location:

5 Norway

Service Filters

Service Names:

17 HTTP
5 SSH
1 UNKNOWN

Ports:

5 22

Hosts

Results: 5 Time: 0.29s

158.37.63.56

Debian Linux UNINETT UNINETT, The Norwegian University & Research Network (224) Vestland, Norway

remote-access

22/SSH 80/HTTP 443/HTTP

158.39.75.129

Linux UNINETT UNINETT, The Norwegian University & Research Network (224) Oslo, Norway

remote-access

22/SSH 80/HTTP 443/HTTP 8000/HTTP

158.37.63.153

Ubuntu Linux UNINETT UNINETT, The Norwegian University & Research Network (224) Vestland, Norway

remote-access

22/SSH 80/HTTP 443/HTTP 801/HTTP 802/UNKNOWN
804/HTTP 807/HTTP 808/HTTP 809/HTTP 810/HTTP
8000/HTTP

Port scanning summary: inventory

- The result of the port scanning has to be summarized in a table (Inventory)
- The inventory should be part of the final pentest report
- The table contains all the discovered hosts with all discovered services in separate rows
- Each service has a comment field if it was compromised during the pentest
- The client can evaluate each service if it should be closed or assign a responsible person for all operating services

Special port scanners: Firewalk, Zmap

Firewalk was a special internal network scanner in the beginning of the 2000s (cannot be used today). It was able to exploit of a flow of the *TCP* implementation and scan the internal network with one hop behind a firewall (it used customized *ttl* values).

Zmap is a superfast layer2 port scanner. It is able to map the whole *ipv4* network range within 45 minutes for one port. (*https://zmap.io/*)

End of lecture