ICT2203/ICT2203X

Network Security

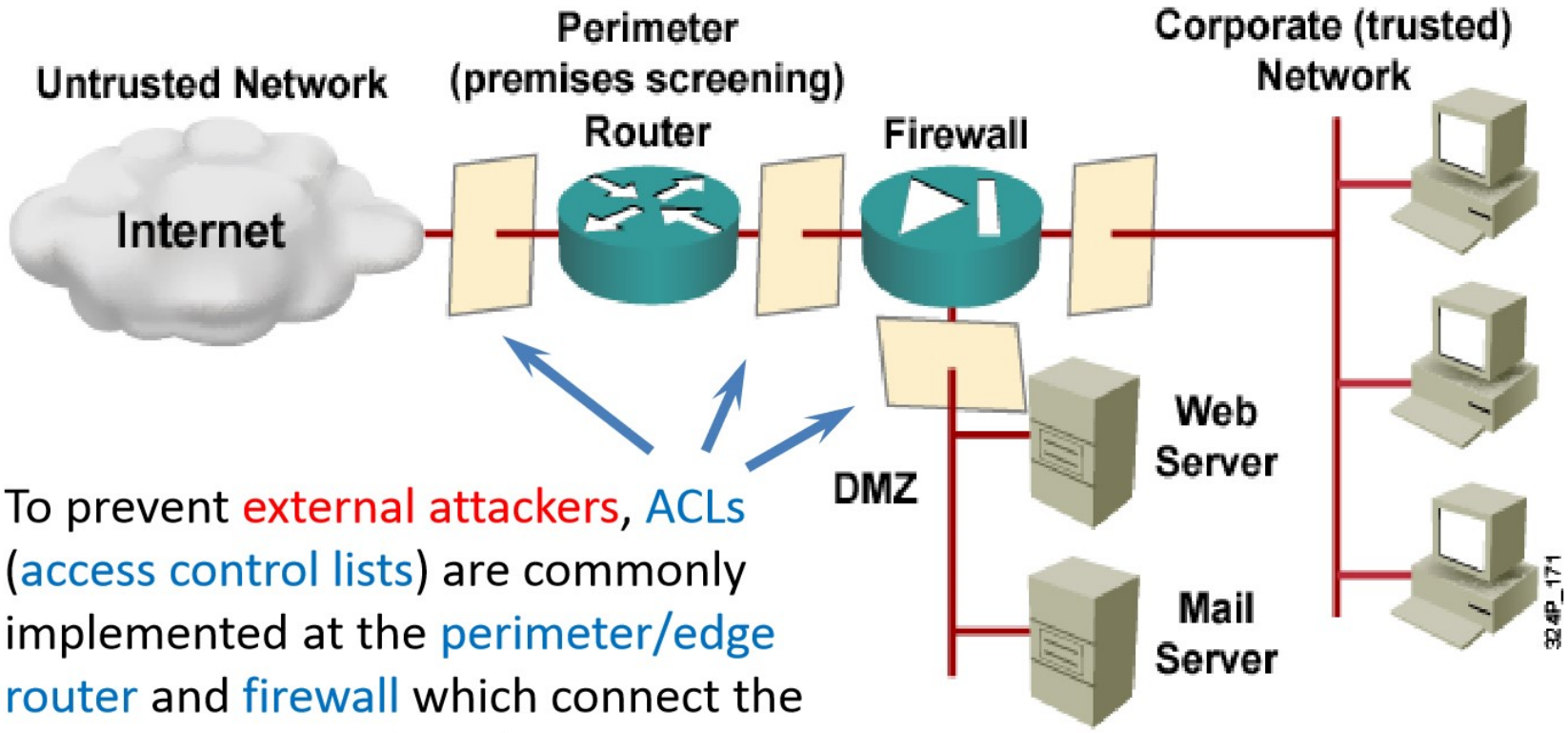
SINGAPORE INSTITUTE OF TECHNOLOGY

Lab 5 Notes:

Attacks and Defense of IP Networks with Routers

2021-2022 Trimester 3

In contrast to internal attacks, the first point of entry for external attackers are typically the routers or firewalls.



To prevent external attackers, ACLs (access control lists) are commonly implemented at the perimeter/edge router and firewall which connect the enterprise network to the Internet.

Lab Exercise 1

SINGAPORE INSTITUTE OF TECHNOLOGY

1.1 Network probing and scanning attack

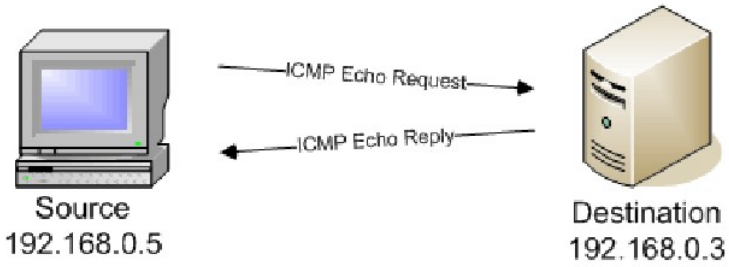
1.2 Defense:

- Minimise using suitable ACLs in routers

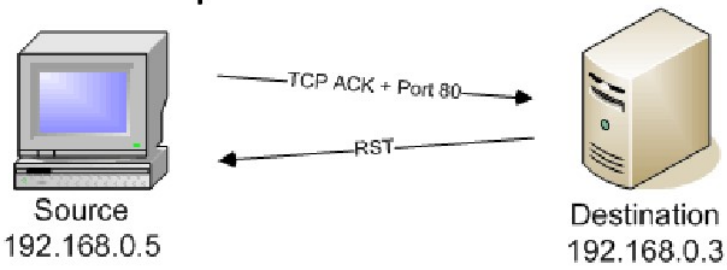
A potential external attack is reconnaissance attack, specifically network probing and port scanning attacks to discover targets which you'll learn more in ICT2204 Ethical Hacking.

Technically, network probing and scanning attacks are based on sending ARP, ICMP, TCP or UDP requests and receiving/not receiving the replies.

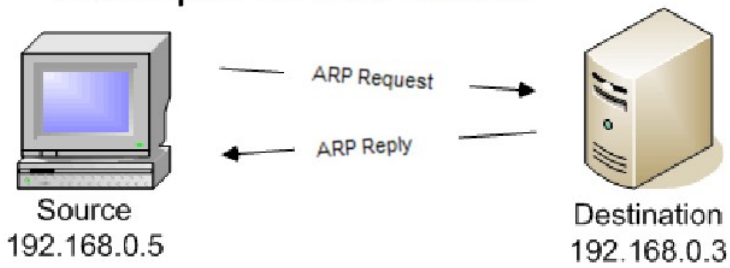
Example of ICMP scan:



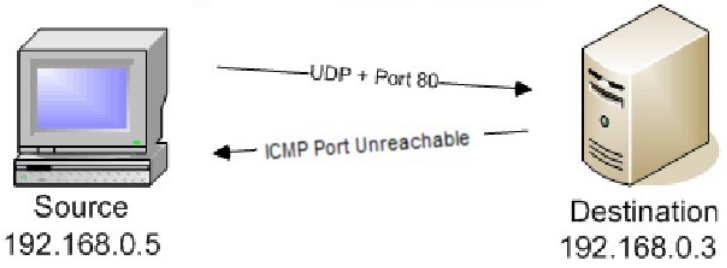
Example of TCP scan:



Example of ARP scan:



Example of UDP scan:



A useful tool for conducting network probing and scanning attacks is nmap which you will try in the lab.

Nmap "Network Mapper"

Base Syntax

```
# nmap [ScanType] [Options] {targets}
```

Target Specification

IPv4 address: 192.168.1.1
IPv6 address: AABB:CCDD::FF%eth0
Host name: www.target.tgt
IP address range: 192.168.0-255.0-255
CIDR block: 192.168.0.0/16
Use file with lists of targets: -iL <filename>

Target Ports

No port range specified scans 1,000 most popular ports

- -F Scan 100 most popular ports
- -p<port1>-<port2> Port range
- -p<port1>,<port2>,... Port List
- -pU:53,U:110,T20-445 Mix TCP and UDP
- -r Scan linearly (do not randomize ports)
- --top-ports <n> Scan n most popular ports
- -p-65535 Leaving off initial port makes Nmap scan start at port 1
- -p0- Leaving off end port makes Nmap scan up to port 65535
- -p- Leaving off start and end port makes Nmap scan ports 1-65535

Probing Options

- -Pn Don't probe (assume all hosts are up)
- -PB Default probe (TCP 80, 445 & ICMP)
- -PS<portlist> Check whether targets are up by probing TCP ports
- -PE Use ICMP Echo Request
- -PP Use ICMP Timestamp Request
- -PM Use ICMP Netmask Request

Scan Types

- -sn Probe only (host discovery, not port scan)
- -sS SYN Scan
- -sT TCP Connect Scan
- -sU UDP Scan
- -sV Version Scan
- -O OS Detection
- --scanflags Set custom list of TCP using URGACKPSHRSTSYNFIN in any order

Aggregate Timing Options

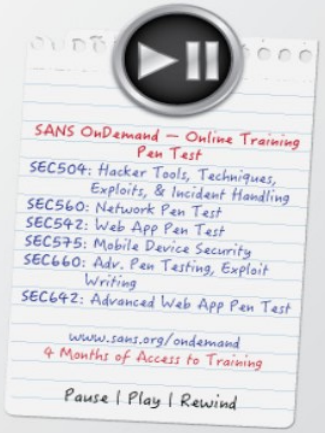
- -T0 Paranoid: Very slow, used for IDS evasion
- -T1 Sneaky: Quite slow, used for IDS evasion
- -T2 Polite: Slows down to consume less bandwidth, runs ~10 times slower than default
- -T3 Normal: Default, a dynamic timing model based on target responsiveness
- -T4 Aggressive: Assumes a fast and reliable network and may overwhelm targets
- -T5 Insane: Very aggressive; will likely overwhelm targets or miss open ports

Output Formats

- -oN Standard Nmap output
- -oG Greppable format
- -oX XML format
- -oA <basename> Generate Nmap, Greppable, and XML output files using basename for files

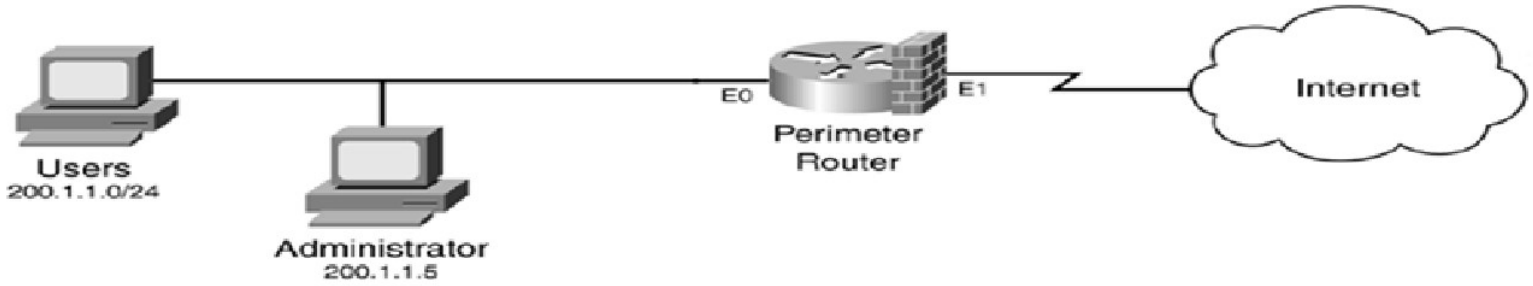
Misc Options

- -n Disable reverse IP address lookups
- -6 Use IPv6 only
- -A Use several features, including OS Detection, Version Detection, Script Scanning (default), and traceroute
- --reason Display reason Nmap thinks port is open, closed, or filtered



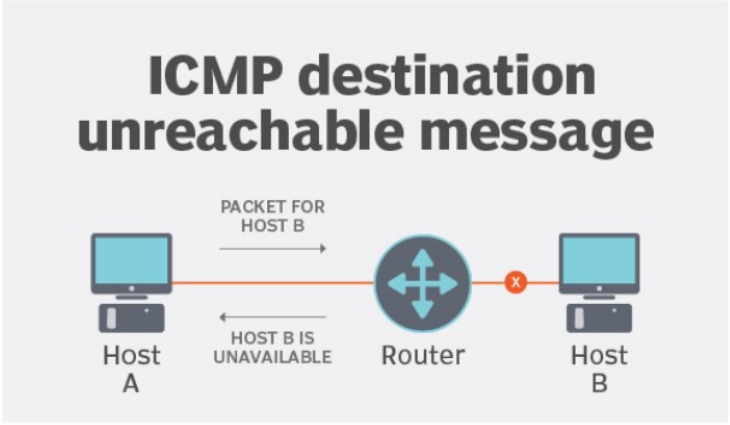
SANS
www.sans.org

Unfortunately, it can be difficult to fully prevent network probing and scanning attacks; but some may be blocked by suitable ACLs; e.g. ingress filtering of ICMP scan as follows:




```
R1(config)# ip access-list extended nmapICMPscan-in
R1(config-ext-nacl)# deny icmp any any echo
R1(config-ext-nacl)# deny icmp any any timestamp-request
R1(config-ext-nacl)# deny icmp any any mask-request
R1(config-ext-nacl)# permit ip any any
R1(config)# interface E1
R1(config-if)# ip access-group nmapICMPscan-in in
R1(config-if)# no ip unreachables
```

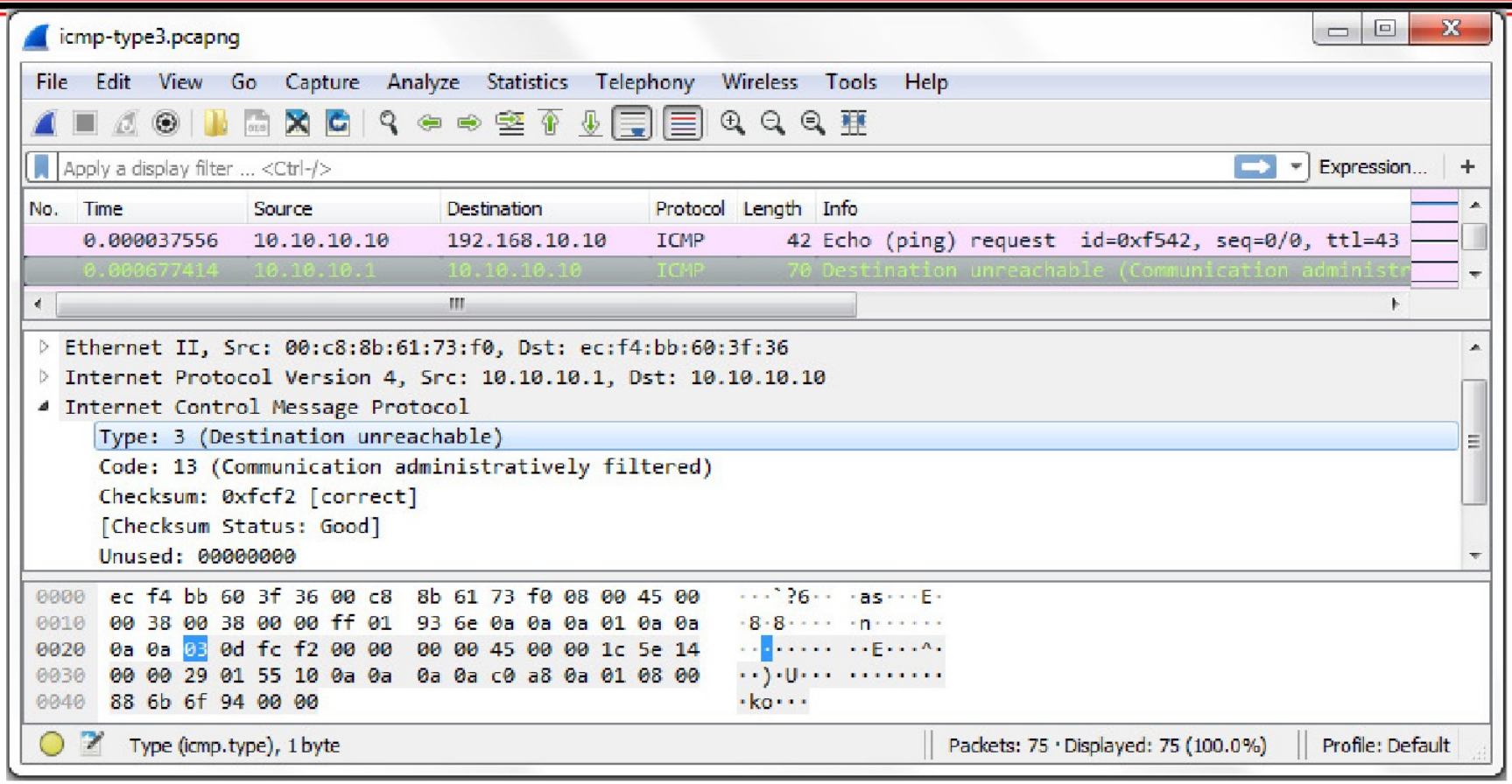
Recall ICT1010 on ICMP error-reporting messages? Note that there are different type 3 destination unreachable error messages for different error conditions (codes) as shown:



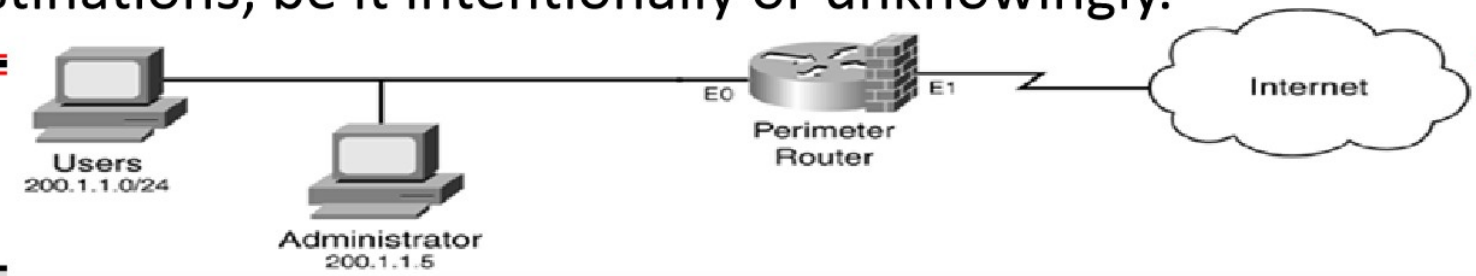
Host A attempts to send an IP packet to Host B. Host B is unreachable, so the router responds with a Type 3 ICMP message.

Type		Code	Description
3	Destination Unreachable	0	Destination network unreachable
		1	Destination host unreachable
		2	Destination protocol unreachable
		3	Destination port unreachable
		4	Fragmentation required, and DF flag set
		5	Source route failed
		6	Destination network unknown
		7	Destination host unknown
		8	Source host isolated
		9	Network administratively prohibited
		10	Host administratively prohibited
		11	Network unreachable for ToS
		12	Host unreachable for ToS
		13	Communication administratively prohibited
		14	Host Precedence Violation
		15	Precedence cutoff in effect

No ip unreachables will prevent a router configured with ACL to drop packets from sending ICMP type 3 destination unreachable which will leak information to attackers, e.g. shown below:



In addition, you may want to implement egress filtering of ICMP to prevent someone inside your network from scanning outside destinations, be it intentionally or unknowingly.




```
R1(config)# ip access-list extended ICMPscan-out
R1(config-ext-nacl)# permit icmp host 200.1.1.5 any echo          ①
R1(config-ext-nacl)# permit icmp 200.1.1.0 0.0.0.255 any
replace-with-actual-ICMP-error-reporting-message-see-below        ②
R1(config-ext-nacl)# deny icmp any any
R1(config-ext-nacl)# permit ip any any
R1(config)# interface E1
R1(config-if)# ip access-group ICMPscan-out out
```

① Allow administrator to ping which is useful for testing connectivity.

② Trade-off allowing ICMP error-reporting messages vs attacks; e.g.

- unreachable: inform destination unreachable but exploited by UDP scan
- time-exceeded: exploited by attacker to trace route to target

Lab Exercise 2

SINGAPORE INSTITUTE OF TECHNOLOGY

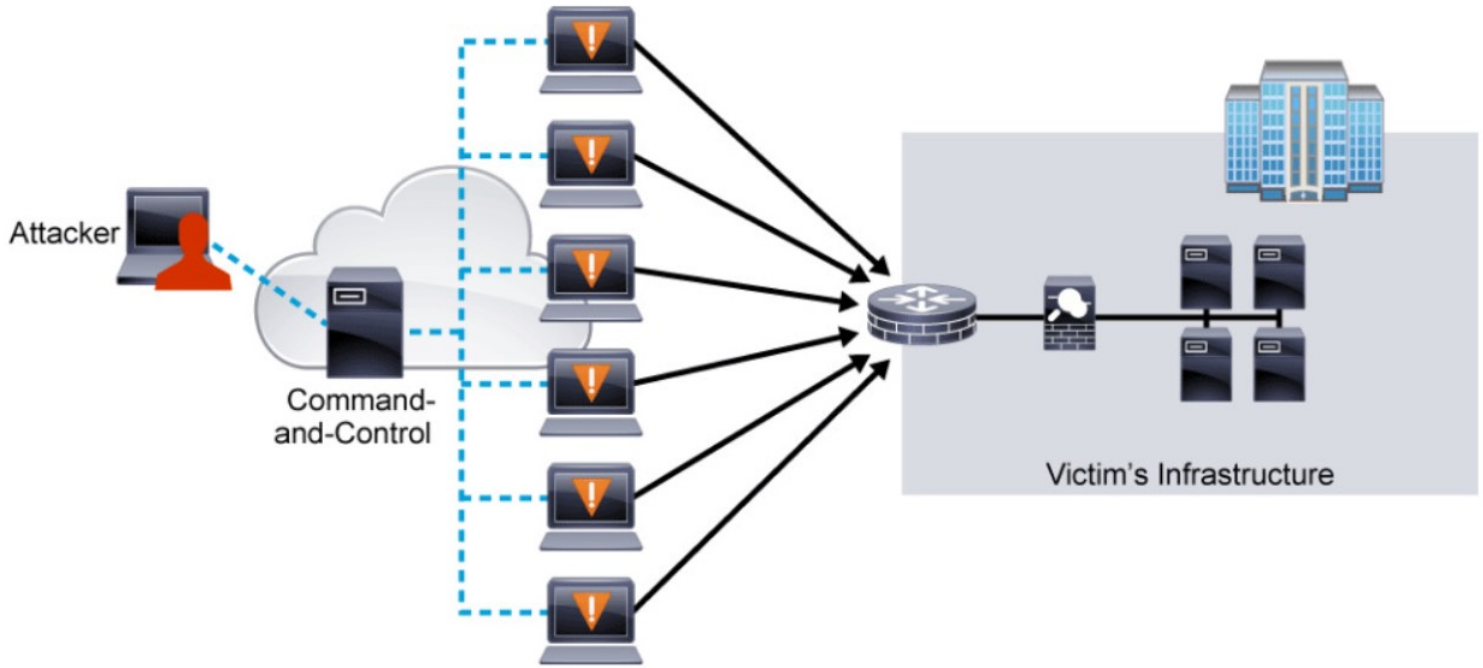
2.1 Smurf attack with spoofed source IP address

2.2 Defenses:

- Disable hosts from replying to broadcast ping
- Disable routers from supporting directed broadcast
- Apply BCP 38/RFC 2827 and anti-spoofing ACLs in routers

Another potential external attack is DoS (Denial-of-Service) or DDoS (Distributed DoS) which aims to make the target unavailable to authorised users.

Typically, DoS/DDoS aims to overload the network bandwidth or resources of the target to make it unavailable.

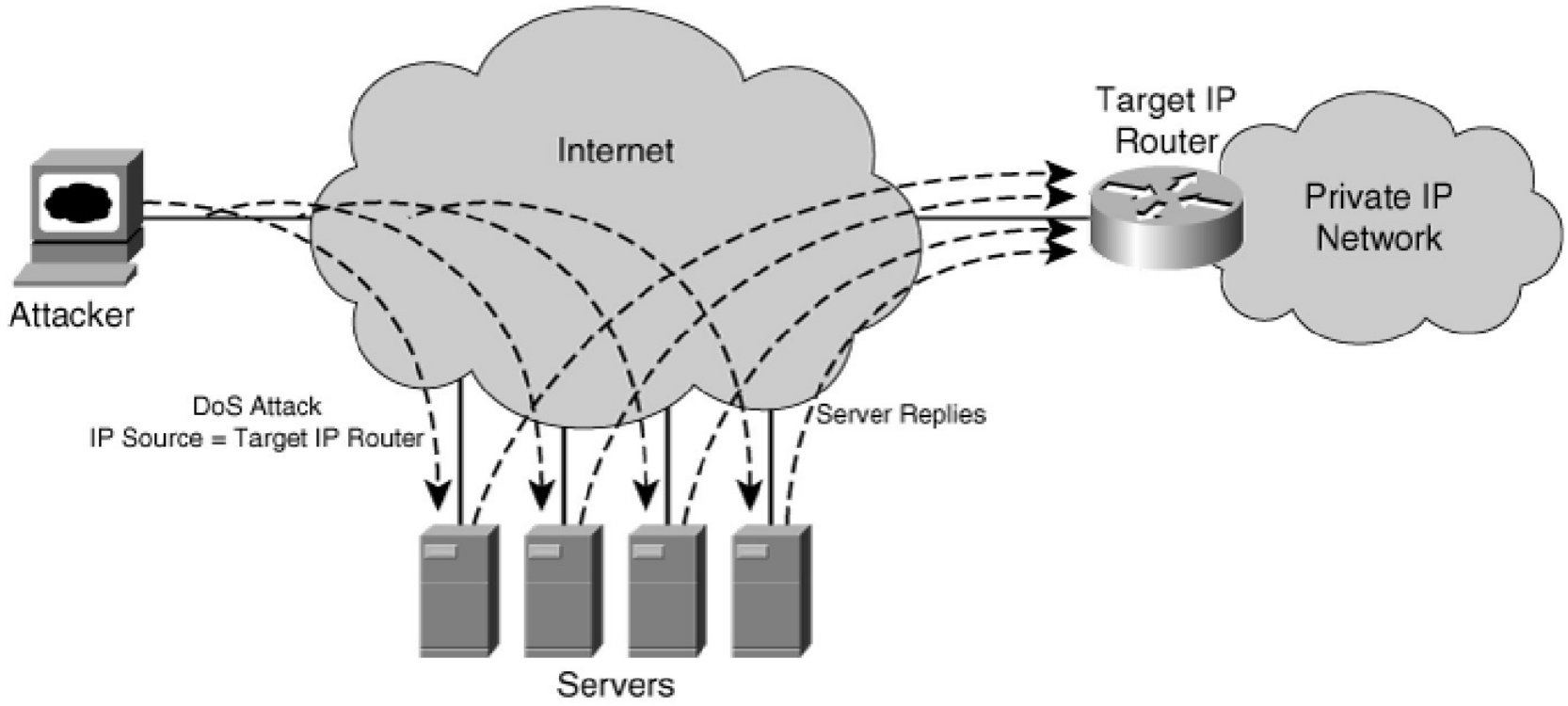


DDoS attack is basically DoS attacks that are launched using an army of compromised hosts called zombies/bots distributed in the Internet.

Broadly, DoS/DDoS attacks may be classified into 2 categories as follows:

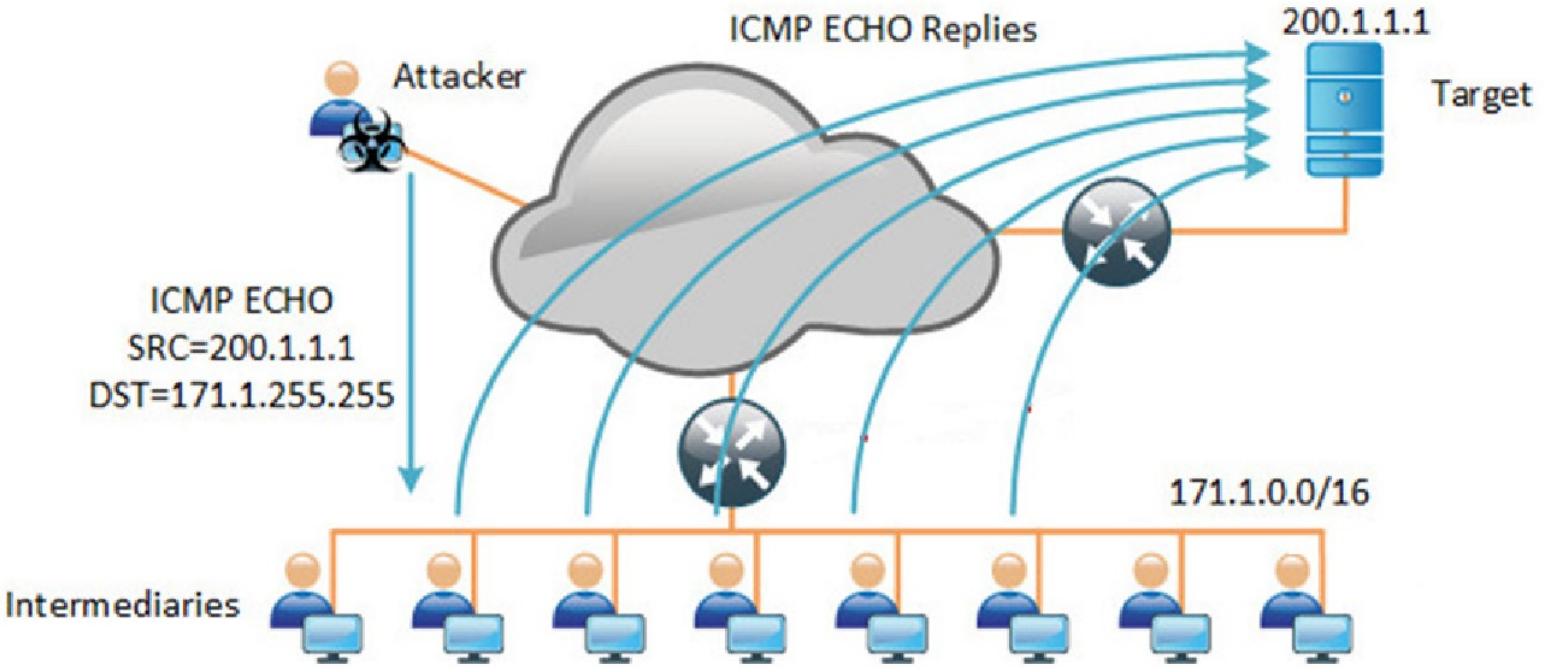
DoS/DDoS categories	Description	Typical attack packet rate	Examples
Overwhelming Quantity of Traffic	Threat actor sends an enormous quantity of data at a rate that the network, host, or application cannot handle, possibly causing slow response times or even crash a device or service.	High	ICMP flood, UDP flood, smurf attack, TCP SYN, etc
Maliciously Crafted Packets	Threat actor sends maliciously crafted packets that the host or application is not expected to handle, possibly causing it to run very slowly or even crash.	Low	Slowloris, ping of death, etc

Reflection attack is a type of DoS/DDoS which works by sending packets to intermediary servers using spoofed source IP belonging to target so that replies are reflected to attack target.



If the reflection attack uses small forged packets which resulted in large replies from the intermediaries, it is also known as amplification attack; e.g. smurf attack appeared in late 1990s.

In smurf attack, a single broadcast ping sent to a directed broadcast address by attacker using spoofed IP source address belonging to target would result in multiple ping replies (each from an intermediary in the directed broadcast subnet) reflected to the target.



To have a better understanding of amplification attack, you will use hping3 in Kali Linux to try smurf attack in the lab.

```
root@kali:~# hping3 -1 --flood -a target_ip directed_broadcast_address
```

Refer *https://www.kali.org/tools/hping3* on the use of hping3:

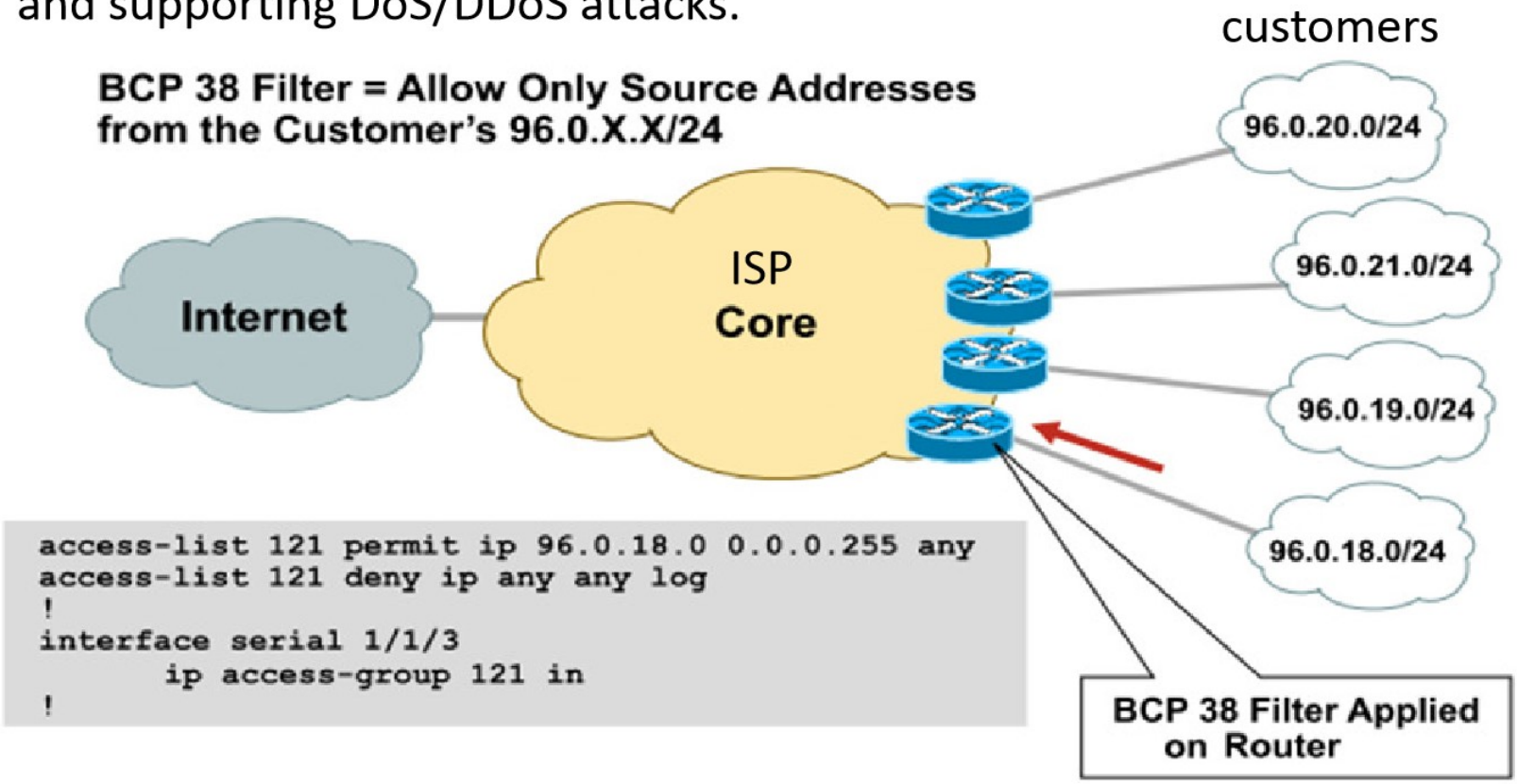
```
┌──(kali㉿kali)-[~]
└─$ hping3 --help
usage: hping3 host [options]
  -h   --help       show this help
  -v   --version    show version
  -c   --count      packet count
  -i   --interval   wait (uX for X microseconds, for example -i u1000)
       --fast       alias for -i u10000 (10 packets for second)
       --faster     alias for -i u1000 (100 packets for second)
       --flood       sent packets as fast as possible. Don't show replies.
  -n   --numeric    numeric output
  -q   --quiet      quiet
  -I   --interface  interface name (otherwise default routing interface)
  -V   --verbose    verbose mode
  -D   --debug      debugging info
  -z   --bind       bind ctrl+z to ttl           (default to dst port)
  -Z   --unbind     unbind ctrl+z
       --beep       beep for every matching packet received
Mode
  default mode      TCP
  -0   --rawip      RAW IP mode
  -1   --icmp       ICMP mode
  -2   --udp        UDP mode
  -8   --scan       SCAN mode.
                    Example: hping --scan 1-30,70-90 -S www.target.host
  -9   --listen     listen mode
IP
  -a   --spoof      spoof source address
```

Today, smurf attack has been largely prevented by default through:

- Disable routers from supporting directed broadcast;
- Disable hosts from replying to broadcast ping.

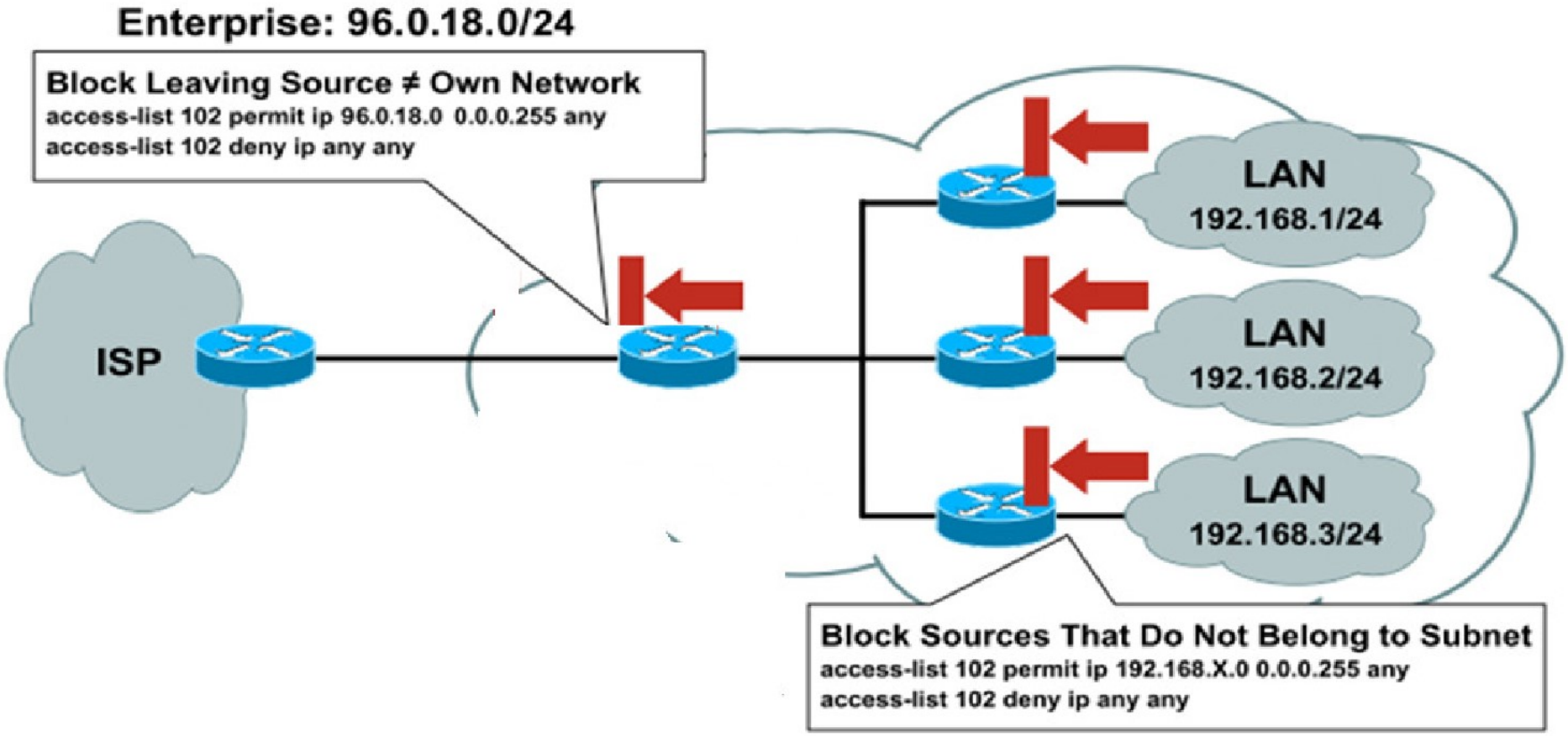
To prevent DoS/DDoS attack using spoofed source IP addresses like smurf attack, ISPs today are recommended to implement Best Current Practice BCP 38/RFC 2827.

BCP 38/RFC 2827 recommends ISPs to implement network ingress filtering to prevent their networks from permitting spoofed IP addresses and supporting DoS/DDoS attacks.

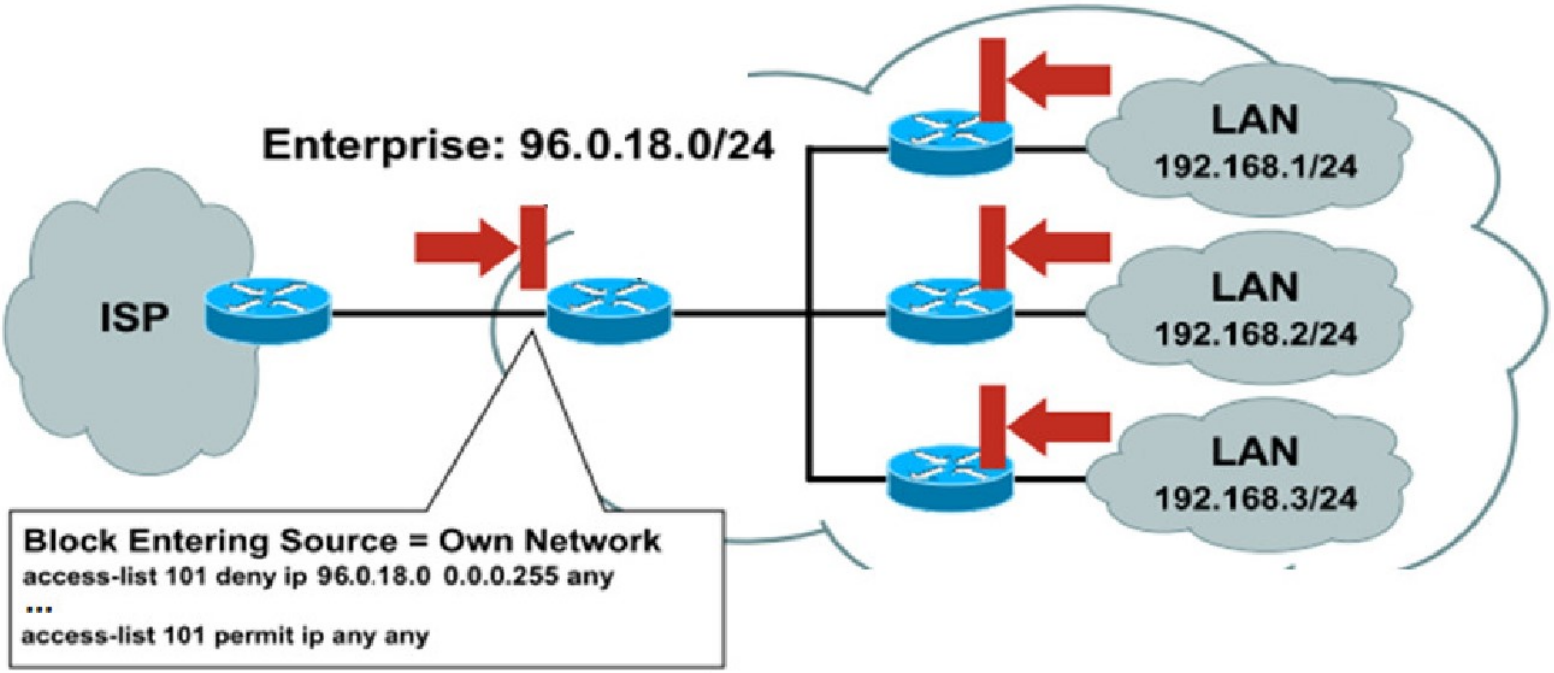



```
access-list 121 permit ip 96.0.18.0 0.0.0.255 any
access-list 121 deny ip any any log
!
interface serial 1/1/3
      ip access-group 121 in
!
```

Similarly, organizations should also implement BCP 38/RFC 2827 at their edge routers to prevent their hosts, possibly zombies, from using spoofed source IP addresses to launch attacks.



In addition, organizations should implement anti-spoofing ACL at their edge routers to prevent being victim to attacks using spoofed IP addresses.

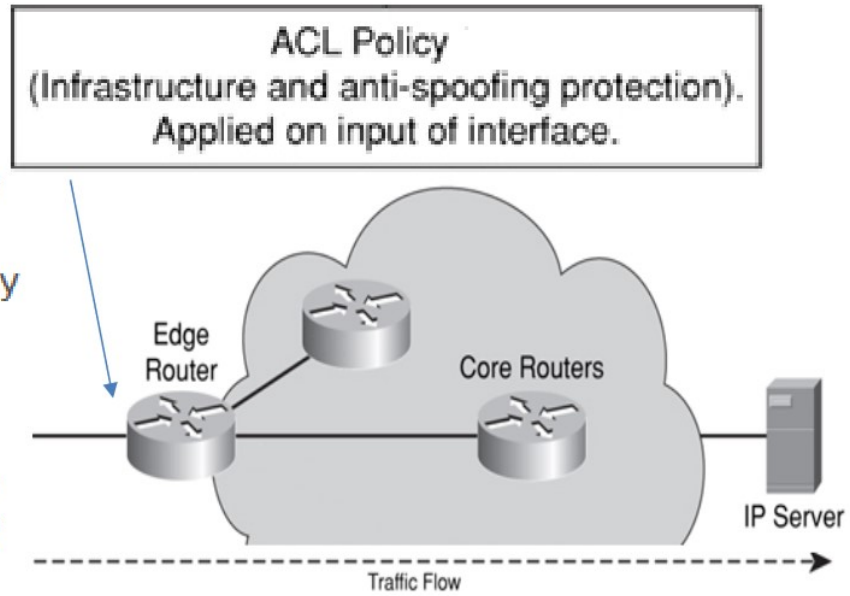


Anti-spoofing ACL is used to deny inbound IP packets that are obviously spoofed; e.g. source addresses that match your allocated public IP addresses, or special addresses in RFC 3330 and RFC 1918 that should not be used.

Typically, anti-spoofing ACL is implemented as a part of overall infrastructure protection ACL at the edge router for ingress filtering of malicious traffic from entering the network.

```
!--- Anti-spoofing entries are shown here.
!--- Deny RFC 3330 special-use address
access-list 110 deny ip host 0.0.0.0 any
access-list 110 deny ip 127.0.0.0 0.255.255.255 any
access-list 110 deny ip 192.0.2.0 0.0.0.255 any
access-list 110 deny ip 224.0.0.0 31.255.255.255 any

!--- Filter RFC 1918 space.
access-list 110 deny ip 10.0.0.0 0.255.255.255 any
access-list 110 deny ip 172.16.0.0 0.15.255.255 any
access-list 110 deny ip 192.168.0.0 0.0.255.255 any
```




```
!--- Deny your space as source from entering your AS. !--- Deploy only at the AS edge.
access-list 110 deny ip YOUR_CIDR_BLOCK any

!--- Deny access to internal infrastructure addresses.
access-list 110 deny ip any INTERNAL_INFRASTRUCTURE_ADDRESSES

!--- Permit transit traffic.
access-list 110 permit ip any any
```

Lab Exercise 3

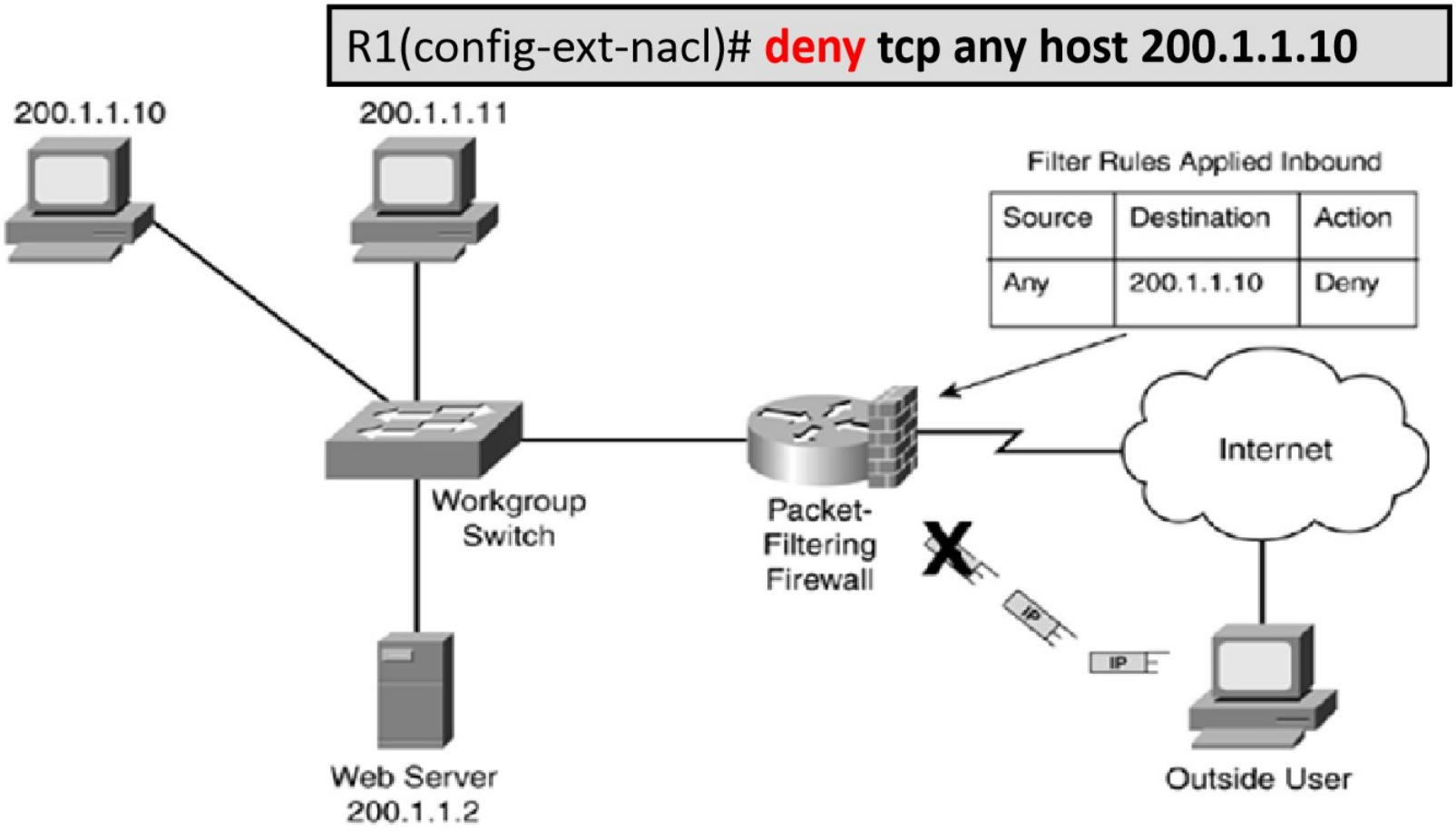
SINGAPORE INSTITUTE OF TECHNOLOGY

3.1 Tiny fragment and overlapping fragment attacks

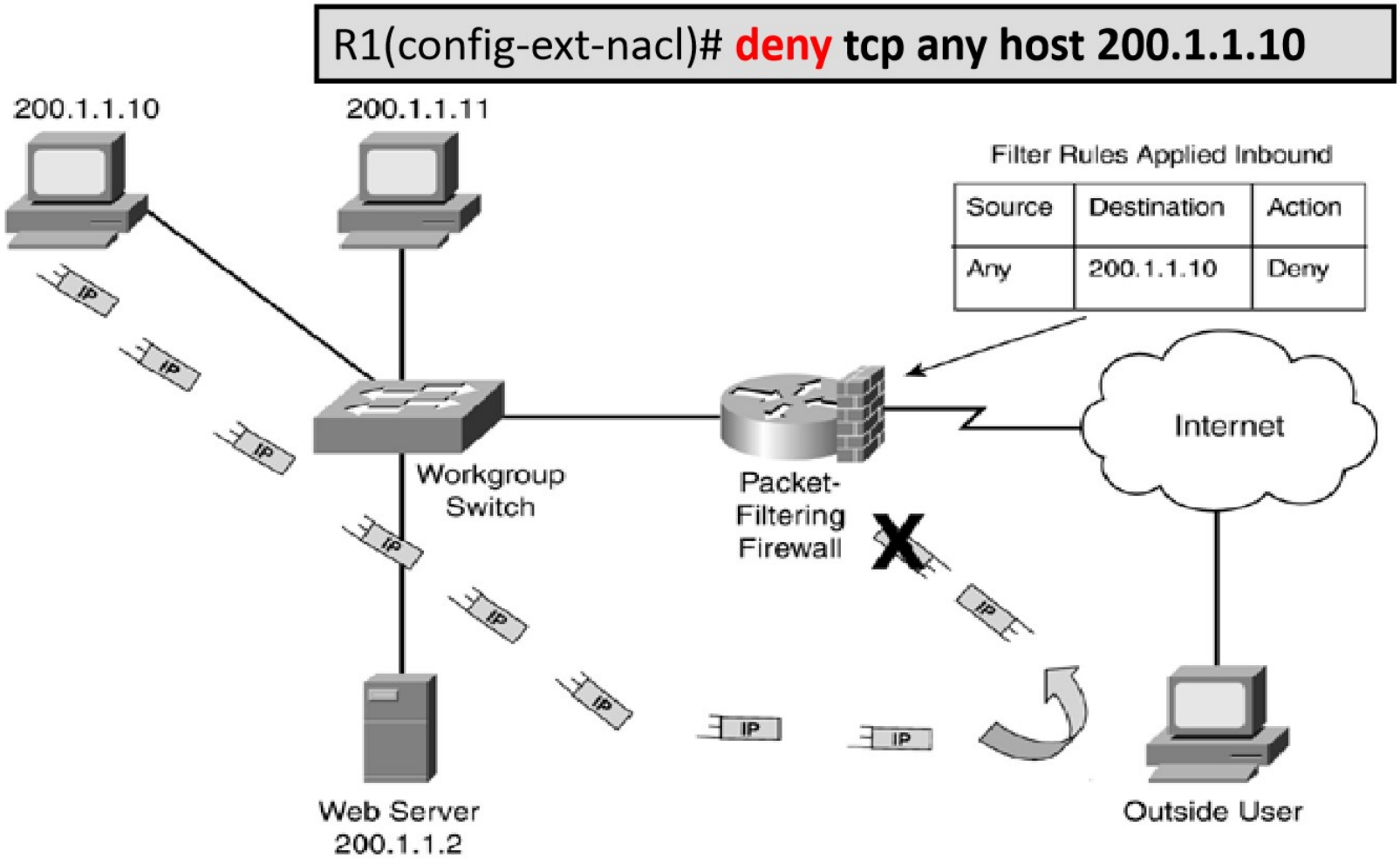
3.2 Defenses:

- RFC 1858 to discard fragments with offset FO = 1
- Apply ACLs with 'fragments' option

To protect internal hosts, one may be tempted to simply use ACL to filter external attackers from reaching them, e.g. for host 200.1.1.10 as follows:



However, blocking external attackers from reaching internal hosts will also lead to blocking internal hosts from accessing the Internet because all returning packets will be blocked!

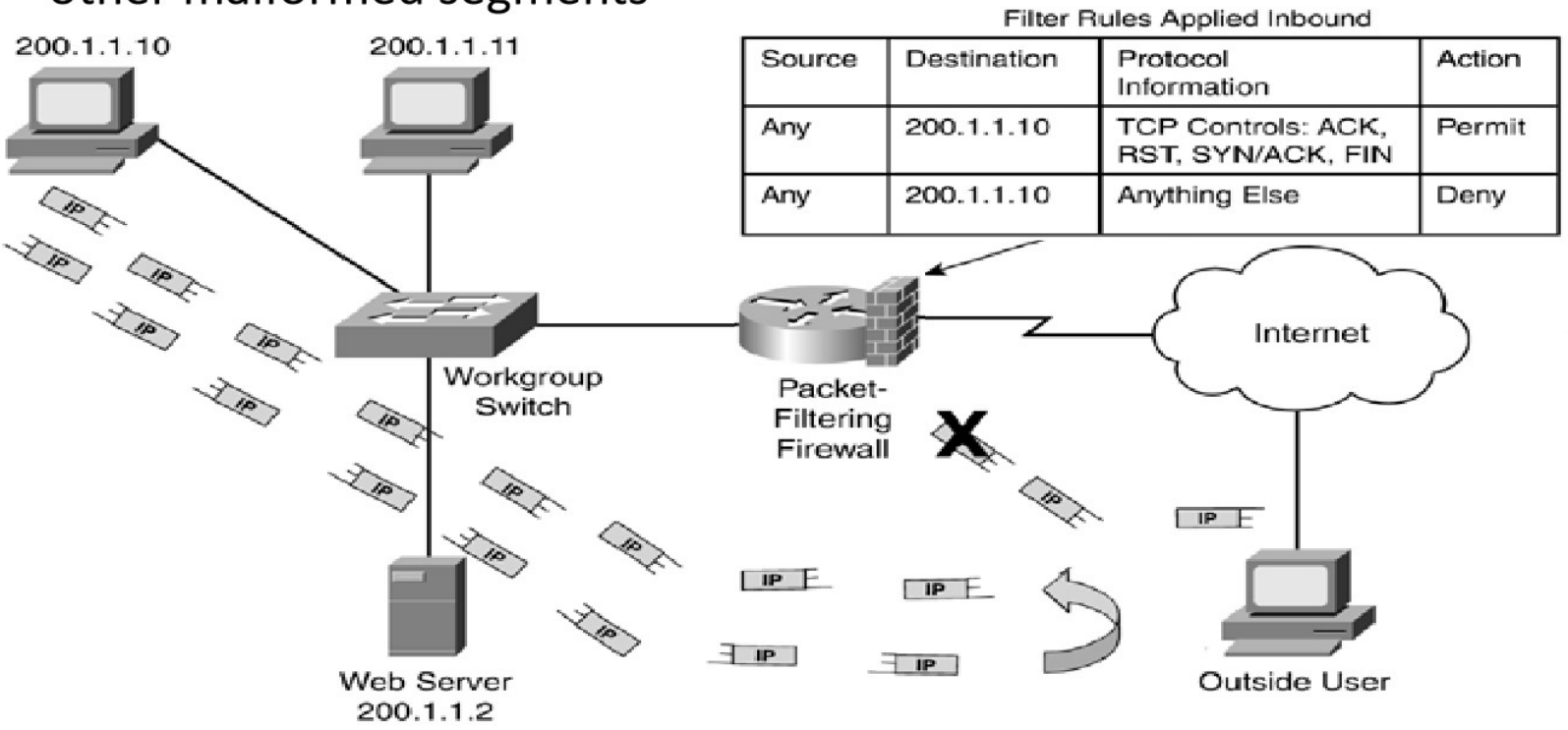


So, ACL statement has been enhanced with an optional 'established' parameter for protecting internal hosts.

R1(config-ext-nacl)# **permit tcp any host 200.1.1.10 established**

The 'established' option applies only to TCP and is meant to:

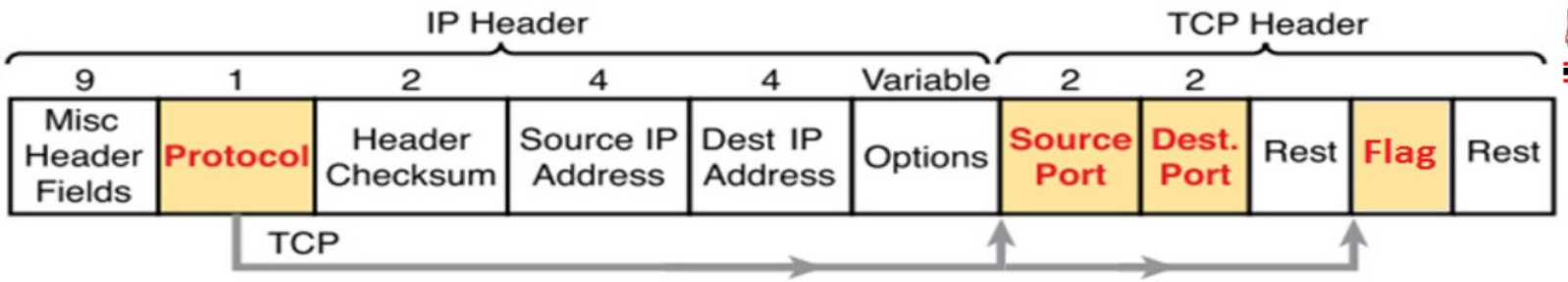
- Allow returning TCP segments of existing connections
- Block TCP segments initiating a new connection from outside, or other malformed segments



Filter Rules Applied Inbound

Source	Destination	Protocol Information	Action
Any	200.1.1.10	TCP Controls: ACK, RST, SYN/ACK, FIN	Permit
Any	200.1.1.10	Anything Else	Deny

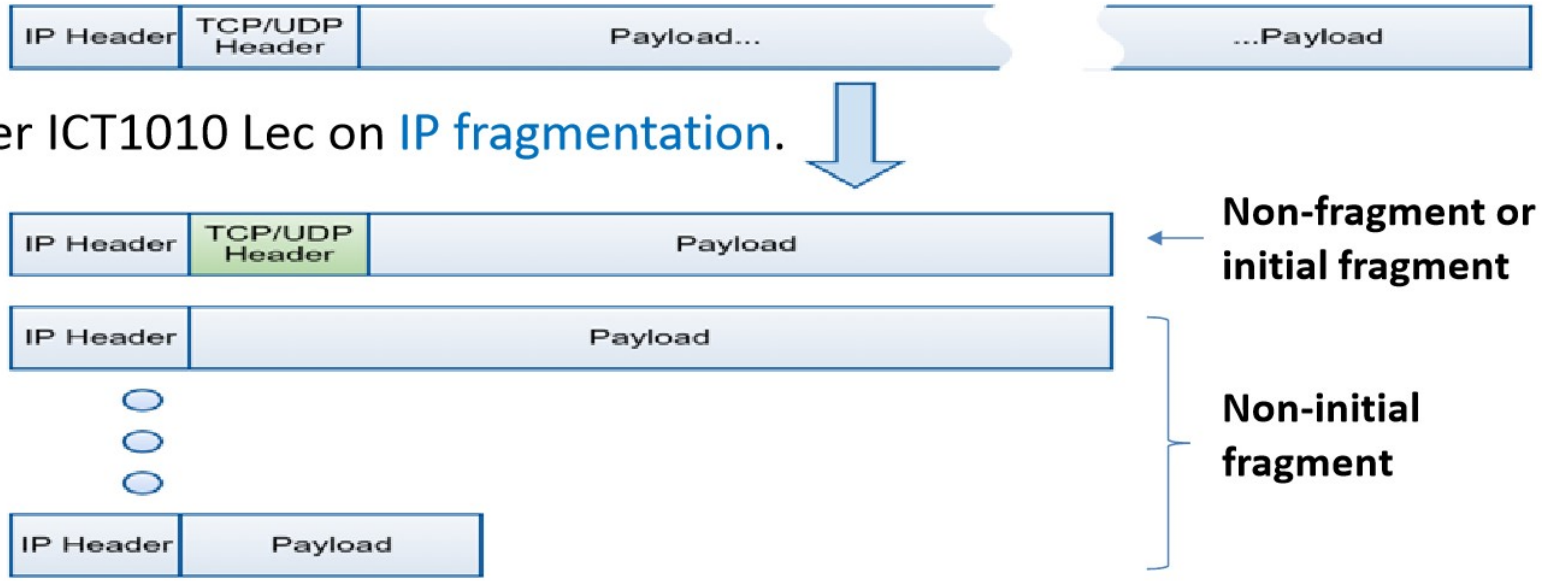
Alternatively, you may configure ACL to examine the TCP flags (refer ICT1010 Lec on TCP) of each packet, e.g. as follows:



- Allow returning TCP segments of existing connections, e.g. TCP flags with +syn +ack, +ack, +fin, +fin +ack, +rst
- Block other TCP segments, e.g. +syn because it is the first segment of 3-way handshake initiating a new connection

```
R1(config)# ip access-list extended TCPFLAG
R1(config-ext-nacl)# permit tcp any any match-all +syn +ack
R1(config-ext-nacl)# permit tcp any any match-all +ack
R1(config-ext-nacl)# permit tcp any any match-all +fin
R1(config-ext-nacl)# permit tcp any any match-all +fin +ack
R1(config-ext-nacl)# permit tcp any any match-all +rst
R1(config-ext-nacl)# deny tcp any any
R1(config-ext-nacl)# permit ip any any
```

Unfortunately, attackers have come up with IP fragmentation attacks to disguise IP packets to evade ACL filtering.



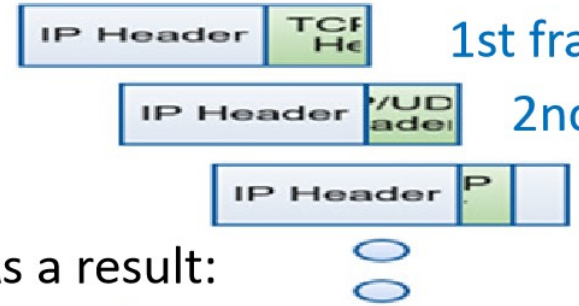
By default, ACLs are only applied to non-fragments and initial fragments.

For efficiency, non-initial fragments are simply passed without filtering because if initial fragment is already filtered, destination host will eventually discard non-initial fragments after reassembly timeout.

Specifically, one technique of using IP fragmentation to evade ACL filtering is tiny fragment attack (RFC 1858).

Tiny fragment attack intentionally breaks an IP packet into many tiny fragments; e.g. 8 bytes of data:

Since TCP header is 20 bytes:

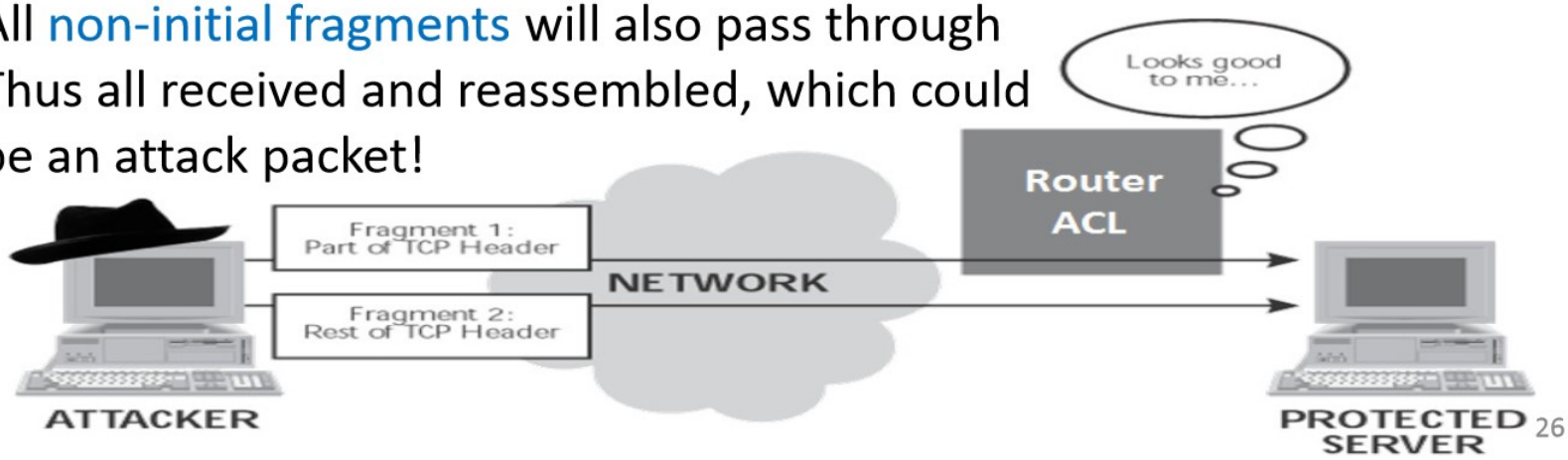


1st fragment – containing TCP port numbers

2nd fragment – containing TCP flags

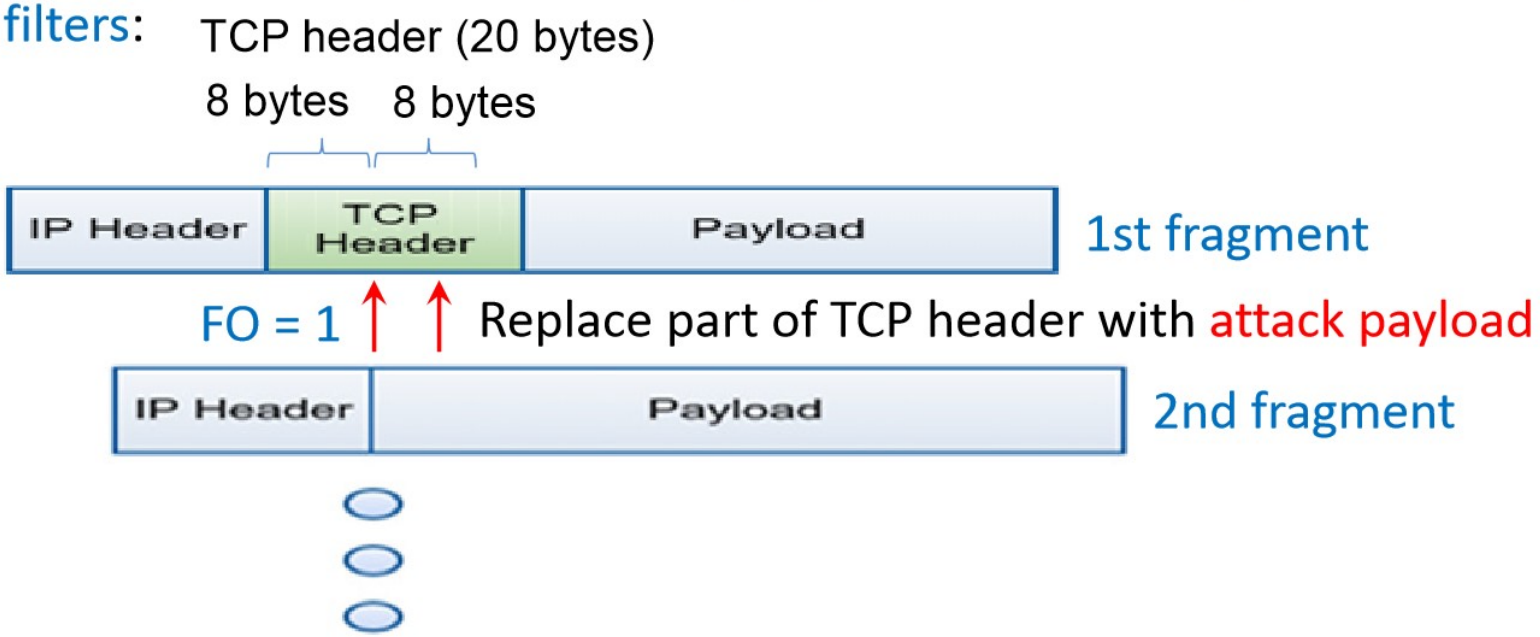
As a result:

- 1st fragment will pass through because ACL that filters based on TCP flags will not be matched
- All non-initial fragments will also pass through
- Thus all received and reassembled, which could be an attack packet!



Another technique of using IP fragmentation to evade ACL filtering is overlapping fragment attack (RFC 1858).

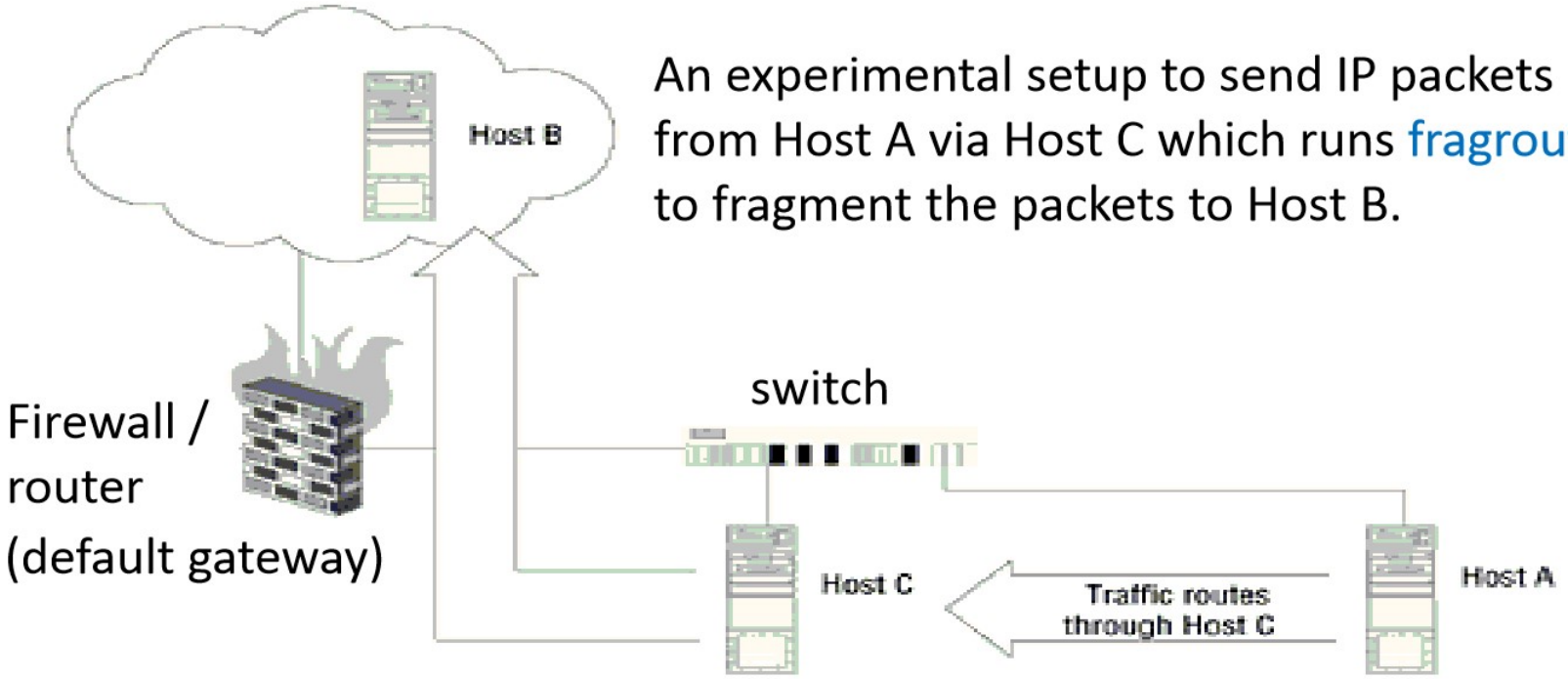
- 1st fragment is constructed with proper TCP header to pass ACL filters:



- 2nd fragment is constructed with desired fragmentation offset (e.g. FO = 1) and attack payload
- When all received and reassembled, attack payload will overlap and replace part of original TCP header, leading to an attack packet!

To experiment with IP fragmentation attacks, you will try a tool called fragrouter in Kali Linux.

Refer *http://linux.die.net/man/8/fragrouter* on the use of fragrouter:



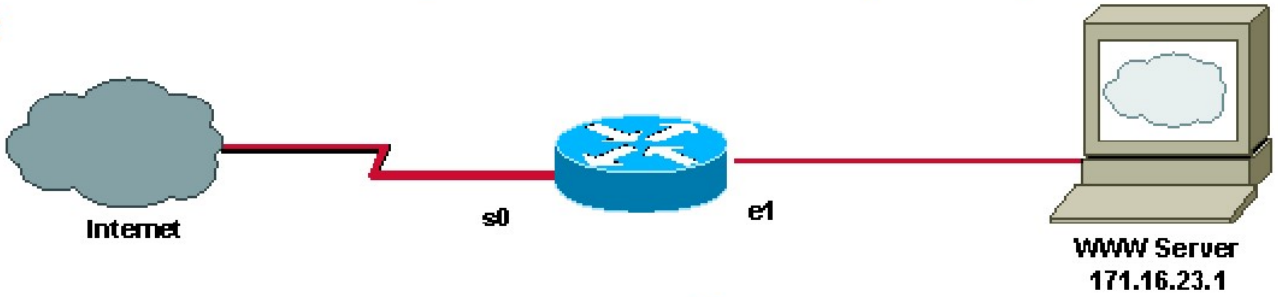
An experimental setup to send IP packets from Host A via Host C which runs fragrouter to fragment the packets to Host B.

Host C default route to default gateway, running fragrouter, e.g. tiny fragment attack:

```
root@kali:~# fragrouter -i eth0 -F1
```

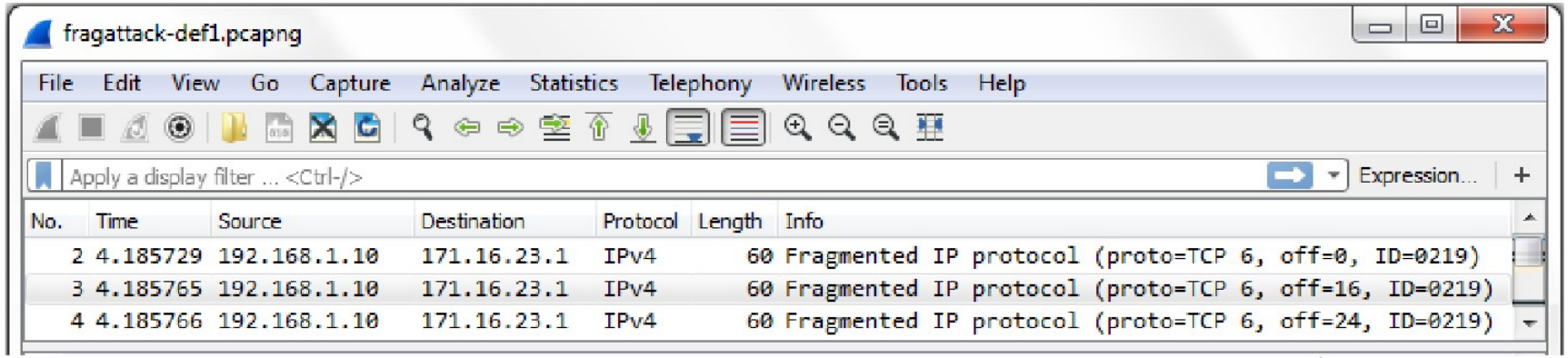
Change Host A default route to Host C IP address instead of default gateway.

To defend against tiny fragment and overlapping fragment attacks, Cisco has implemented RFC 1858 to drop IP fragment carrying TCP with fragment offset = 1 (FO=1).



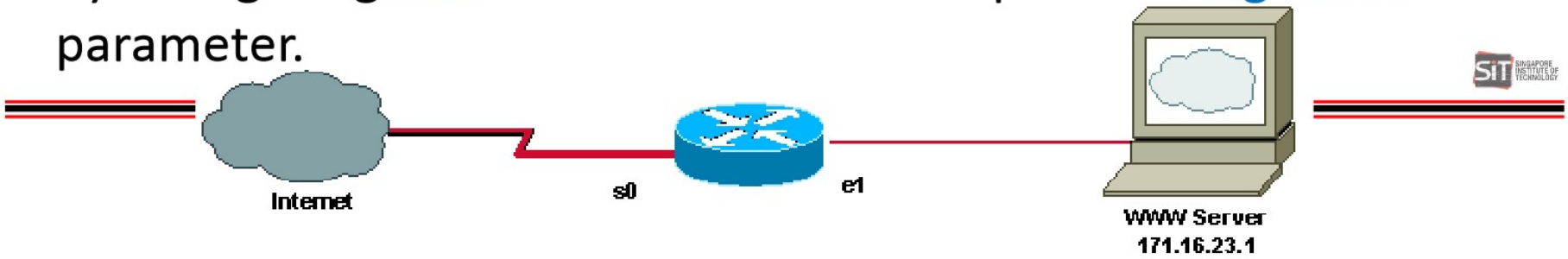
Applying following ACL for ingress filtering at interface s0:

```
access-list 100 permit tcp any host 171.16.23.1 eq 80
access-list 100 deny ip any any
```



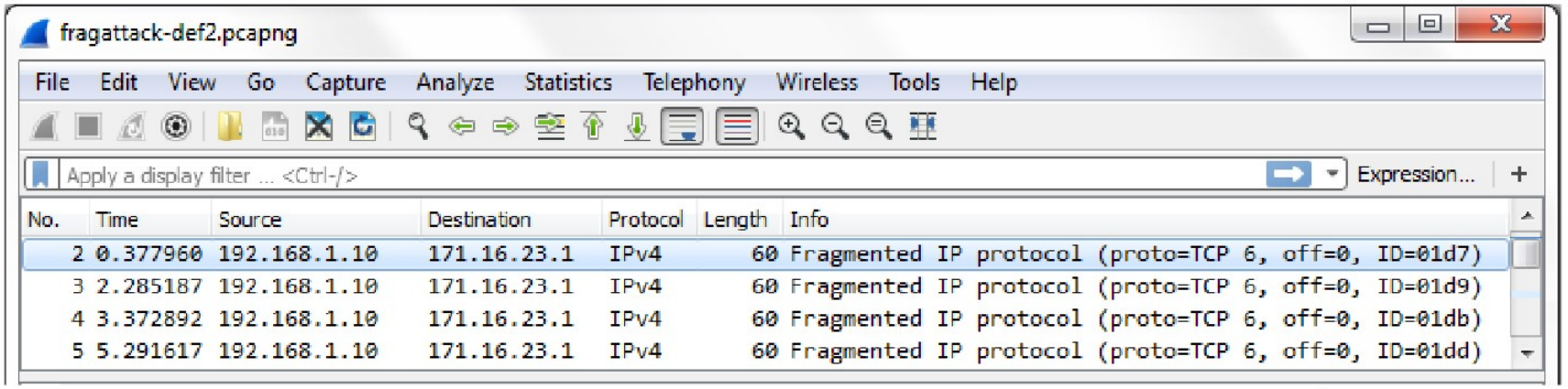
Notice fragment FO=1, denoted as off=8 in Wireshark, was dropped.

Alternatively, filtering of non-initial fragments can also be done by configuring ACL statements with the optional 'fragments' parameter.



Applying following ACL for ingress filtering at interface s0:

```
access-list 101 deny tcp any host 171.16.23.1 fragments
access-list 101 permit tcp any host 171.16.23.1 eq 80
access-list 101 deny ip any any
```



Notice all fragments are now dropped except initial fragment.