

Cyber Security Technologies

Session 10 – Firewalls

Shawn Davis
ITMS 448 – Spring 2016

Slides contain original content from Davis, S. and Lidinsky, W.
and may contain content from Ch.9 of Stallings, W. and Brown,
B, Computer Security 2ed; Pearson Education, Inc. 2012

OVA

- We will not be using RADISH for this session
- We will be using an OVA file to import an xubuntu VM into VirtualBox on your physical desktop
- Make sure the following share is mapped to your physical lab computer:
 - <\\coulson.otsads.iit.edu\itm448>
- Copy the xubuntu1404_64bit.ova file to your desktop

OVA (Cont.)

- Double-click on the OVA file on the desktop after it has finished copying
- When Vbox opens, select “Import”
- Go into the settings of the VM and disable the USB Controller (Remove the check)
- Start the VM after import (Don’t worry about the BIOS message)
- Logon = admin2
- Password = admin2

Overview

Part I – Firewall Introduction

Part II – Stateless Packet Filters

Part III – Stateful Packet Inspection Filters

Part IV – Proxy Servers

Part V – NAT

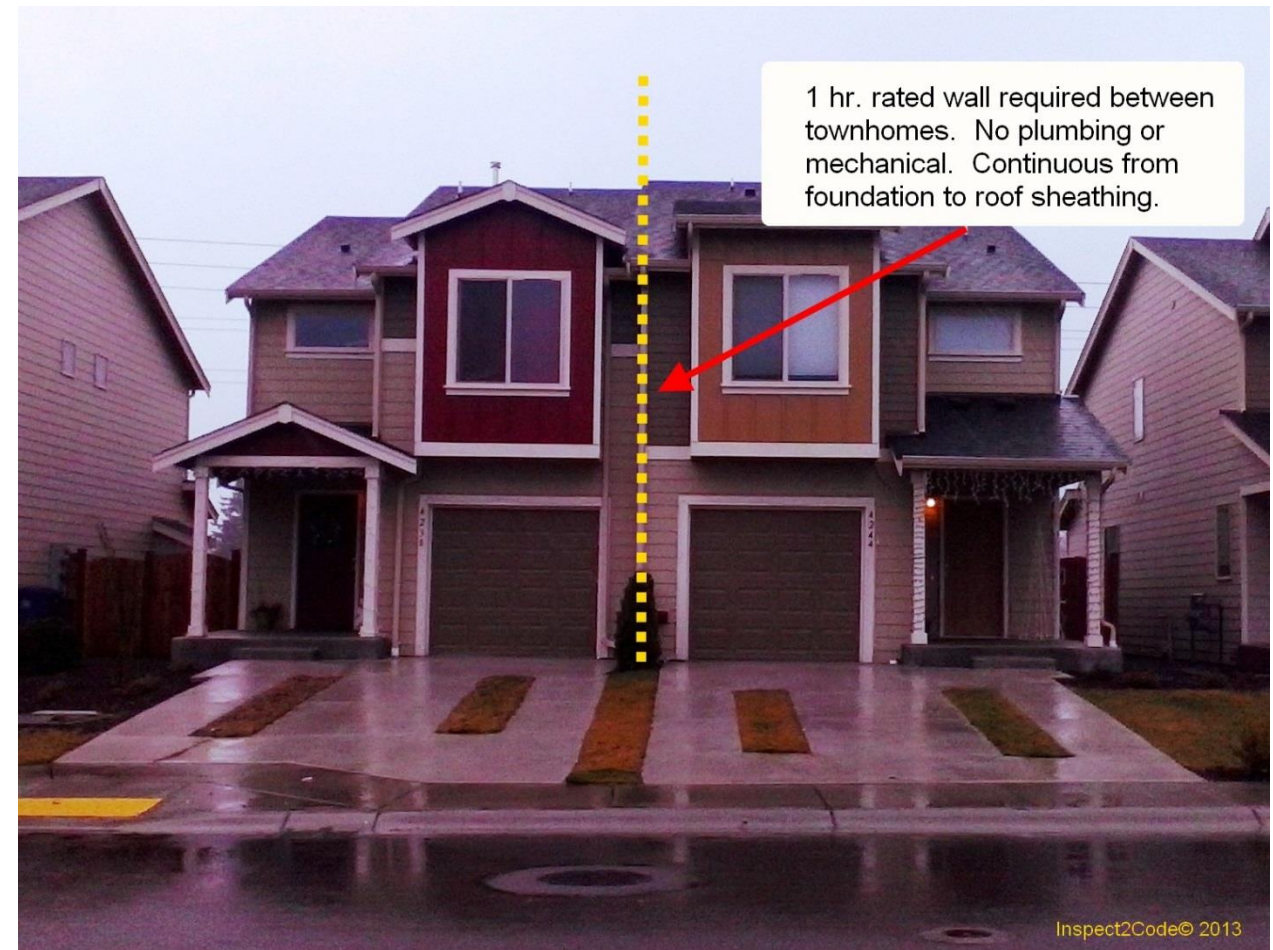
Part VI – Hands On: iptables

Part I

Firewall Introduction

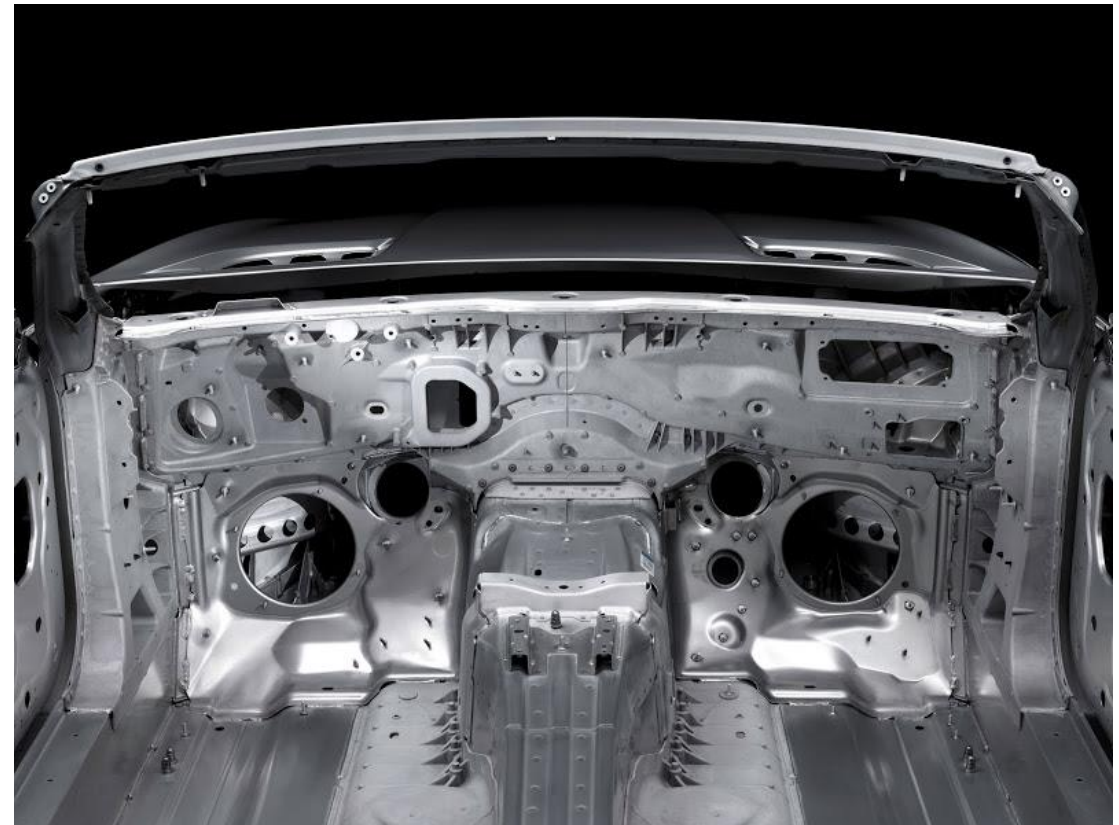
What is a Firewall?

- Traditionally, it is a wall that separates parts of a building from a fire



What is a Firewall? (Cont.)

- It is also the metal that separates an automobile passenger from the engine compartment



<http://4.bp.blogspot.com/-uJEexrjsC3w/TvdwmK72qAI/AAAAAAAAAMZY/pm0qF653nlk/s800/2013-Mercedes-Benz-SL-Roadster-FrontBass-Muic-system-Dash.jpg>

What is the purpose of these two Firewalls?

- To protect one area from an event that occurs in another area.
 - One room in a house is protected from a fire burning in another room
 - The passenger is protected against a fire burning in an automobile's engine compartment

Firewall (for our purposes)











- Protects a single host or a network segment of multiple hosts against intrusion or attack
- Still provides availability for host(s) to access the outside world through the Internet
- Can block internal as well as external threats

Host-based vs Network Firewall

- Host-based
 - Software on host system that allows or blocks incoming or outgoing transmissions
 - Linux:
 - ❖ iptables
 - Windows:
 - ❖ Windows Firewall with Advanced Security
- Network Firewall
 - Hardware firewall device that allows or blocks incoming or outgoing transmissions for network segment
 - Cisco, Juniper, WatchGuard, Barracuda, Sonicwall, pfSense, iptables, etc.

Windows Host-based Firewall

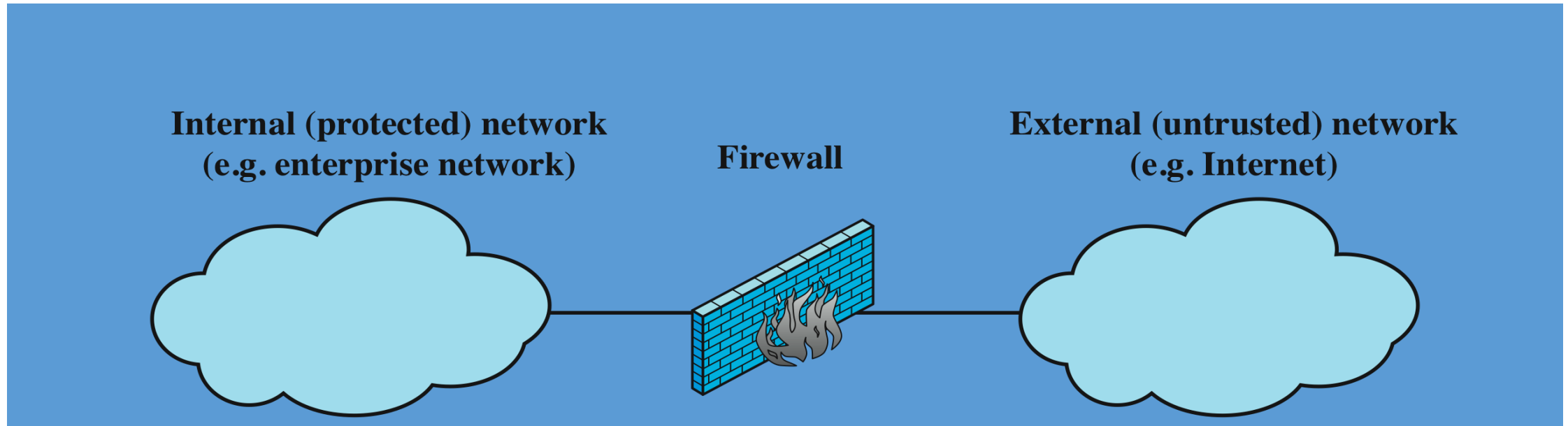
Inbound Rules

Name	Group	Profile	Enabled	Action	Override	Program	Local Address	Remote Address	Protocol	Local Port	Remote Port	Allowed Users	Allowed Computers
 8080		Domai...	Yes	Allow	No	Any	Any	Any	TCP	8080	Any	Any	Any
 AirPort		All	Yes	Allow	No	C:\Progr...	Any	Any	Any	Any	Any	Any	Any
 AirPort Utility		Public	Yes	Block	No	C:\progr...	Any	Any	TCP	Any	Any	Any	Any
 AirPort Utility		Public	Yes	Block	No	C:\progr...	Any	Any	UDP	Any	Any	Any	Any
 AirPort Utility		Private	Yes	Allow	No	C:\progr...	Any	Any	UDP	Any	Any	Any	Any
 AirPort Utility		Private	Yes	Allow	No	C:\progr...	Any	Any	TCP	Any	Any	Any	Any
 Backup		Private	Yes	Allow	No	C:\progr...	Any	Any	UDP	Any	Any	Any	Any
 Backup		Private	Yes	Allow	No	C:\progr...	Any	Any	TCP	Any	Any	Any	Any
 Backup		Public	Yes	Block	No	C:\progr...	Any	Any	TCP	Any	Any	Any	Any
 Backup		Public	Yes	Block	No	C:\progr...	Any	Any	UDP	Any	Any	Any	Any

Today

- This lecture will focus on network firewalls but we will use iptables for the hands on demo
- iptables can be configured as a host-based or network firewall
- We will be using iptables as a host-based firewall today in the demos for sake of simplicity
 - INPUT & OUTPUT Chains
 - No FORWARD Chain

Network Firewall



Firewall Characteristics

- All traffic must pass through the firewall
- Only authorized traffic (as defined by local security policy) will be allowed to pass
- Firewall itself must be immune to penetration
 - HTTPs for any Web GUI configuration
 - Strong password
 - Change default credentials
 - SSH (no telnet)
 - Patches up to date. Heartbleed anyone???

Four Firewall Techniques

1. Service Control
2. Direction Control
3. User Control
4. Behavior Control

Firewall Techniques

1. Service control

- Determines types of services that can be accessed inbound or outbound
- Filter based on:
 - IP address
 - Port number
 - Protocol
 - Content (if proxy)

2. Direction control

- Determines which direction traffic is allowed to flow through the firewall

Firewall Techniques (Cont.)

3. User control

- Controls access to service based on user attempting to access it

4. Behavior control

- Controls how specific services are accessed.
 - Firewall may filter e-mail or limit accessible information from a web server

Firewall Capabilities

- Defines a single choke point
- Creates logs for monitoring security events
- Often hosts other useful functions:
 - NAT
 - Web usage auditing
 - IPS/IDS
- Can often be used to implement virtual private networks (VPN)

Question

- What are some ways to defeat or get around a network firewall???

Firewall Limitations

- Cannot protect against attacks that don't flow through the firewall
 - Ex: A dial-up modem, mobile hot-spot, etc.
- May not protect against all internal threats
 - Ex: A public or personal proxy server, VPN service, TOR network, etc.
- Cannot protect against rogue wireless access points
- Infected USB, personal laptop, or mobile device may be brought in and attached to the network

Firewall Limitations (Cont.)

- Also, most firewalls block unknown inbound traffic but allow returning traffic after an outbound request
- This is why exploit kits are so dangerous
 - The user visits the page and the firewall allows the returning traffic just like any other web page

Two Types of Firewalls

1. Packet Filters:

- Stateless packet filtering firewall
- Stateful inspection firewall

2. Proxy Servers:

- Application proxy firewall
- Circuit-level proxy firewall

Packet Filters – General Traits

- Can either:
 - Allow packets that meet defined criteria to pass through the firewall
 - Reject packets that meet defined criteria to be blocked from passing through the firewall
- May examine the following attributes of a packet:
 - Protocol headers
 - Payload
 - Pattern generated by a sequence of packets

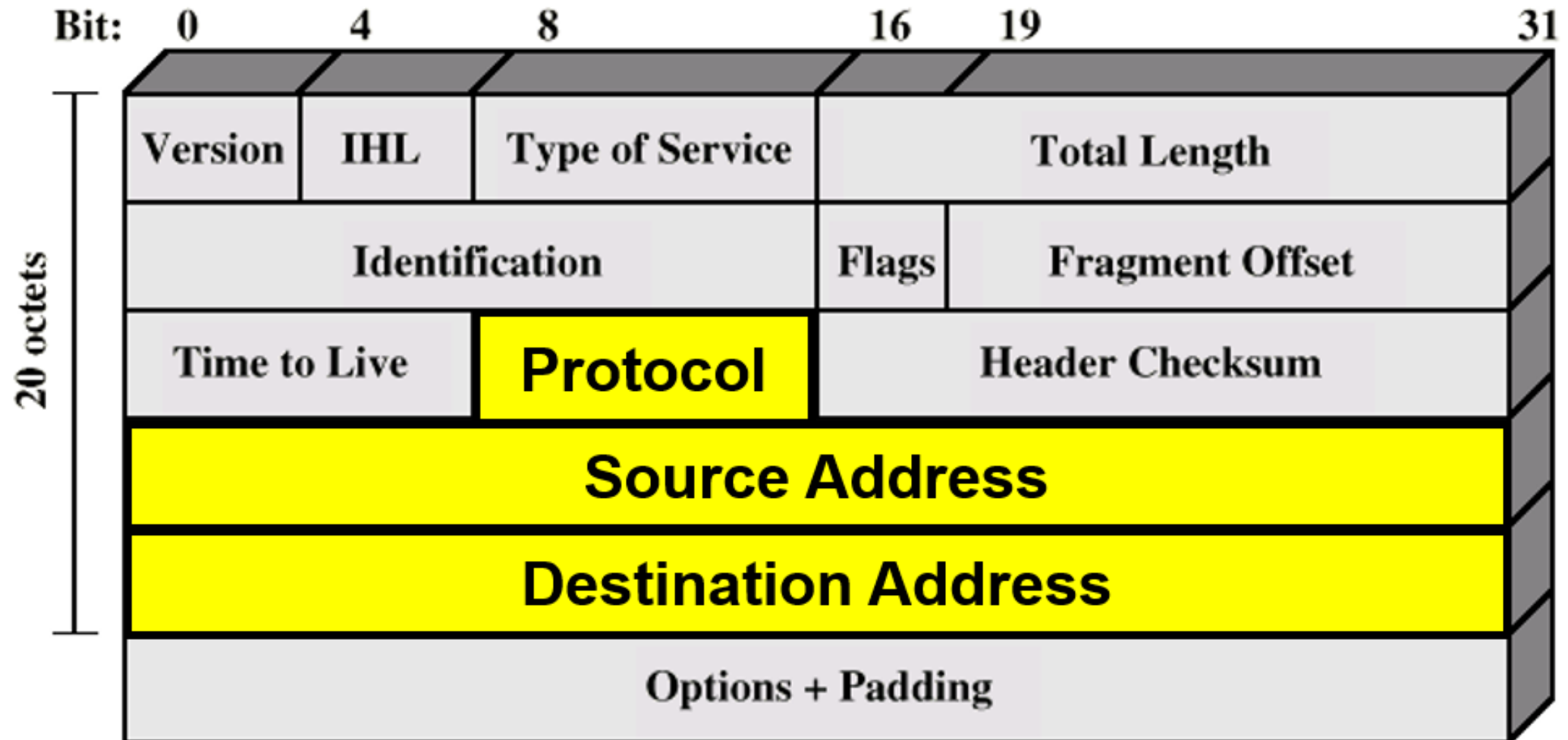
Part II

Stateless Packet Filters

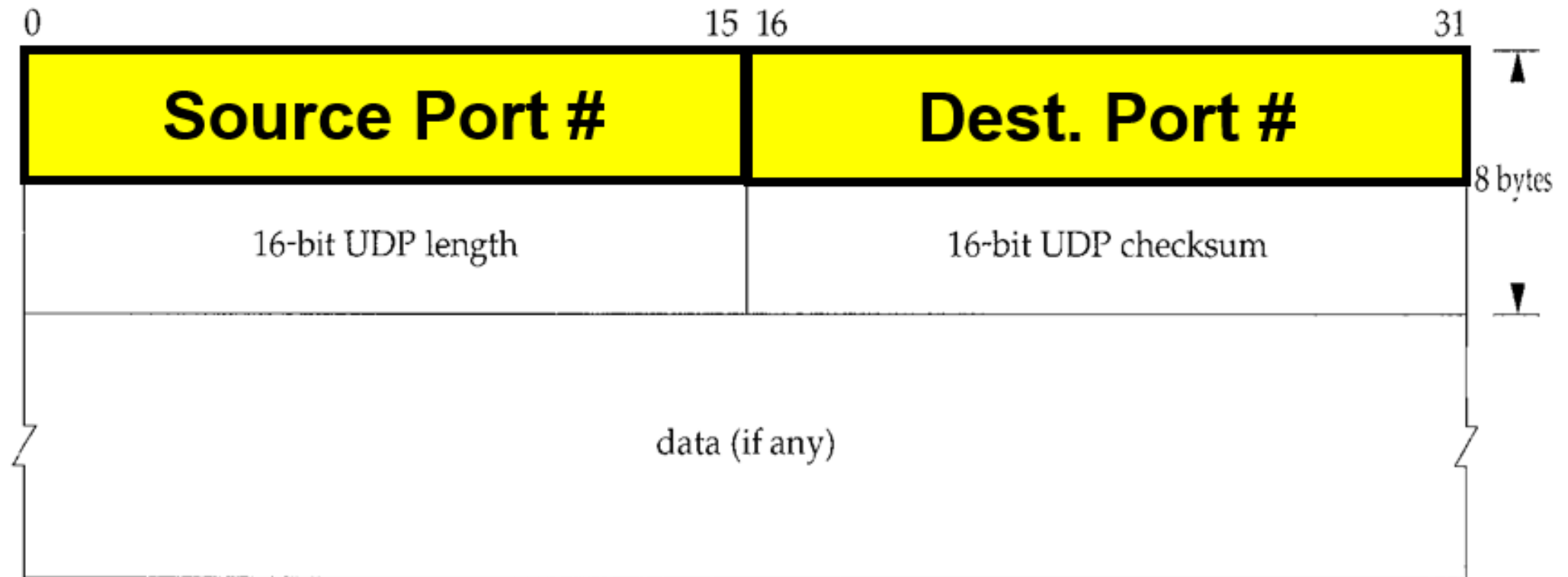
Stateless Packet Filtering Firewall

- Applies rules to each incoming and outgoing packet based on:
 - IP Header
 - TCP/UDP Headers
 - Firewall Interface

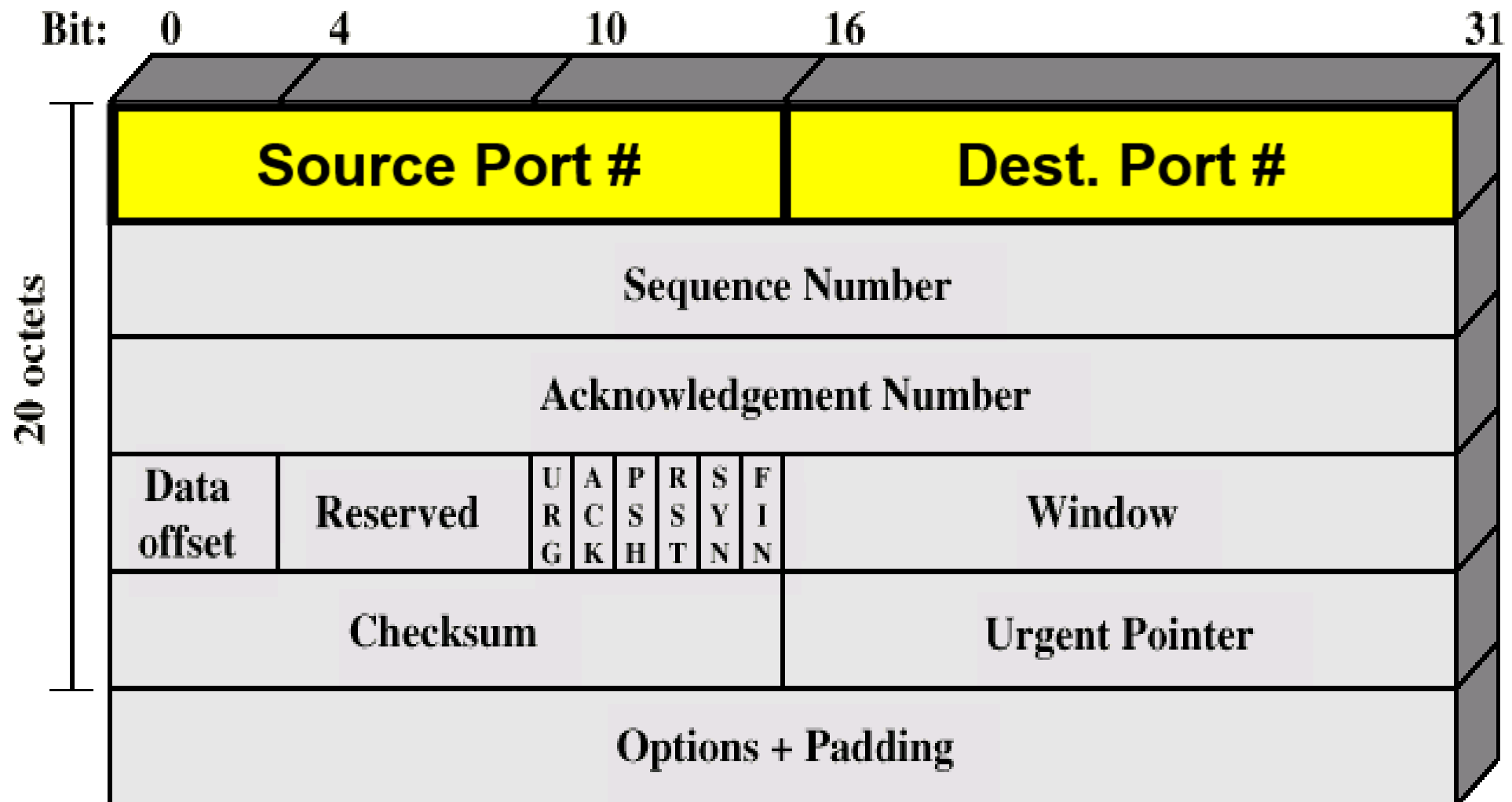
IP Header Fields Always Considered by Packet Filters



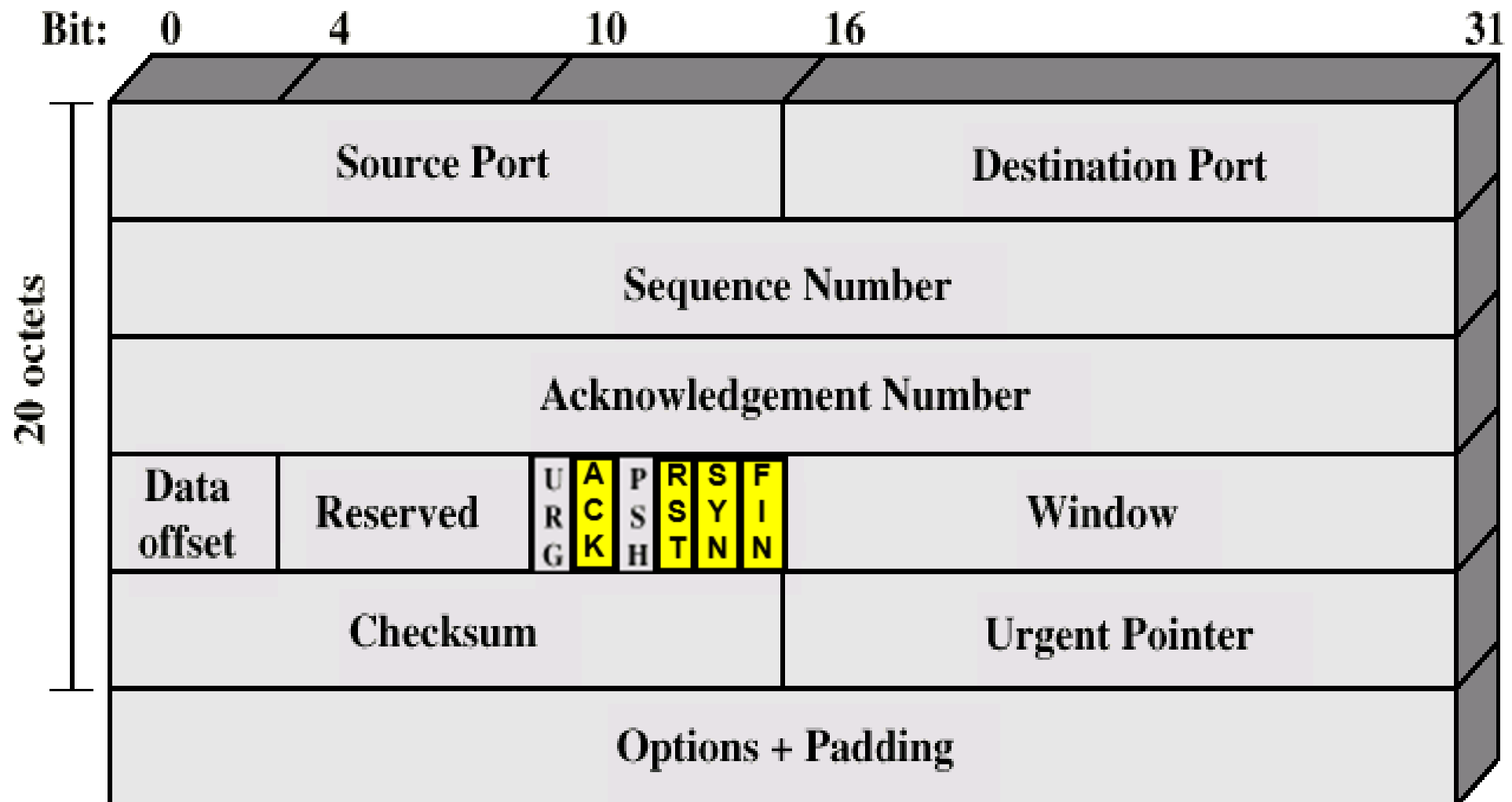
UDP Header Fields Always Considered by Packet Filters



TCP Header Fields Always Considered by Packet Filters



TCP Header Fields Considered by More Sophisticated Packet Filters



Packet Filter: Default Policy Choices

- Implicit Deny
 - All traffic is implicitly blocked unless you explicitly allow it
- Implicit Allow
 - All traffic is implicitly allowed unless you explicitly block it

Rule Creation

- Rules may be implicit or explicit:
 - Implicit
 - Firewall has this rule be default
 - No logging occurs
 - Explicit
 - This rule must be created
 - Logging occurs

Most Popular Default Choice

- Implicit Allow Policy to accept all traffic by default
- Add explicit rules allowing certain traffic to enter
- Add explicit deny at bottom of rule chain to reject all other traffic that isn't explicitly allowed in the prior rules
- Why is this better than just using Implicit Deny and then adding explicit entry rules for traffic???

Most Popular Default Choice (Cont.)

- If you accidentally flush (delete) all of your firewall rules, Implicit Allow will let you ssh back into the firewall and fix your mistake
- If you have Implicit Deny set and accidentally flush all of your explicit allow rules, you will not be able to ssh in and will have to drive to the site and logon to the firewall console via serial port

Examples

- The next few slides show a firewall with an implicit allow rule for all traffic
- Explain what each of the explicit rules do

Stateless Packet Filter Rules: Example 1

Rule Order	Direction	Protocol	Source IP	Source Port	Dest. IP	Dest. Port	Action
1	IN	TCP	ANY	ANY	192.168.1.5	22	ALLOW
4	IN	ANY	ANY	ANY	ANY	ANY	DROP
5	OUT	TCP	192.168.1.5	22	ANY	ANY	ALLOW
8	OUT	ANY	ANY	ANY	ANY	ANY	DROP

Explanation

- The SSH Server at 192.168.1.5:
 - Accepts inbound connections on its port 22 from any IP address and port on the Internet
 - Allows outbound communications emanating from port 22 to any IP address and port on the Internet

Stateless Packet Filter Rules: Example 2

Rule Order	Direction	Protocol	Source IP	Source Port	Dest. IP	Dest. Port	Action
1	OUT	TCP	192.168.1.0 /24	ANY	ANY	80	ALLOW
2	OUT	TCP	192.168.1.0 /24	ANY	ANY	443	ALLOW
3	OUT	UDP	192.168.1.0 /24	ANY	ANY	53	ALLOW
4	OUT	ANY	ANY	ANY	ANY	ANY	DROP
5	IN	TCP	ANY	80	192.168.1.0 /24	ANY	ALLOW
6	IN	TCP	ANY	443	192.168.1.0 /24	ANY	ALLOW
7	IN	UDP	ANY	53	192.168.1.0 /24	ANY	ALLOW
8	IN	ANY	ANY	ANY	ANY	ANY	DROP

Explanation

- All clients on the subnet 192.168.1.0/24:
 - Can connect to any web server on the Internet via ports 80 and 443
 - Can query any DNS server on the Internet via port 53
 - Can receive replies from any server on the Internet that uses ports 80 and 443
 - Can receive responses from any DNS server on the Internet that uses port 53

Question???

- Is the prior web usage firewall rule set secure???
- Why or why not???

Answer

- Is the prior web usage firewall rule set secure???
 - No
- Why not???
 - Any system on the internet that uses port 80, 443, and 53 can attempt to hammer our host with a DoS
- What do we do to stop that???
 - We need to ensure that the firewall doesn't accept new inbound traffic and only accepts traffic in response to an established connection
 - **A stateful firewall does this which we'll cover in the next section!**

Attacks Against Stateless Packet Filters

- IP address spoofing
 - Defense is to discard packets with the source IP of an internal host that arrives on an external interface
- Source routing attacks
 - Allows sender to specify route packet will take through the network
 - Defense is to drop all packets that use source routing
- Tiny fragment attacks
 - Defense is to enforce rule that first packet must contain minimum amount of the transport header, otherwise it gets rejected

Last Note on Stateless Packet Filters

- Most do not look at the packet headers for:
 - TCP flags (SYN, ACK, FIN, RST bits)
 - IP flags (M bit)
- Some that are more sophisticated do, but attacks can still get through
- Stateless Packet Filters also have a harder time preventing network scanning

Part III

Stateful Packet Inspection Filters

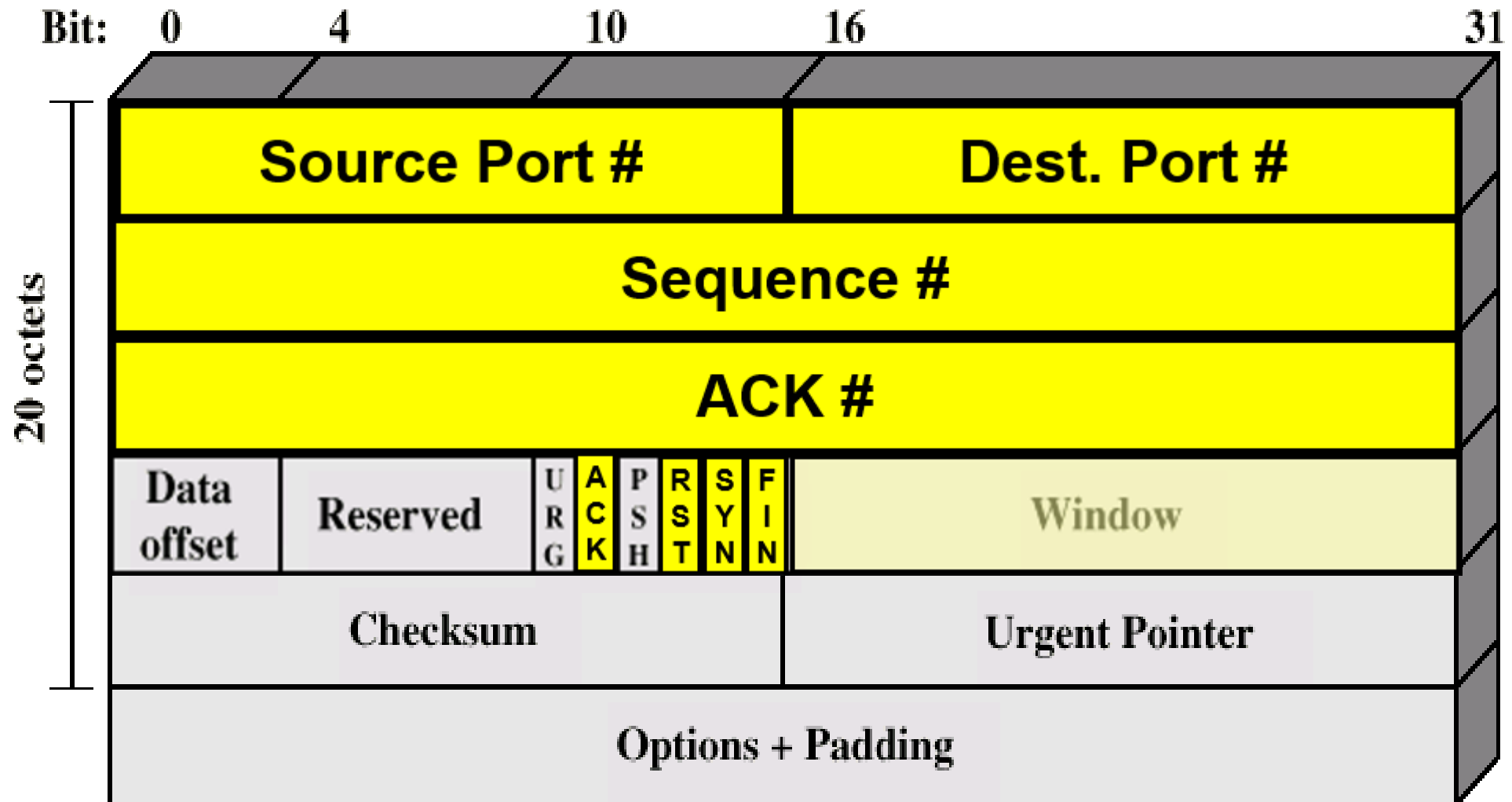
Stateless vs Stateful Packet Filters

- Stateless (Static):
 - Evaluates contents of each packet but doesn't keep track of the connection state
 - Similar to an Access Control List (ACL)
 - Low overhead / High throughput
 - Inexpensive
- Stateful (Dynamic):
 - Keeps track of connection state in order to make decisions
 - Higher overhead / Lower throughput
 - More costly

Stateful Packet Filtering Firewall

- Applies rules to each incoming and outgoing packet based on:
 - IP Header
 - TCP/UDP Headers
 - Evaluates Sequence and ACK #s
 - Remembers connection state (TCP 3-Way Handshake)
 - Firewall Interface

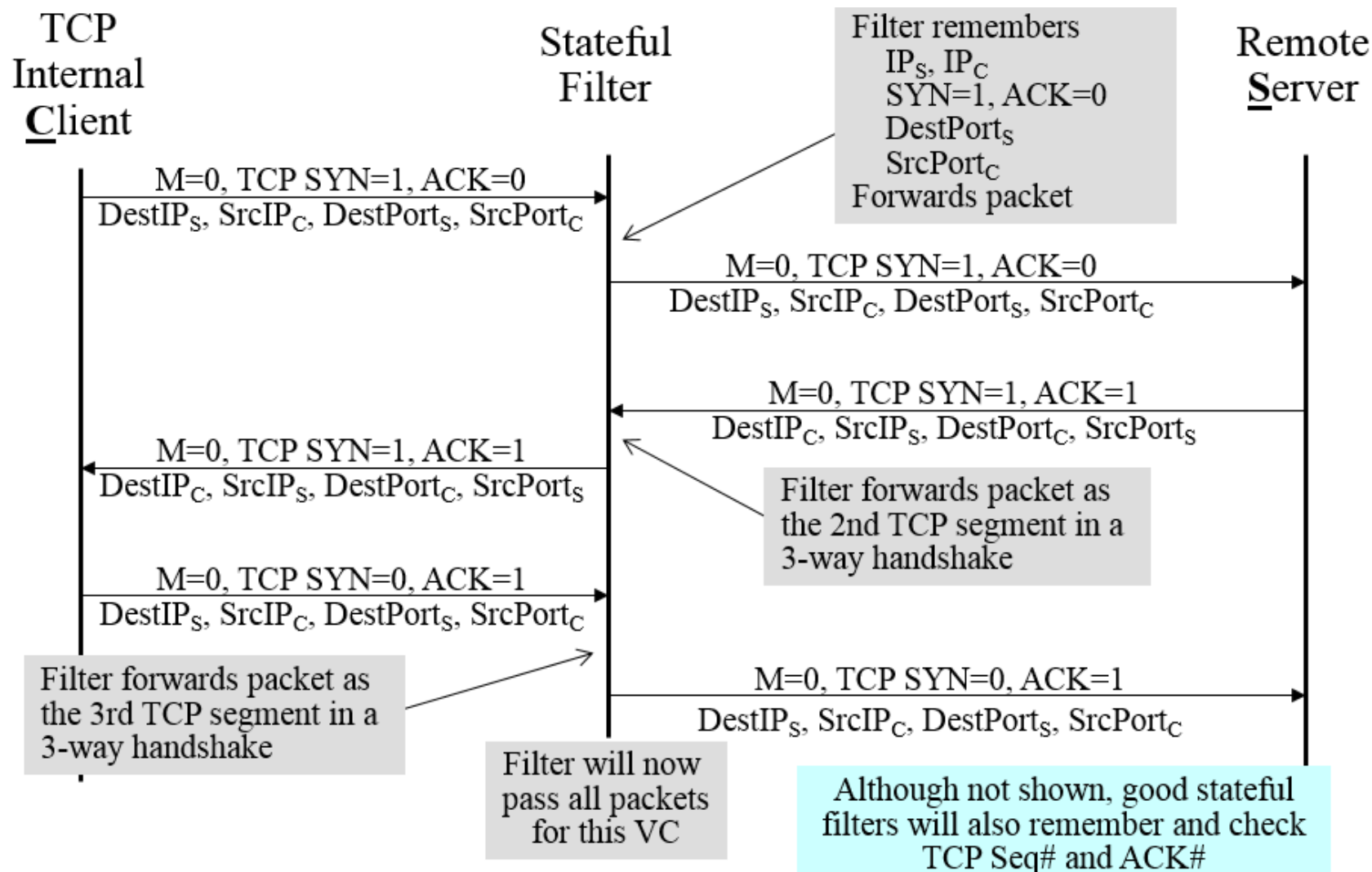
TCP Header Fields Often Considered by Stateful Filters



Stateful Firewall Connection State Table

Source Address	Source Port	Destination Address	Destination Port	Connection State
192.168.1.100	1030	210.9.88.29	80	Established
192.168.1.102	1031	216.32.42.123	80	Established
192.168.1.101	1033	173.66.32.122	25	Established
192.168.1.106	1035	177.231.32.12	79	Established
223.43.21.231	1990	192.168.1.6	80	Established
219.22.123.32	2112	192.168.1.6	80	Established
210.99.212.18	3321	192.168.1.6	80	Established
24.102.32.23	1025	192.168.1.6	80	Established
223.21.22.12	1046	192.168.1.6	80	Established

Stateful Packet Filters – TCP Example



Stateful Packet Filters – TCP Example

- The previous example allows TCP connections to be setup from any internal host to any remote host if it initiated by the internal host
- The filter might drop packets such as
 - TCP connections initiated by remote hosts
 - ACK scans
 - SYN/ACK scans
 - RST scans
 - ...

Stateful Packet Filters – UDP, ICMP Comments

- Stateful filtering is not usually useful for UDP
 - UDP itself is stateless
- Stateful filtering can be useful for passing ICMP echos
 - How?
 - Must be capable of looking into ICMP header
 - Echo request remembered and passed
 - Echo reply only allowed back if SRC_{IP} and DST_{IP} match but are reversed from that of the echo request

Stateful Packet Filters – Preventing Network Applications

- Many stateful packet filters will keep state for network applications such as TELNET, HTTP, FTP, and NetBIOS
- This is not filtering based upon port numbers
- This is interference with operation of the application level protocol
- Lets look at FTP as an example

Original (Active) FTP Example

The original FTP client/server protocol requires 2 server and 2 client ports and 2 TCP connections

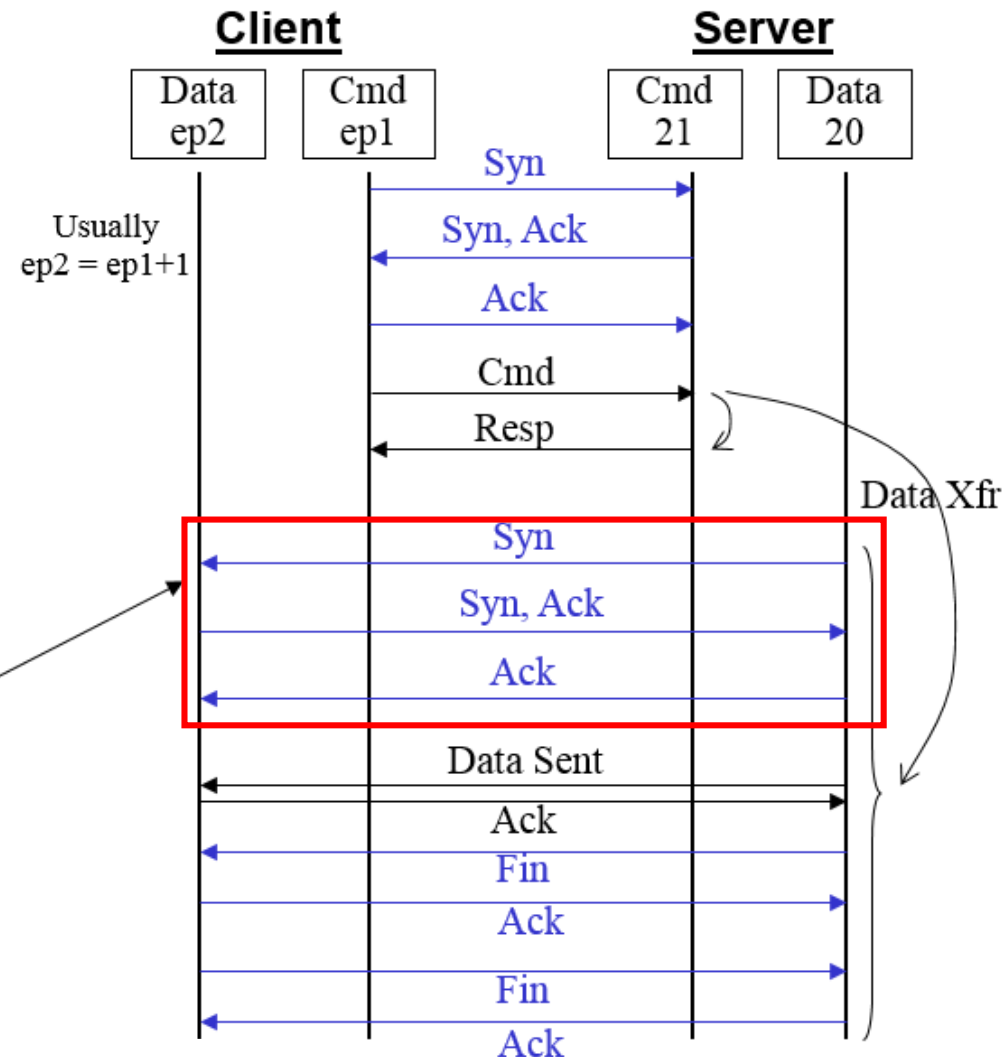
Command connection:

Initiated by client

Data connection: Initiated by server

Called "Active FTP"

But firewalls will usually block TCP connections from outside
Preventing Active FTP from working

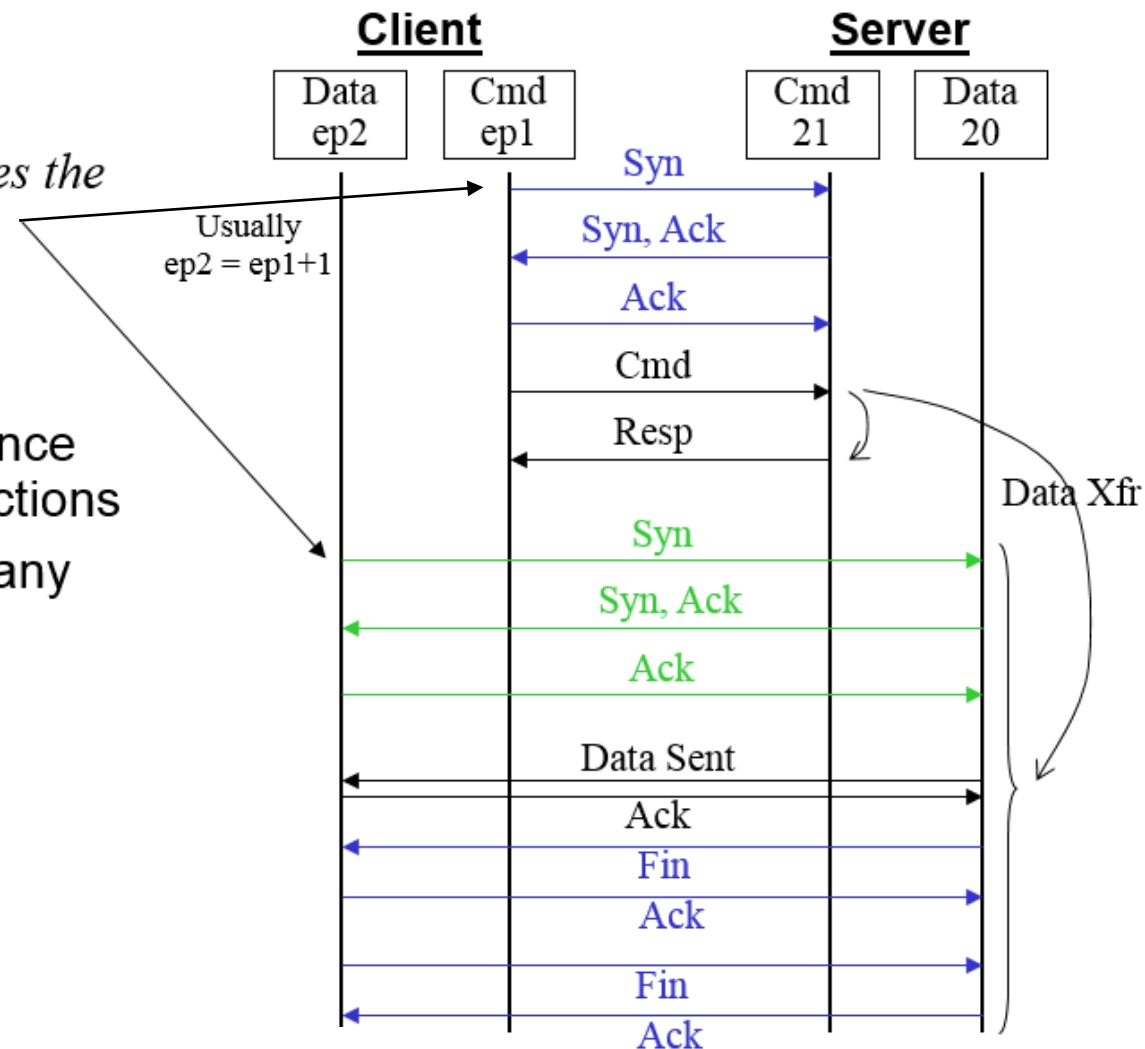


Passive FTP Example

Passive FTP evolved as a solution

Client (not Server) initiates the Data connection

Avoids Firewall blocking since client makes both connections
But server is now open to any remote connection



Network Applications and Stateful Packet Filters

- In FTP example, a stateful firewall can allow either form of FTP to work with reasonable safety if
 - Firewall keeps history on two items
 - ftp **PORT** commands that go across the control connection, and
 - Both IP addresses
- But to do this the Stateful Packet Filter must have knowledge of the application
- It must start to have some of the properties of a Proxy Server
 - But it doesn't terminate TCP VCs

Firewall Topologies

host-resident firewall

- includes personal firewall software and firewall software on servers

screening router

- single router between internal and external networks with stateless or full packet filtering

single bastion inline

- single firewall device between an internal and external router

single bastion T

- has a third network interface on bastion to a DMZ where externally visible servers are placed

double bastion inline

- DMZ is sandwiched between bastion firewalls

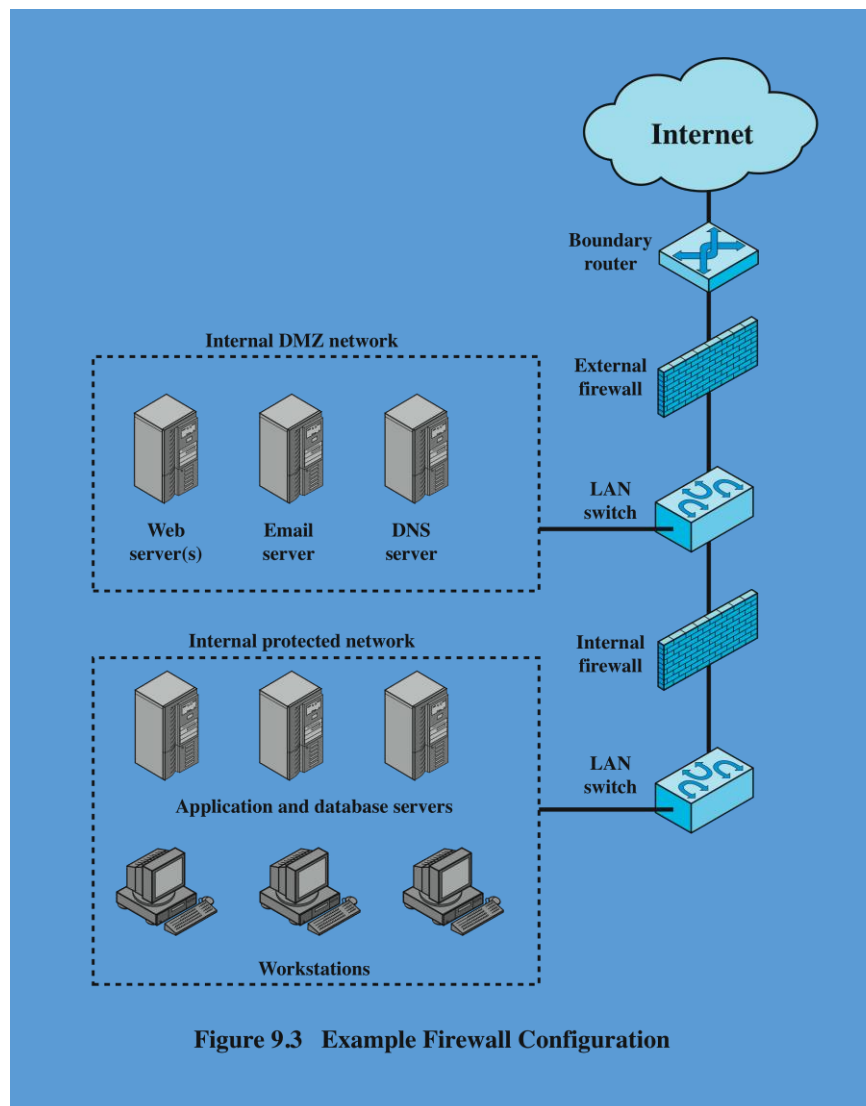
double bastion T

- DMZ is on a separate network interface on the bastion firewall

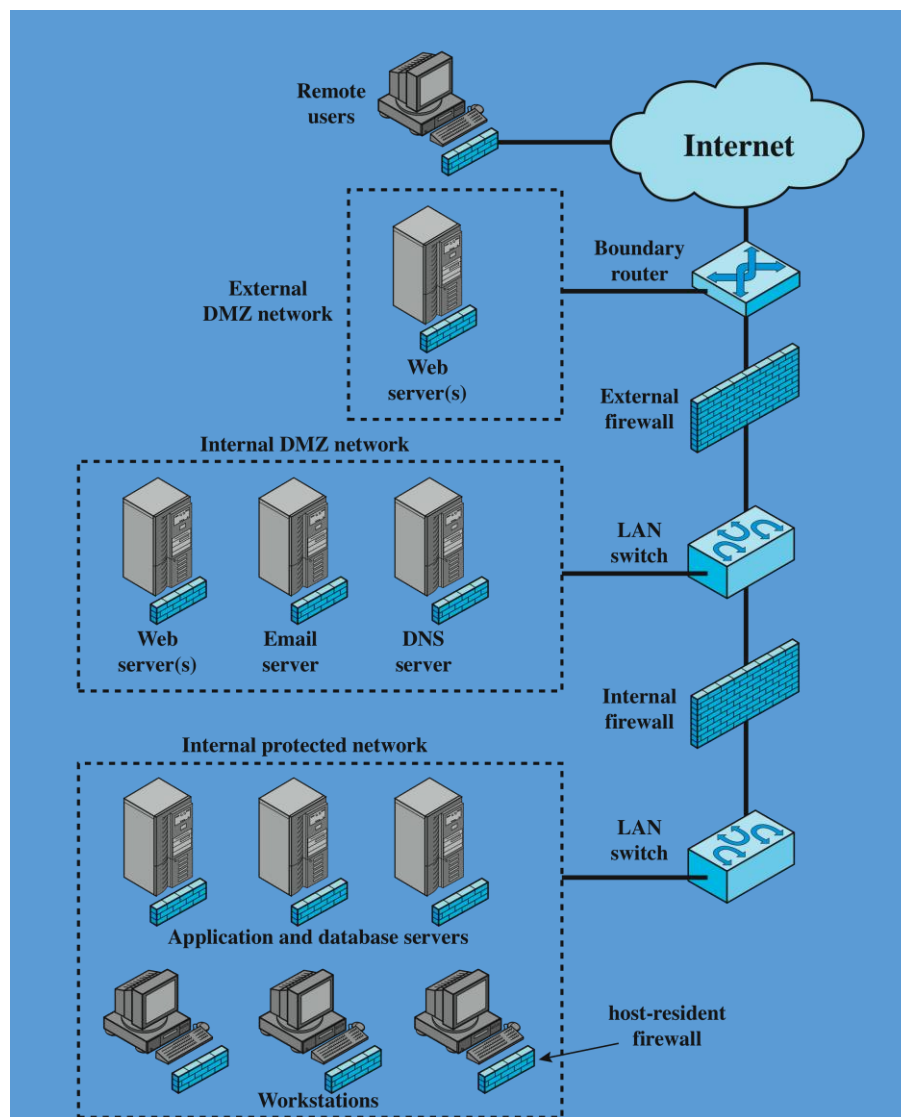
distributed firewall configuration

- used by large businesses and government organizations

Firewall Placement: Example 1



Firewall Placement: Example 2



Part IV

Application-Level & Circuit-Level Proxy Servers

Proxy Server

- Proxy Server (or Application Gateway) acts as surrogates for the real servers at a network site
- Operates at the application layer of the reference model
- Terminates TCP connections both from internal hosts and external hosts, whether client or server
 - Changes IP addresses, port numbers, etc.
 - No direct connection between client and server

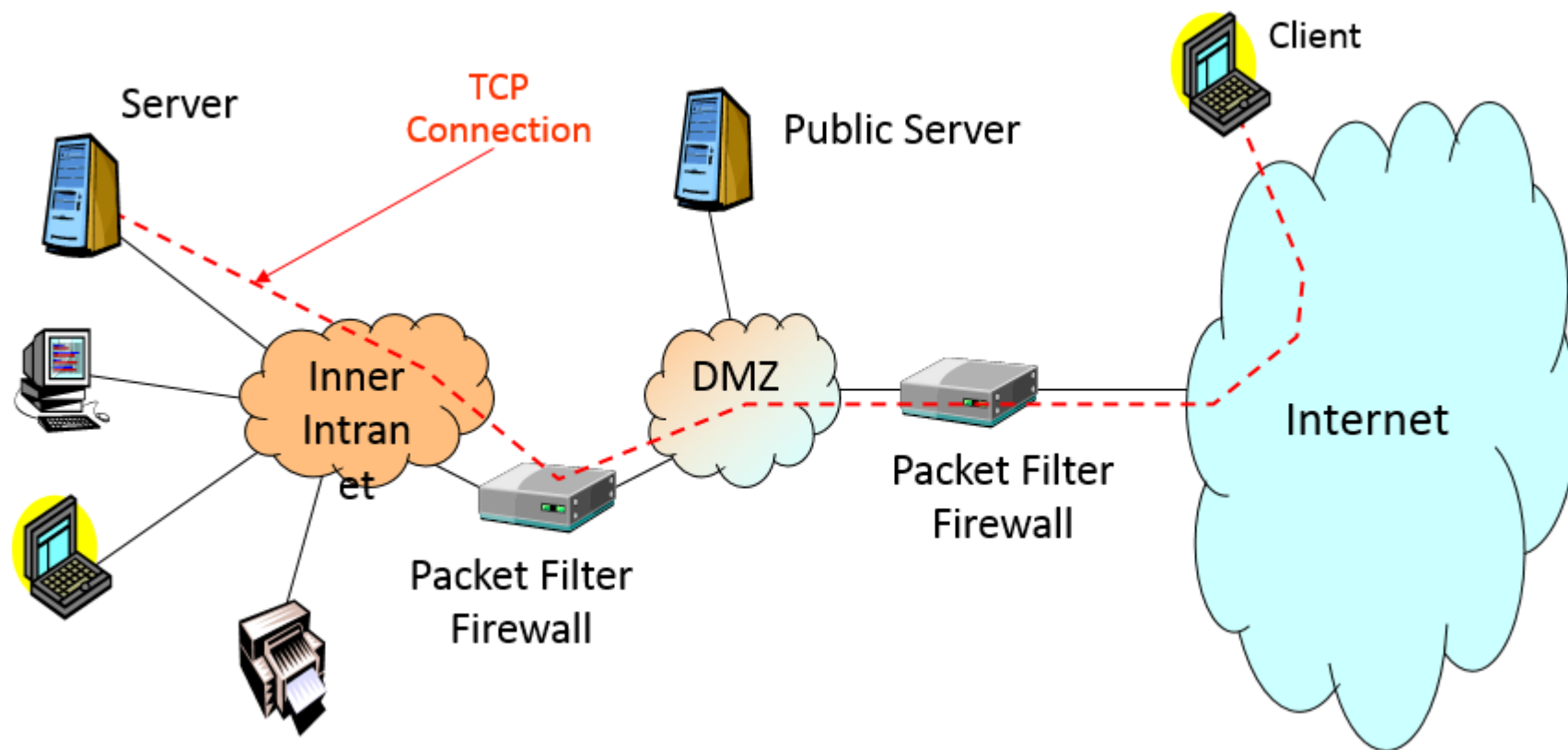
Proxy Server Purpose & Function

- Isolate the client and server from each other
 - Thus greatly reducing possibility of hacking either client or server when they are only allowed to interact through the proxy server
- Permits or denies connection based upon what the client or server is trying to do
- Proxy Servers
 - "Look" into the packet payloads
 - Understand the contents of the payloads in terms of the intended applications

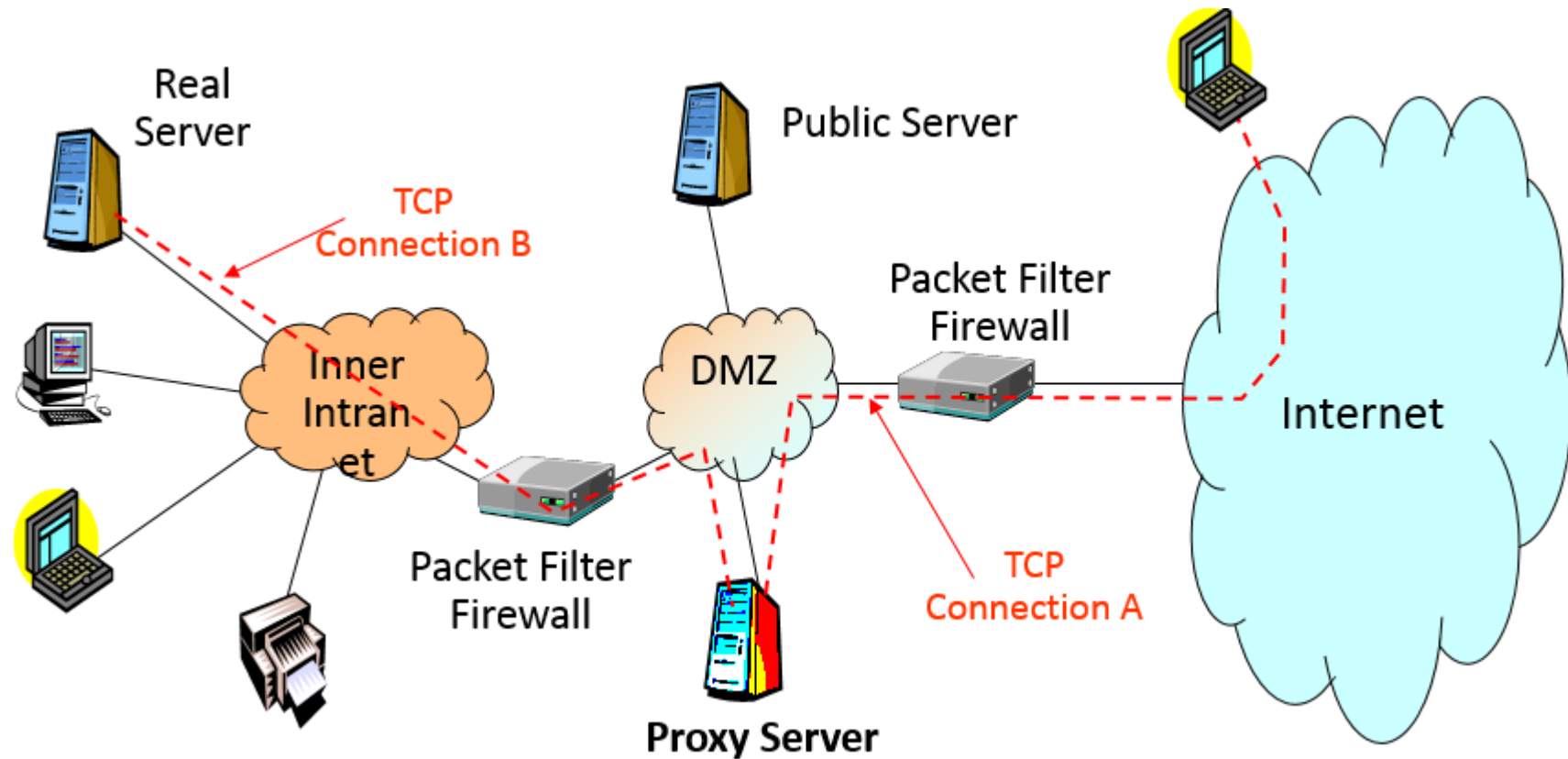
Contrast Packet Filters and Proxy Servers

- Client/Server Connections
 - Packet filters allow direct connections
 - Proxy servers prevent direct connections
 - TCP: Instead the Proxy Server splices two TCP connections together
 - UDP: Proxy Server investigates the payload
- Application
 - Packet filters do not understand the application or service
 - Or mostly do not
 - Proxy servers do understand the application or service

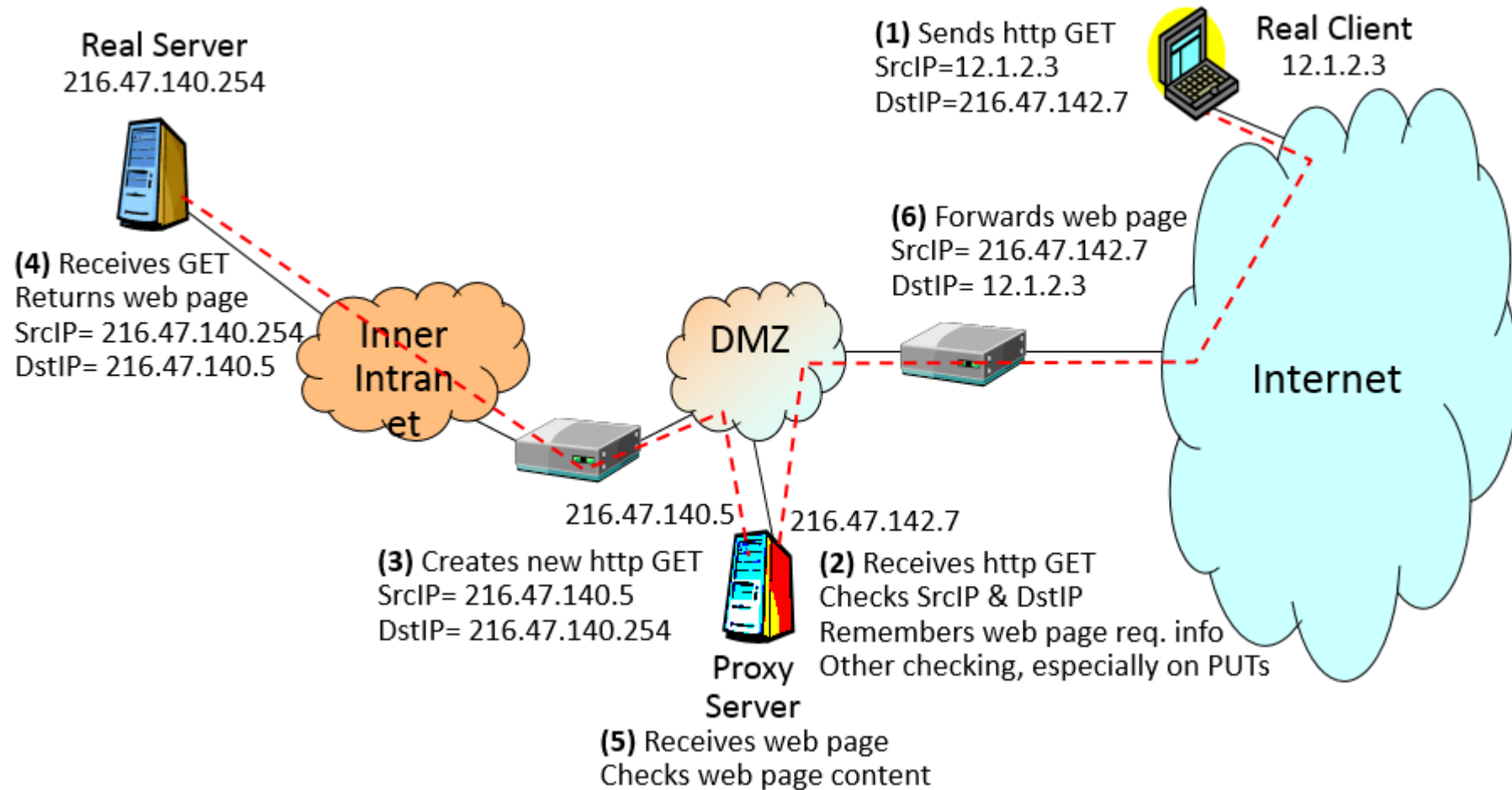
TCP Connection Without Proxy Server



TCP Connection With Proxy Server



Proxy Server Example



Proxy Server *Comments*

- Substantially increases security
- Proxy servers must handle both directions for both clients and servers
- Example showed protection of both client and server
 - Server protected from hackers posing as clients
 - Hacker prevented from using such commands as http PUT and POST to infect server
 - Client protected from being infected by infected web pages being served

Proxy Server *Pros*

- Both clients and servers are invisible to each other
 - They don't know each others IP addresses
 - Cannot know each other's internal networks or host file structures
- Proxy server can be used to log all information by application and possibly intent
 - Can log occurrences of malicious application code

Proxy Server *Pros* (Cont.)

- Proxy server can do content filtering
 - Can detect occurrences of malicious application code
 - Search for keywords and phrases (e.g., obscenities)
 - Filter Java applets and Active-X controls
- Real servers don't need to be as securely configured
 - Although its a good idea anyway

Proxy Server *Cons*

- Cost, both initial and ongoing
- Multiple proxies
 - A substantial issue for organizations that have many different types of applications
- Single point of failure
 - The proxy server becomes a really critical resource

Proxy Server *Cons* (Cont.)

- Single point for hacking
 - A proxy server, if hacked, can be a gateway for the hacker
 - Thus a proxy server **MUST** be securely configured
 - In the past available proxy software was often not written to be secure
 - Proxy software was originally written for caching web pages
 - Often could be easily hacked

Circuit-Level Proxy Servers

- Sets up two TCP connections
 - One between itself and a TCP user
 - One is on inner host and one is on outer host
 - Relays TCP segments from one connection to the other without examining contents
 - Security function consists of determining which connections will be allowed
- *Typically used when inside users are trusted

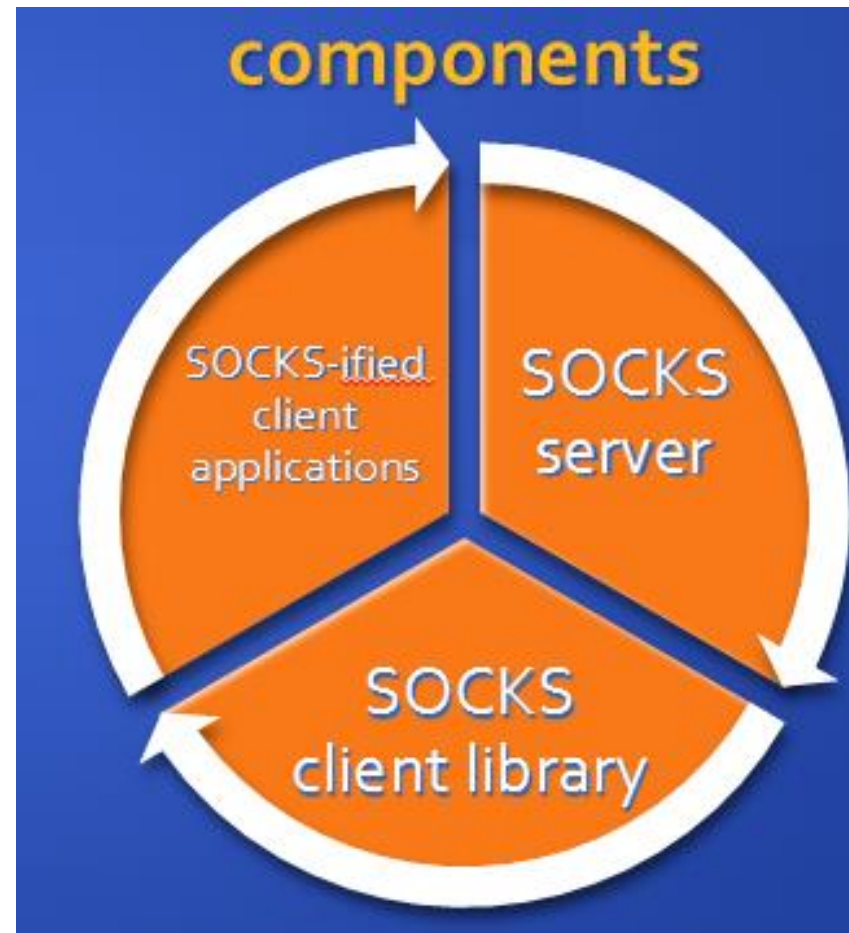
Circuit-Level Proxy Servers (Cont.)

- Has lower overhead
- May use Application-Level Proxy inbound and Circuit-Level Proxy outbound

SOCKS Circuit-Level Proxy

- SOCKS V5 defined in RFC1928
- Framework for client-server applications in TCP/UDP domains to conveniently and securely use the services of a network firewall
- Client application contacts SOCKS server, authenticates, and sends relay request
 - Server evaluations the request
 - Establishes or denies the connection

SOCKS Circuit-Level Proxy



Circuit-Level Proxy Example: Torsocks

- Allows SOCK-ified client applications to use the TOR network
- Works on Linux/Mac OSX
- Example:
 - **`usewithtor wget www.randomsite.com`**

Circuit-Level Proxy Example: Torsocks

- SOCK-ified client applications compatible with usewithtor:
- Administrative:
 - ssh, telnet, svn, gpg
- Messaging:
 - pidgin, kopete, conversation, irssi, silc
- Email:
 - claws-mail, thunderbird
- File transfer:
 - wget, ftp

Circuit-Level Proxy Example: Torsocks

- Some of the SOCK-ified client applications leak DNS requests and/or other information
- See <https://code.google.com/p/torsocks/> for more details

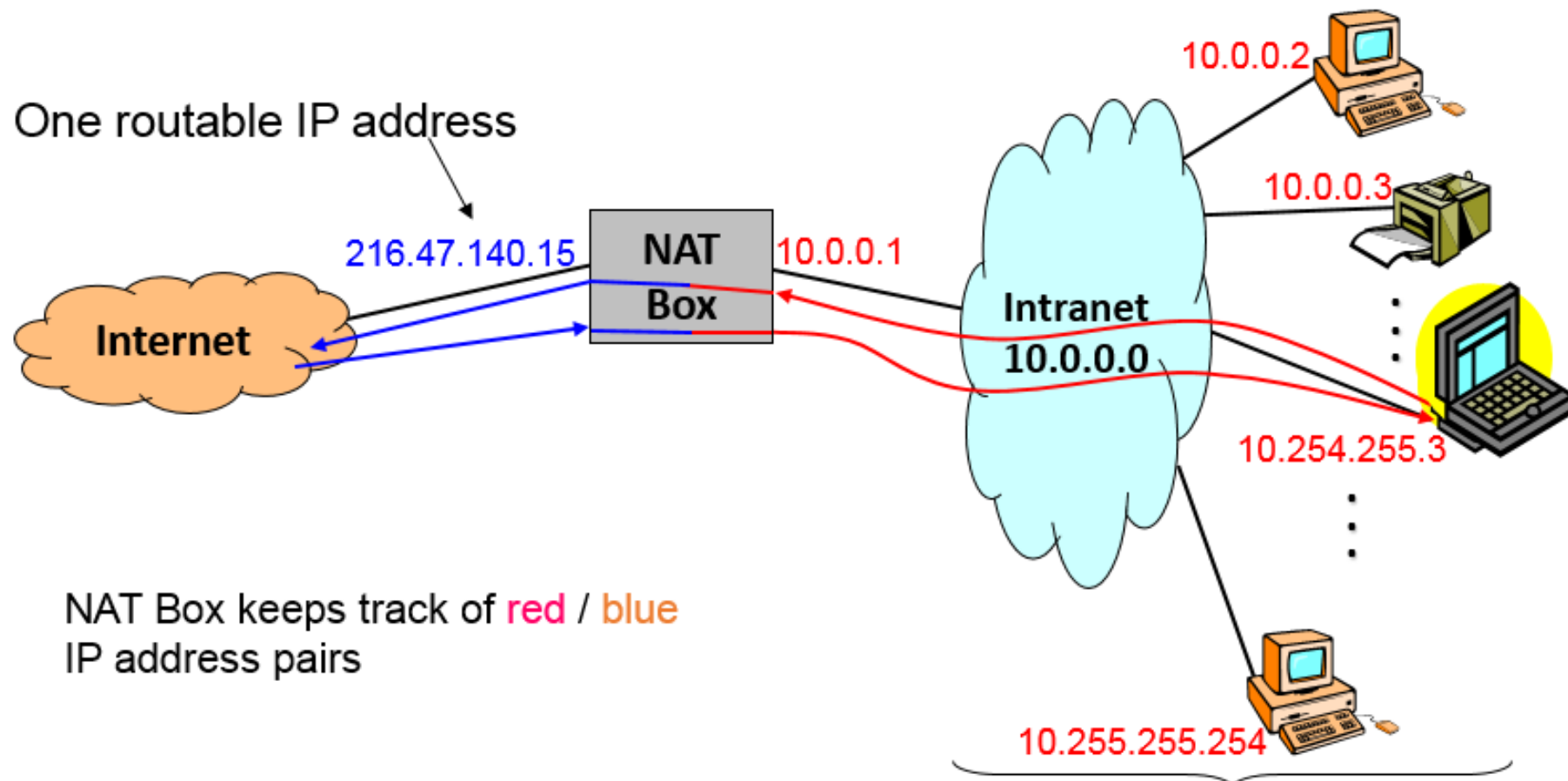
Part V

NAT

NAT Function

- Network Address Translation (NAT) is a function that translates all local Intranet IP addresses to a single Internet address and visa versa

NAT Example

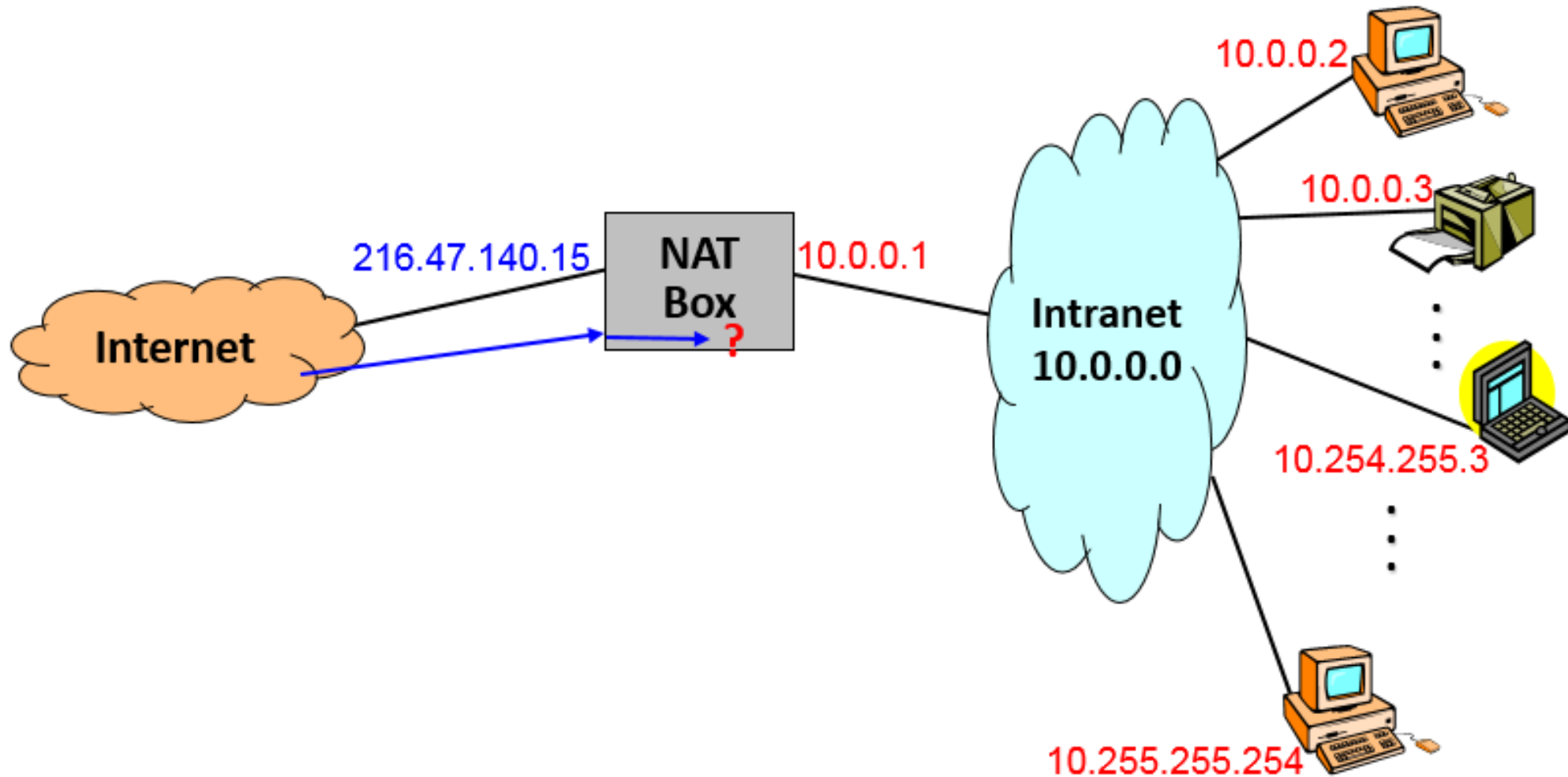


Up to $2^{24} - 2 = \sim 17$ million non-routable
i.e., private IP addresses

NAT Box as a Firewall*

- Can Network Address Translation by itself be partially effective as a firewall?
 - If so, why?

NAT as a Partial Firewall*



NAT Box as a Firewall*

- Remote hosts on the Internet don't know what addresses are behind the NAT Box

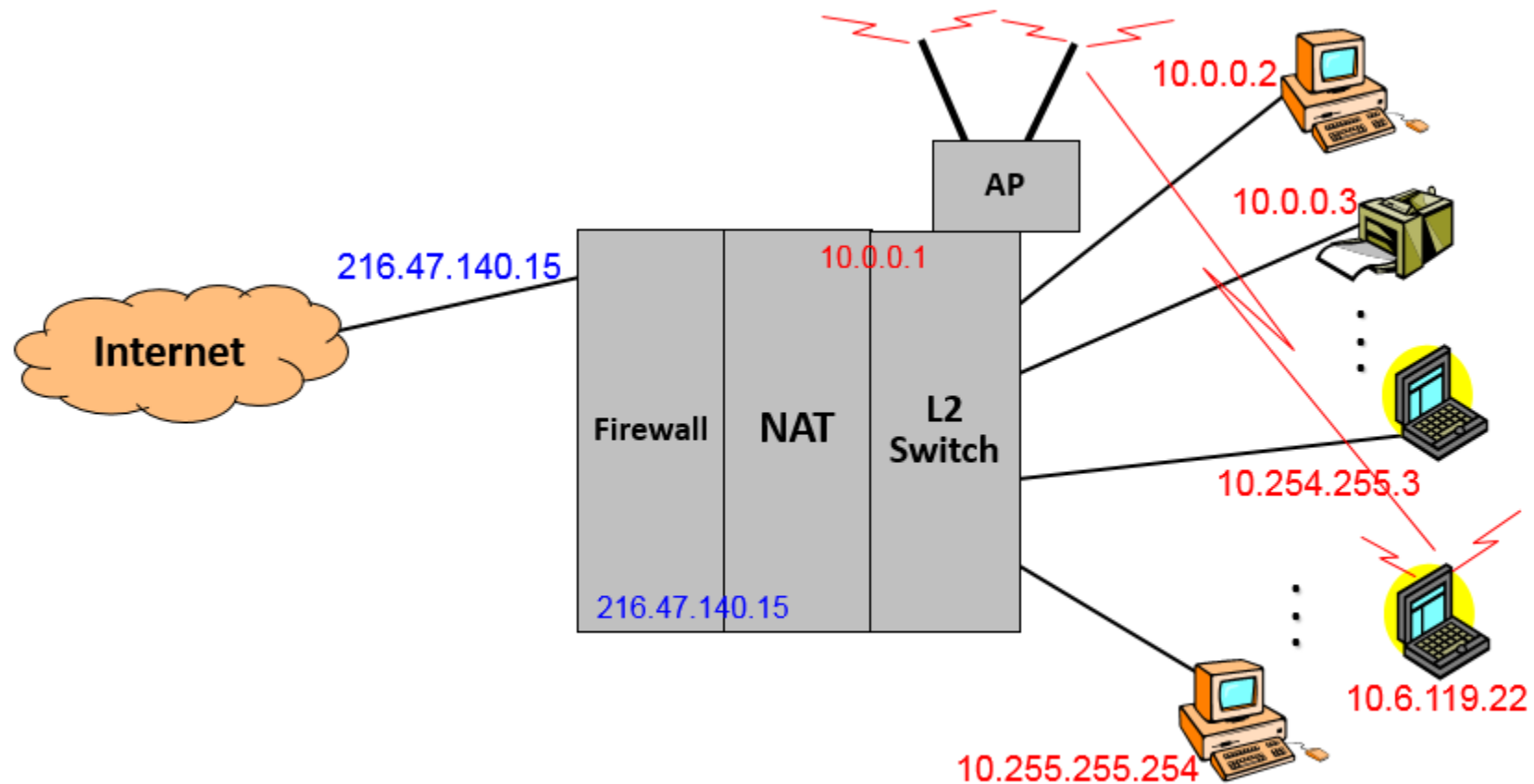
NAT Box as a Firewall*

- But it's reasonable to assume that there might be private (aka non-routable) addresses behind a NAT box. Would this do the attacker any good?
 - Non-routable IP addresses should be dropped by properly configured routers
 - So if an attacker tries to target a private address, routers should drop the attack packet(s)
 - Therefore the attack packet will never get to the target
- But unfortunately routers are sometimes improperly configured to route the non-routable IP addresses

NAT Box as a Firewall*

- Must NAT Boxes be stateful?
 - Yes!
- How can malware get into the Intranet through a NAT Box?
 - Via packet payloads such as email, ftp etc.
 - Also TCP hijacking

What's Available Today



NAT Box as a Firewall*

- Let's say you are running a web server and network camera behind a NAT box on ports 80 and 4455 respectively
- They both therefore have private IP addresses
- How would a remote user be able to connect to either of those services?
 - Configure Port Forwarding on the NAT box

Part VI

Hands On: iptables

OVA

- You should have already copied the xubuntu1404_64bit.ova file to your desktop from the below share, imported it into vbox, and started the VM.
 - <\\coulson.otsads.iit.edu\itm448>
- Logon = admin2
- Password = admin2

OVA (Cont.)

- Double-click on the OVA file on the desktop after it has finished copying
- When Vbox opens, select “Import”
- Start the VM after import (Don’t worry about the BIOS message)
- Logon = admin2
- Password = admin2

Iptables Exploratory Lab (Advanced Students)

- Create firewall rules that:
- Allow only established connections on ports 443, 80, and 53 for inbound traffic
- Allow new and established connections on ports 443, 80, and 53 for outbound traffic

Note

- Open the Firefox Browser
 - Top Left Blue Icon / Web Browser
- Open the Terminal Emulator

Note

- Normally, you would need to download an application called conntrack that will keep track of the state of the connections
- This application has already been downloaded for you in the VM.

Iptables Config/Syntax Overview

- iptables is a Linux based firewall which uses rules to process packets coming and going from your system.
- Hierarchy is:
 - Tables (lower case name)
 - Chains (upper case name)
 - Rules

Iptables Config/Syntax Overview

- 4 Tables:
 - **Filter Table**
 - Default firewall table you will use the most to filter packets
 - **NAT Table**
 - For routing of packets
 - **Mangle Table**
 - Special packet alteration
 - **Raw Table**
 - For configuration exceptions

Iptables Config/Syntax Overview

- Filter Table has three chains:
 - INPUT
 - Packets coming in
 - OUTPUT
 - Packets going out
 - FORWARD
 - Packets routed through two network interfaces on the local server. Example: eth0 to eth1

Iptables Config/Syntax Overview

- xubuntu ships with the iptables firewall
- No rules set by default, security wide open.
- Pull up your terminal
- **sudo iptables -t filter --list**
 - Shows Filter table and three chains.
 - They are all blank which means there are no rules set by default.

Iptables Config/Syntax Overview

- We will focus on the Filter table but to see the rules for other tables use:
 - **sudo iptables -t nat --list**
 - **sudo iptables -t mangle --list**
 - **sudo iptables -t raw --list**
- An easier way to see the rules for the default filter table (which we will be using) is with:
 - **sudo iptables -L**

Iptables Config/Syntax Overview

- Rules have actions.
- Main actions are:
 - ACCEPT
 - Packet is accepted through the firewall into the system.
 - REJECT
 - Packet is rejected and ICMP or TCP response is returned to sender.
 - DROP
 - Packet is rejected and no response is returned.

Iptables Config/Syntax Overview

- Iptables is stateful
- conntrack module:
 - Tracks state of packets/connection
- --ctstate
 - List of the connection states to match

States for --ctstate

INVALID

meaning that the packet is associated with no known connection

NEW

meaning that the packet has started a new connection, or otherwise associated with a connection which has not seen packets in both directions, and

ESTABLISHED

meaning that the packet is associated with a connection which has seen packets in both directions,

RELATED

meaning that the packet is starting a new connection, but is associated with an existing connection, such as an FTP data transfer, or an ICMP error.

UNTRACKED

meaning that the packet is not tracked at all, which happens if you use the NOTRACK target in raw table.

SNAT

A virtual state, matching if the original source address differs from the reply destination.

DNAT

A virtual state, matching if the original destination differs from the reply source.

Iptables Config/Syntax Overview

- Now, let's create a few rules
- In the Firefox Browser, verify you can connect to web.iit.edu
- Let's now set an explicit DROP for all incoming traffic
- **`sudo iptables -A INPUT -i eth0 -j DROP`**

Iptables Config/Syntax Overview

- Now, let's make sure that rule was created
- **sudo iptables -L**



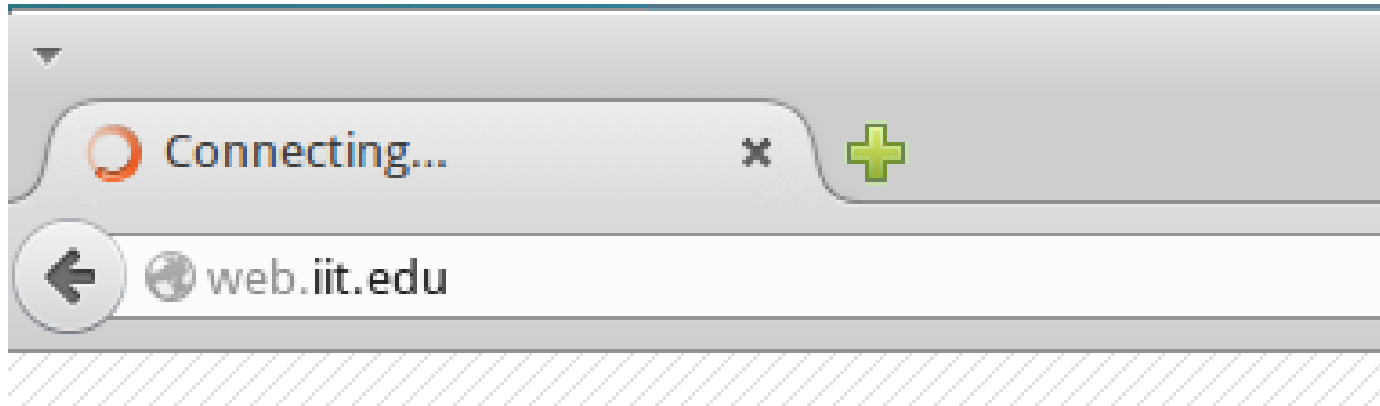
```
admin2@admin2-VirtualBox:~$ sudo iptables -L
Chain INPUT (policy ACCEPT)
target      prot opt source                destination
DROP        all  --  anywhere               anywhere

Chain FORWARD (policy ACCEPT)
target      prot opt source                destination

Chain OUTPUT (policy ACCEPT)
target      prot opt source                destination
```

Iptables Config/Syntax Overview

- Now, try to connect to www.iit.edu again
- What do you see???



- Will eventually time out but go ahead and click the X at the right side of the address bar to stop the attempt

Iptables Config/Syntax Overview

- What rules do we need to create to allow Internet browsing to flow in both directions?
 - Allow inbound and outbound traffic for:
 - TCP ports 80, 443
 - UDP port 53

Iptables Config/Syntax Overview

- **sudo iptables -A OUTPUT -o eth0 -p tcp --dport 443 -m conntrack --ctstate NEW,ESTABLISHED -j ACCEPT**
- **sudo iptables -A OUTPUT -o eth0 -p tcp --dport 80 -m conntrack --ctstate NEW,ESTABLISHED -j ACCEPT**
- **sudo iptables -A OUTPUT -o eth0 -p udp --dport 53 -m conntrack --ctstate NEW,ESTABLISHED -j ACCEPT**

Iptables Config/Syntax Overview

- **sudo iptables -A INPUT -i eth0 -p tcp --sport 443 -m conntrack --ctstate ESTABLISHED -j ACCEPT**
- **sudo iptables -A INPUT -i eth0 -p tcp --sport 80 -m conntrack --ctstate ESTABLISHED -j ACCEPT**
- **sudo iptables -A INPUT -i eth0 -p udp --sport 53 -m conntrack --ctstate ESTABLISHED -j ACCEPT**

Iptables Config/Syntax Overview

- **sudo iptables -L**

```
admin2@admin2-VirtualBox:~$ sudo iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source                destination
DROP       all  --  anywhere              anywhere
ACCEPT     tcp  --  anywhere              anywhere             tcp spt:https ctsta
state ESTABLISHED
ACCEPT     tcp  --  anywhere              anywhere             tcp spt:http ctsta
state ESTABLISHED
ACCEPT     udp  --  anywhere              anywhere             udp spt:domain cts
state ESTABLISHED

Chain FORWARD (policy ACCEPT)
target     prot opt source                destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source                destination
ACCEPT     tcp  --  anywhere              anywhere             tcp dpt:https ctsta
state NEW,ESTABLISHED
ACCEPT     tcp  --  anywhere              anywhere             tcp dpt:http ctsta
state NEW,ESTABLISHED
ACCEPT     udp  --  anywhere              anywhere             udp dpt:domain cts
state NEW,ESTABLISHED
```

Iptables Config/Syntax Overview

- Can you connect to web.iit.edu now???
- Why not???
- Hit the X in the address bar again

```
admin2@admin2-VirtualBox:~$ sudo iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source                               destination
DROP      all  --  anywhere                               anywhere
ACCEPT     tcp  --  anywhere                               anywhere      tcp spt:https ctst
state ESTABLISHED
ACCEPT     tcp  --  anywhere                               anywhere      tcp spt:http ctsta
state ESTABLISHED
ACCEPT     udp  --  anywhere                               anywhere      udp spt:domain cts
state ESTABLISHED

Chain FORWARD (policy ACCEPT)
target     prot opt source                               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source                               destination
ACCEPT     tcp  --  anywhere                               anywhere      tcp dpt:https ctst
state NEW,ESTABLISHED
ACCEPT     tcp  --  anywhere                               anywhere      tcp dpt:http ctsta
state NEW,ESTABLISHED
ACCEPT     udp  --  anywhere                               anywhere      udp dpt:domain cts
state NEW,ESTABLISHED
```

Iptables Config/Syntax Overview

- Delete the Explicit DROP and add it again at the bottom of the INPUT chain.
 - **sudo iptables -D INPUT 1**
 - **sudo iptables -I INPUT 4 -i eth0 -j DROP**
- Now, try to connect to web.iit.edu as well as yahoo.com
- Does it work???
 - Yes!

Iptables Config/Syntax Overview

- Correct rule set

```
admin2@admin2-VirtualBox:~$ sudo iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source                destination          tcp spt:https ctsta
ate ESTABLISHED
ACCEPT     tcp  --  anywhere              anywhere             tcp spt:http ctsta
te ESTABLISHED
ACCEPT     udp  --  anywhere              anywhere             udp spt:domain cts
tate ESTABLISHED
DROP       all  --  anywhere              anywhere

Chain FORWARD (policy ACCEPT)
target     prot opt source                destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source                destination          tcp dpt:https ctst
ate NEW,ESTABLISHED
ACCEPT     tcp  --  anywhere              anywhere             tcp dpt:http ctsta
te NEW,ESTABLISHED
ACCEPT     udp  --  anywhere              anywhere             udp dpt:domain cts
tate NEW,ESTABLISHED
```

Iptables Config/Syntax Overview

- Default Implicit Policy is set to ACCEPT for chains

```
admin2@admin2-VirtualBox:~$ sudo iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source                destination
ACCEPT     tcp  --  anywhere              anywhere             tcp spt:https ctst
ACCEPT     tcp  --  anywhere              anywhere             tcp spt:http ctsta
ACCEPT     udp  --  anywhere              anywhere             udp spt:domain
DROP       all  --  anywhere              anywhere

Chain FORWARD (policy ACCEPT)
target     prot opt source                destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source                destination
ACCEPT     tcp  --  anywhere              anywhere             tcp dpt:https ctst
ACCEPT     tcp  --  anywhere              anywhere             tcp dpt:http ctsta
ACCEPT     udp  --  anywhere              anywhere             udp dpt:domain
```

Iptables Config/Syntax Overview

- Instead of adding a DROP rule at the end of the chain, you could change the default policy from IMPLICIT ACCEPT to DROP

Iptables Config/Syntax Overview

- **sudo iptables -P INPUT DROP**
- **sudo iptables -L**

Chain INPUT (policy DROP)

```
target      prot opt source                destination
ACCEPT      tcp  --  anywhere              anywhere             tcp spt:https ctst
ate ESTABLISHED
ACCEPT      tcp  --  anywhere              anywhere             tcp spt:http ctsta
te ESTABLISHED
ACCEPT      udp  --  anywhere              anywhere             udp spt:domain cts
tate ESTABLISHED
DROP        all  --  anywhere              anywhere
```

Chain FORWARD (policy ACCEPT)

```
target      prot opt source                destination
```

Iptables Config/Syntax Overview

- Let's change it back:
 - **sudo iptables -P INPUT ACCEPT**
 - **sudo iptables -L**

```
Chain INPUT (policy ACCEPT)
```

```
target      prot opt source                destination
ACCEPT      tcp  --  anywhere              anywhere            tcp spt:https ctsta
state ESTABLISHED
ACCEPT      tcp  --  anywhere              anywhere            tcp spt:http ctsta
state ESTABLISHED
ACCEPT      udp  --  anywhere              anywhere            udp spt:domain cts
state ESTABLISHED
DROP        all  --  anywhere              anywhere
```

```
Chain FORWARD (policy ACCEPT)
```

```
target      prot opt source                destination
```

Iptables Config/Syntax Overview

- Why did we say it might it be a bad idea to change your default chain policies to an implicit DROP???
- Better just to have the default policy as Implicit ACCEPT and add the Explicit DROP rule at the end

Iptables Config/Syntax Overview

- When you reboot your rules are not persistent.
- Save your rules in the Documents folder:
 - **sudo iptables-save > iptables.dump**
- Now let's flush (delete) all of the rules.
 - **sudo iptables -F**
- Now list the rules and you should notice they have disappeared

Iptables Config/Syntax Overview

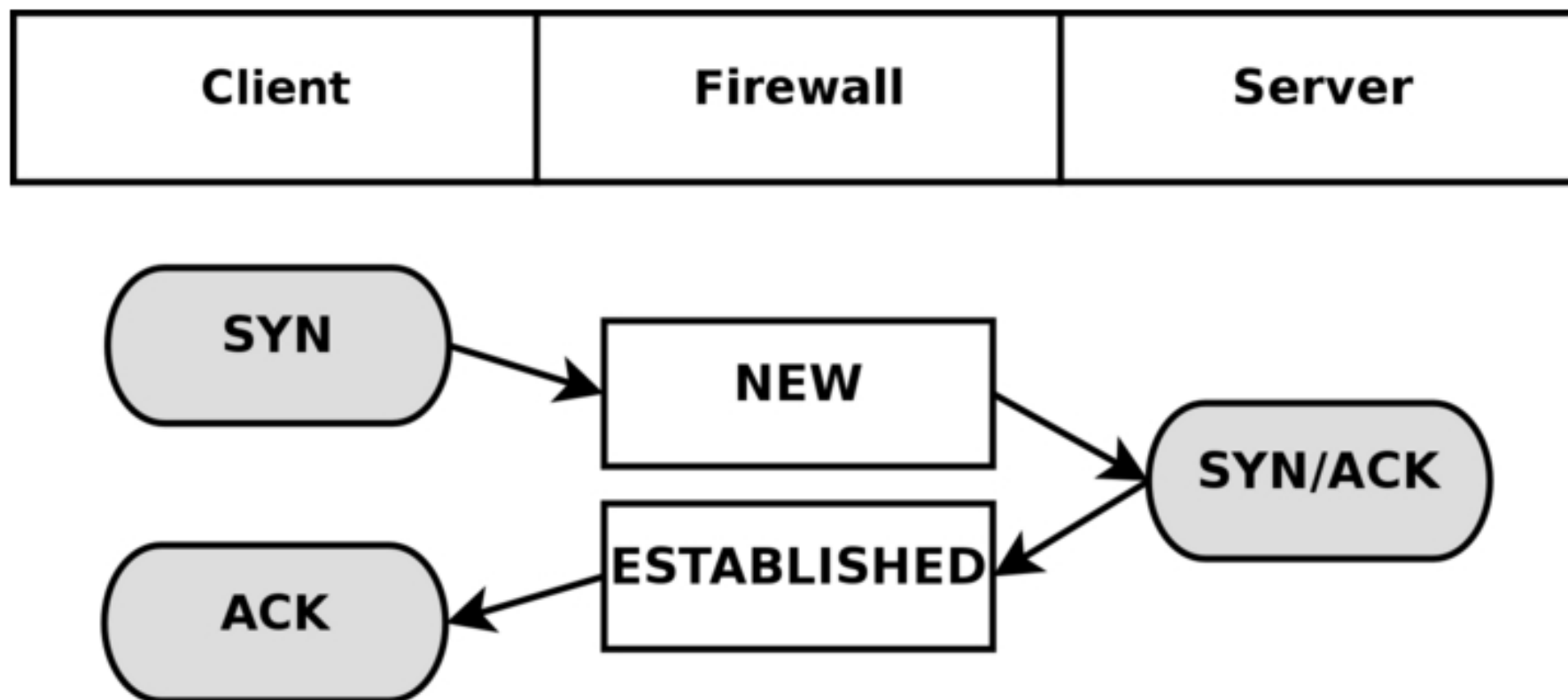
- Now let's reload the rules:
 - **`sudo iptables-restore < iptables.dump`**
- List the rules and you should notice they are all back in place

Conntrack

- Since iptables is stateful, we have been tracking the connection state.
- Our rules so far have allowed:
 - New and established connections outbound
 - Only already established connections inbound

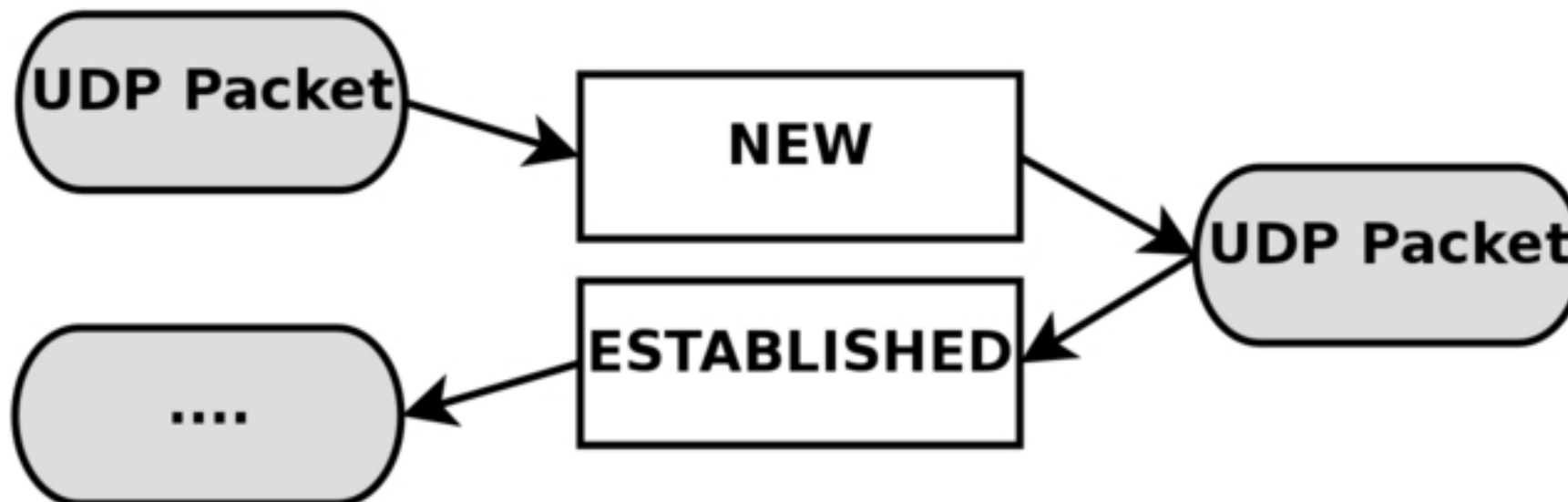
Conntrack

- Here is what conntrack looks at for TCP



Conntrack

- Here is what conntrack looks at for UDP



Conntrack

- Let's view some state information in real time (which will also show you the tcp flags):
- First, let's clear the conntrack cache
 - **sudo conntrack -F**
- Now, start the real-time connection tracking
 - **sudo conntrack -E**
- Connect to web.iit.edu and watch the terminal
 - CTRL-C to stop conntrack after about 10 seconds

Conntrack

```
[NEW] tcp      6 120 SYN_SENT src=10.0.2.15 dst=74.125.228.2 sport=36600 dpo
rt=80 [UNREPLIED] src=74.125.228.2 dst=10.0.2.15 sport=80 dport=36600
[NEW] tcp      6 120 SYN_SENT src=10.0.2.15 dst=74.125.228.2 sport=36601 dpo
rt=80 [UNREPLIED] src=74.125.228.2 dst=10.0.2.15 sport=80 dport=36601
[UPDATE] tcp    6 60 SYN_RECV src=10.0.2.15 dst=74.125.228.2 sport=36600 dpor
t=80 src=74.125.228.2 dst=10.0.2.15 sport=80 dport=36600
[UPDATE] tcp    6 432000 ESTABLISHED src=10.0.2.15 dst=74.125.228.2 sport=366
00 dport=80 src=74.125.228.2 dst=10.0.2.15 sport=80 dport=36600 [ASSURED]
```

```
[UPDATE] udp    17 30 src=127.0.0.1 dst=127.0.1.1 sport=58525 dport=53 src=12
7.0.1.1 dst=127.0.0.1 sport=53 dport=58525
[UPDATE] udp    17 180 src=127.0.0.1 dst=127.0.1.1 sport=58525 dport=53 src=1
27.0.1.1 dst=127.0.0.1 sport=53 dport=58525 [ASSURED]
```

- “ASSURED” means the connection will not be erased or overwritten by other traffic in the queue

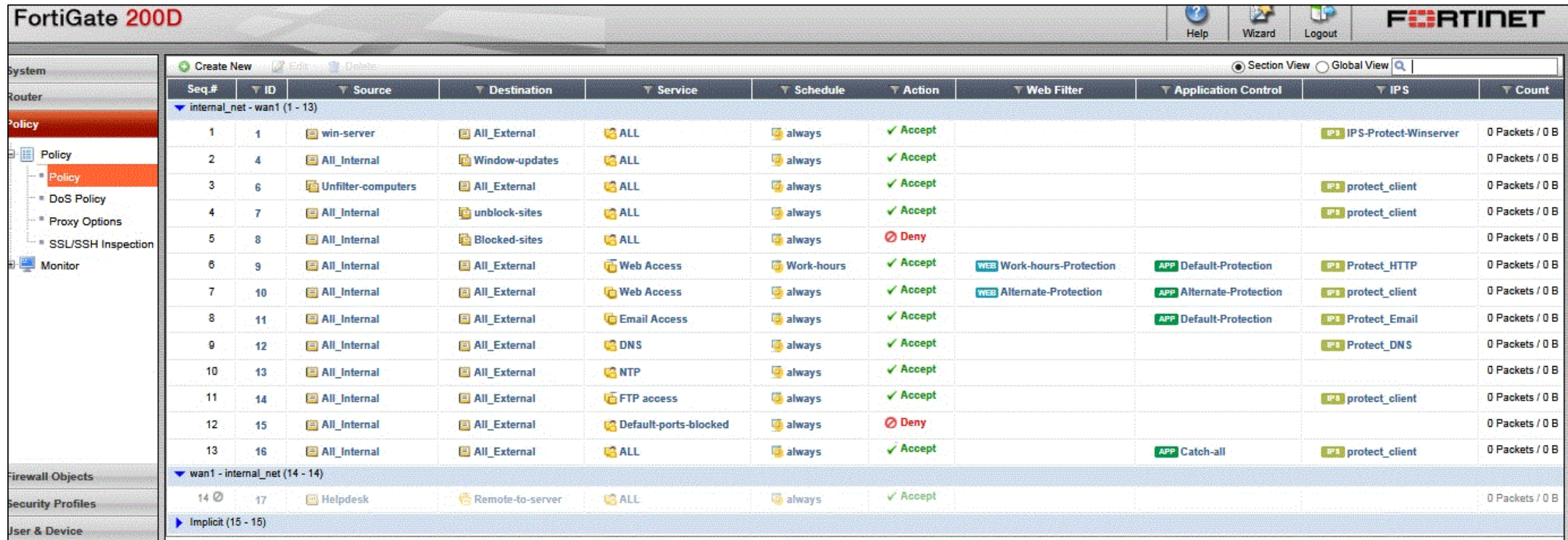
Conntrack

- You can also list the current connection information
- **sudo conntrack -L**

```
udp      17 166 src=127.0.0.1 dst=127.0.1.1 sport=36118 dport=53 src=127.0.1.1 d
st=127.0.0.1 sport=53 dport=36118 [ASSURED] mark=0 use=1
udp      17 168 src=127.0.0.1 dst=127.0.1.1 sport=44068 dport=53 src=127.0.1.1 d
st=127.0.0.1 sport=53 dport=44068 [ASSURED] mark=0 use=1
tcp      6 431992 ESTABLISHED src=10.0.2.15 dst=98.139.183.24 sport=45610 dport=
443 src=98.139.183.24 dst=10.0.2.15 sport=443 dport=45610 [ASSURED] mark=0 use=2
udp      17 18 src=10.0.2.15 dst=172.29.0.11 sport=46911 dport=53 src=172.29.0.1
1 dst=10.0.2.15 sport=53 dport=46911 mark=0 use=1
udp      17 17 src=127.0.0.1 dst=127.0.1.1 sport=34327 dport=53 src=127.0.1.1 ds
t=127.0.0.1 sport=53 dport=34327 mark=0 use=1
udp      17 168 src=127.0.0.1 dst=127.0.1.1 sport=51250 dport=53 src=127.0.1.1 d
st=127.0.0.1 sport=53 dport=51250 [ASSURED] mark=0 use=1
```

Iptables Exploratory

- Was anyone able to figure out all of these rules on their own?



Examples of Commercial Network Firewalls

The screenshot displays a web-based configuration interface for a commercial network firewall. The interface includes a menu bar (File, View, Tools, Wizards, Window, Help) and a toolbar with icons for Home, Configuration, Monitoring, Save, Refresh, Back, Forward, and Help. A "Look For:" search bar is located in the top right corner.

The left sidebar contains a "Device List" with a search field and a "Firewall" section with a tree view of configuration categories: Access Rules, NAT Rules, Service Policy Rules, AAA Rules, Filter Rules, Public Servers, URL Filtering Servers, Threat Detection, Objects, Unified Communications, and Advanced.

The main content area is titled "Configuration > Firewall > Access Rules" and features a toolbar with icons for Add, Edit, Delete, and other actions. Below the toolbar is a table of access rules.

#	Enabled	Source	Destination	Service	Action	Hits	Logging	Time	Description
inside (4 incoming rules)									
1	<input checked="" type="checkbox"/>	inside-network/24	any	tcp-udp	Per...				
2	<input checked="" type="checkbox"/>	inside-network/24	any	icmp	Per...				
3	<input checked="" type="checkbox"/>	inside-network/24	any	ip	Per...				
4		any	any	ip	Deny				Implicit rule
outside (3 incoming rules)									
1	<input checked="" type="checkbox"/>	any	outside	http https	Per...				
2	<input checked="" type="checkbox"/>	any	outside	icmp	Per...				
3		any	any	ip	Deny				Implicit rule

Checkpoint

- Difference between host based and network based firewall?
- What are the two types of firewalls?
- Difference between stateless and stateful packet filters?
- What is a DMZ used for?
- Why do proxy servers substantially increase security?

Checkpoint

- What is the difference between an Application Level Proxy and a SOCKS Circuit Level Proxy?
- Why can't attackers directly access an internal host behind a NAT box by its IP address
- What do the following chains in the iptables filter table do?
 - INPUT
 - OUTPUT
 - FORWARD

Checkpoint

- What is conntrack used for?
- If iptables shows “Chain INPUT (policy ACCEPT)”
 - Is that implicit or explicit?
 - How would we get rid of all other traffic that we didn't explicitly create rules for?
- Why might an implicit DROP be a bad idea?
- Are iptables rule changes persistent on reboot?
 - How could we make them persistent on reboot?

Homework

- Complete Homework11 located on Blackboard under “Homework Assignments”
 - Homework 11 is due before midnight on Sunday, April 3rd.
- Don't forget that your homework from the Crypto lecture is due next Sunday, April 27th and includes the next step of your individual project.
 - If you have questions, you must email me between now and Friday. I will be unavailable this coming weekend.