CS 155 Spring 2014

Network Security Protocols and Defensive Mechanisms

John Mitchell

Plan for today

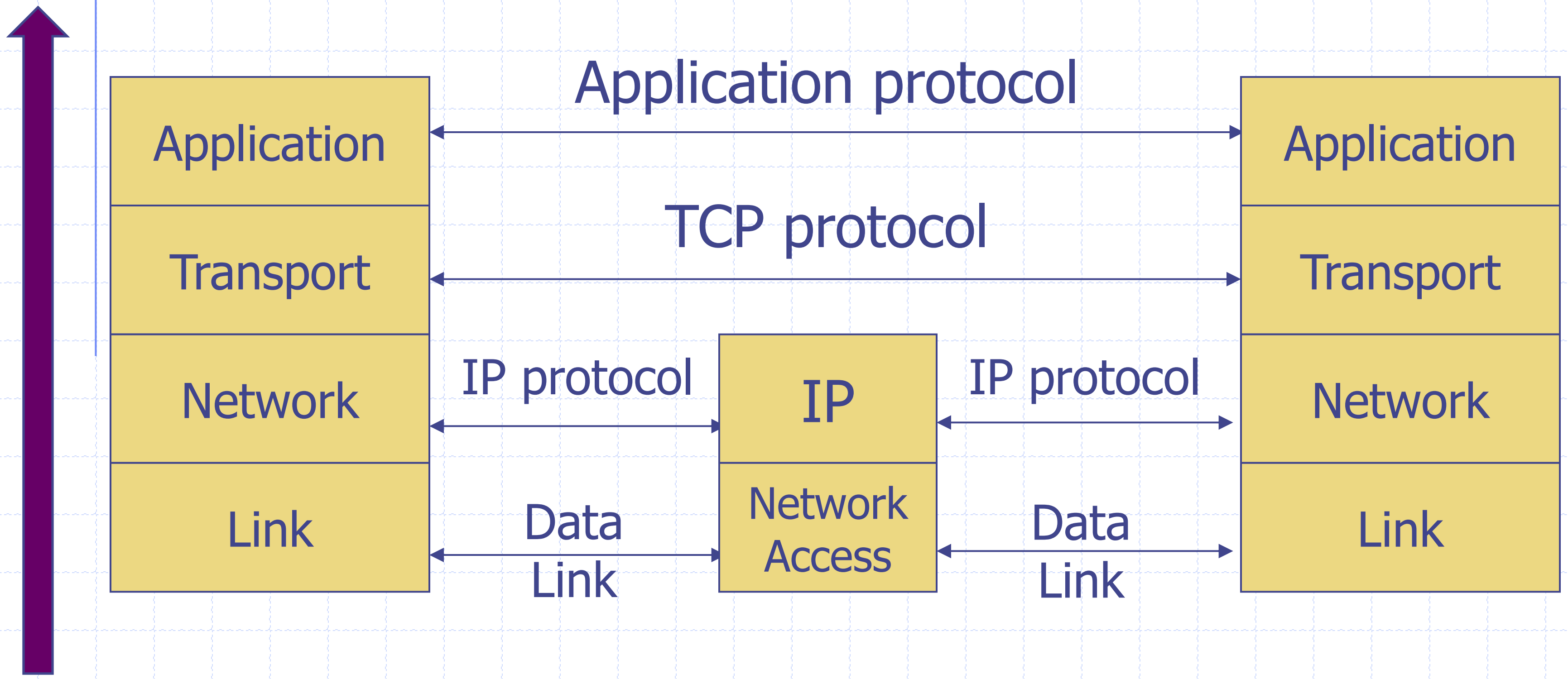
- Network protocol security
  - Wireless access– 802.11i/WPA2
  - IPSEC
  - BGP instability and S-BGP
  - DNS rebinding and DNSSEC
- Standard network defenses
  - Firewall
    - Packet filter (stateless, stateful), Application layer proxies
  - Intrusion detection
    - Anomaly and misuse detection

© art.com

Last lecture

- Basic network protocols
  - IP, TCP, UDP, BGP, DNS
- Problems with them
  - TCP/IP
    - No SRC authentication: can’t tell where packet is from
    - Packet sniffing
    - Connection spoofing, sequence numbers
  - BGP: advertise bad routes or close good ones
  - DNS: cache poisoning, rebinding
    - Web security mechanisms rely on DNS

Network Protocol Stack



TCP/IP connectivity

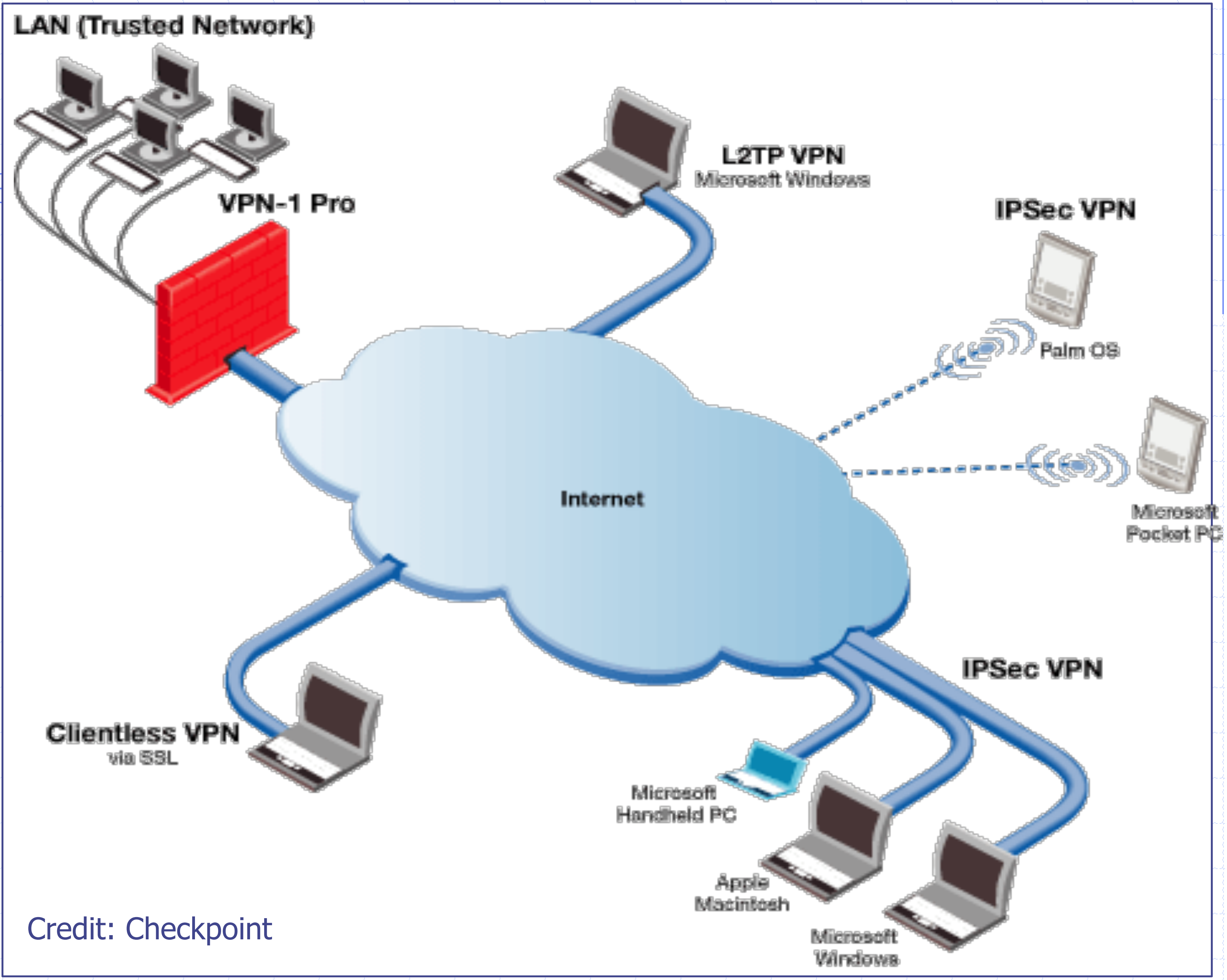
Basic Layer 2-3 Security Problems

- Network packets pass by untrusted hosts
  - Eavesdropping, packet sniffing
  - Especially easy when attacker controls a machine close to victim
- TCP state can be easy to guess
  - Enables spoofing and session hijacking

Virtual Private Network (VPN)

- Three different modes of use:
  - Remote access client connections
  - LAN-to-LAN internetworking
  - Controlled access within an intranet
- Several different protocols
  - PPTP – Point-to-point tunneling protocol
  - L2TP – Layer-2 tunneling protocol

    } Data layer
  - IPsec (Layer-3: network layer)

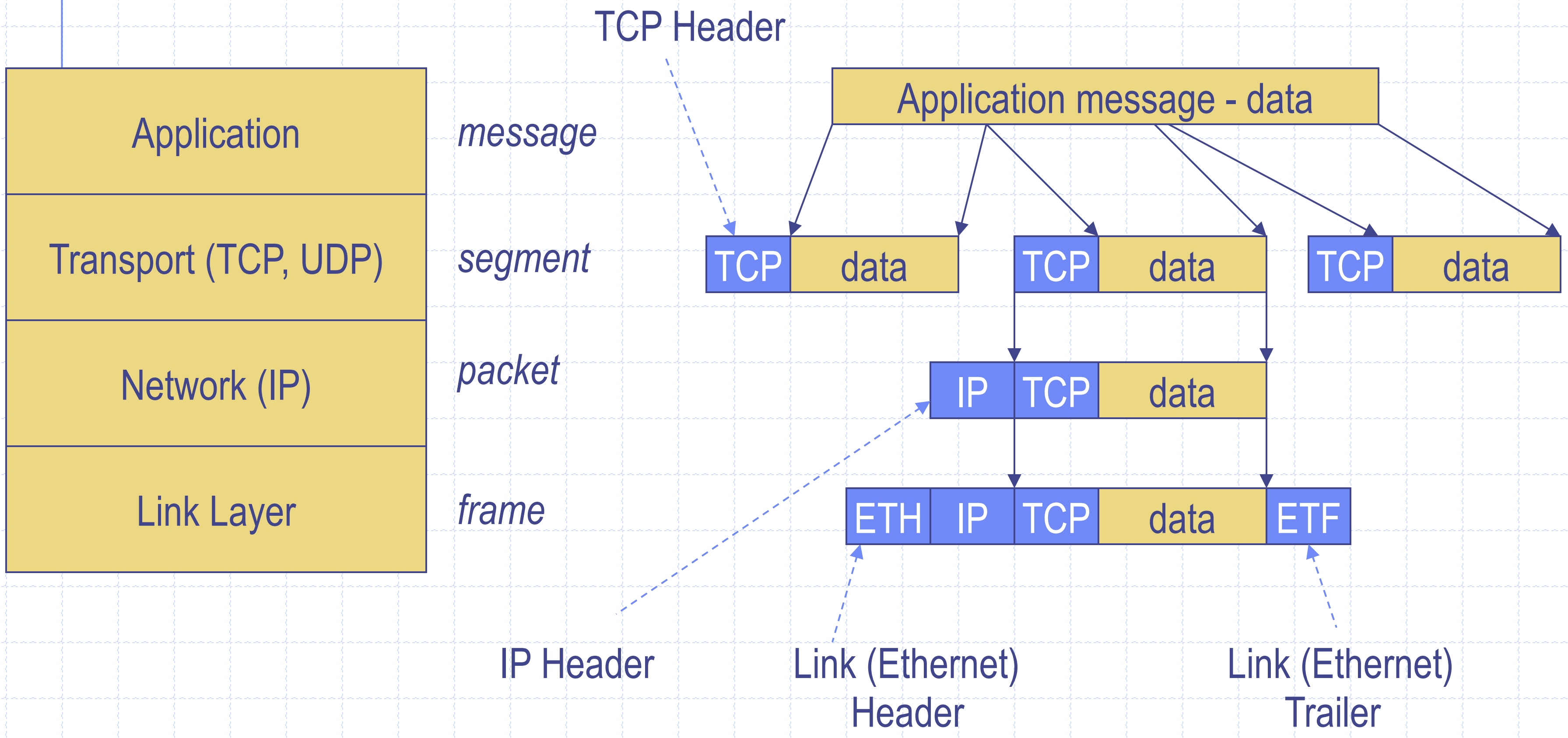


Credit: Checkpoint

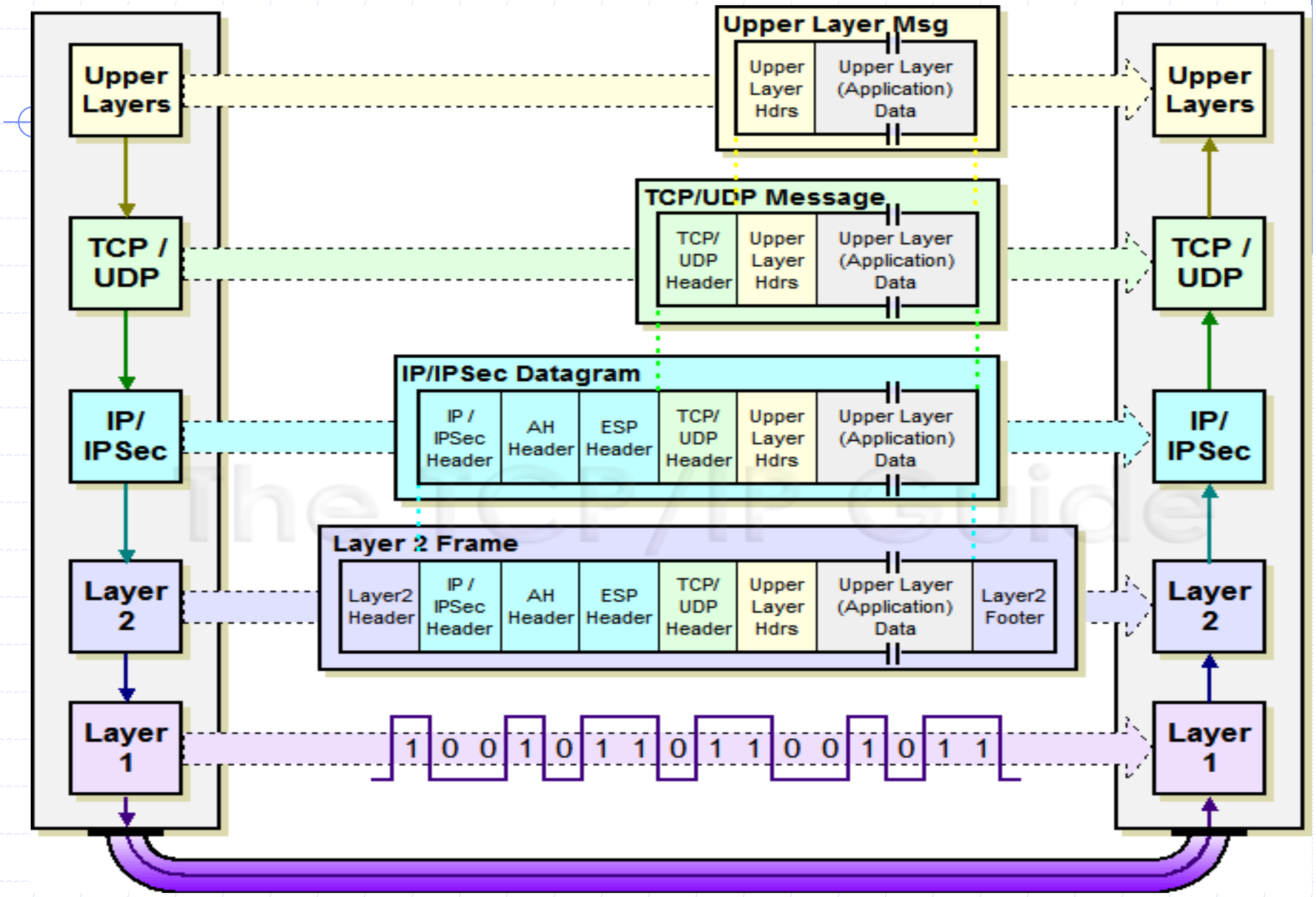
IPSEC

- Security extensions for IPv4 and IPv6
- IP Authentication Header (AH)
  - Authentication and integrity of payload and header
- IP Encapsulating Security Protocol (ESP)
  - Confidentiality of payload
- ESP with optional ICV (integrity check value)
  - Confidentiality, authentication and integrity of payload

Recall packet formats and layers

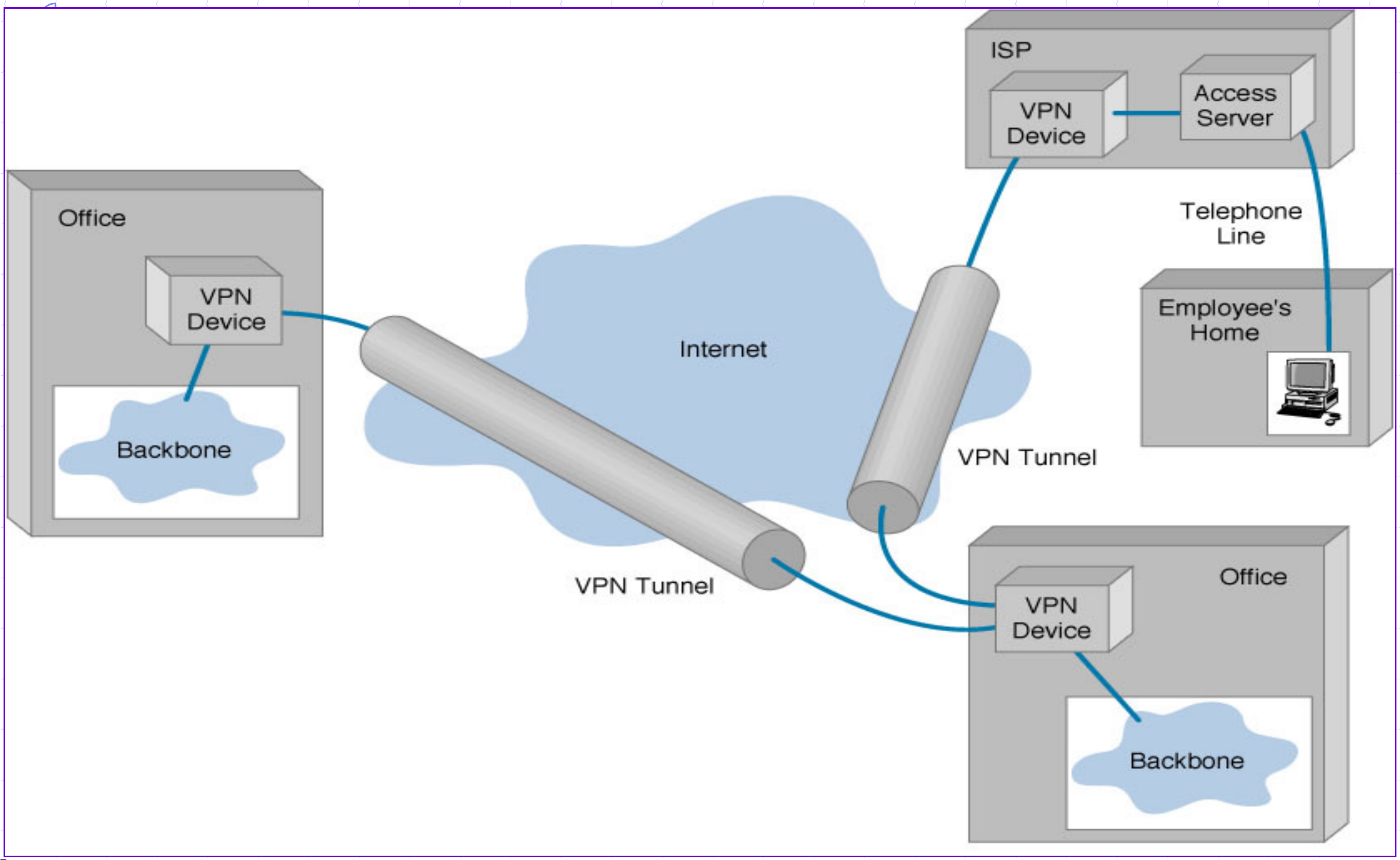


IPSec Transport Mode: IPSEC instead of IP header

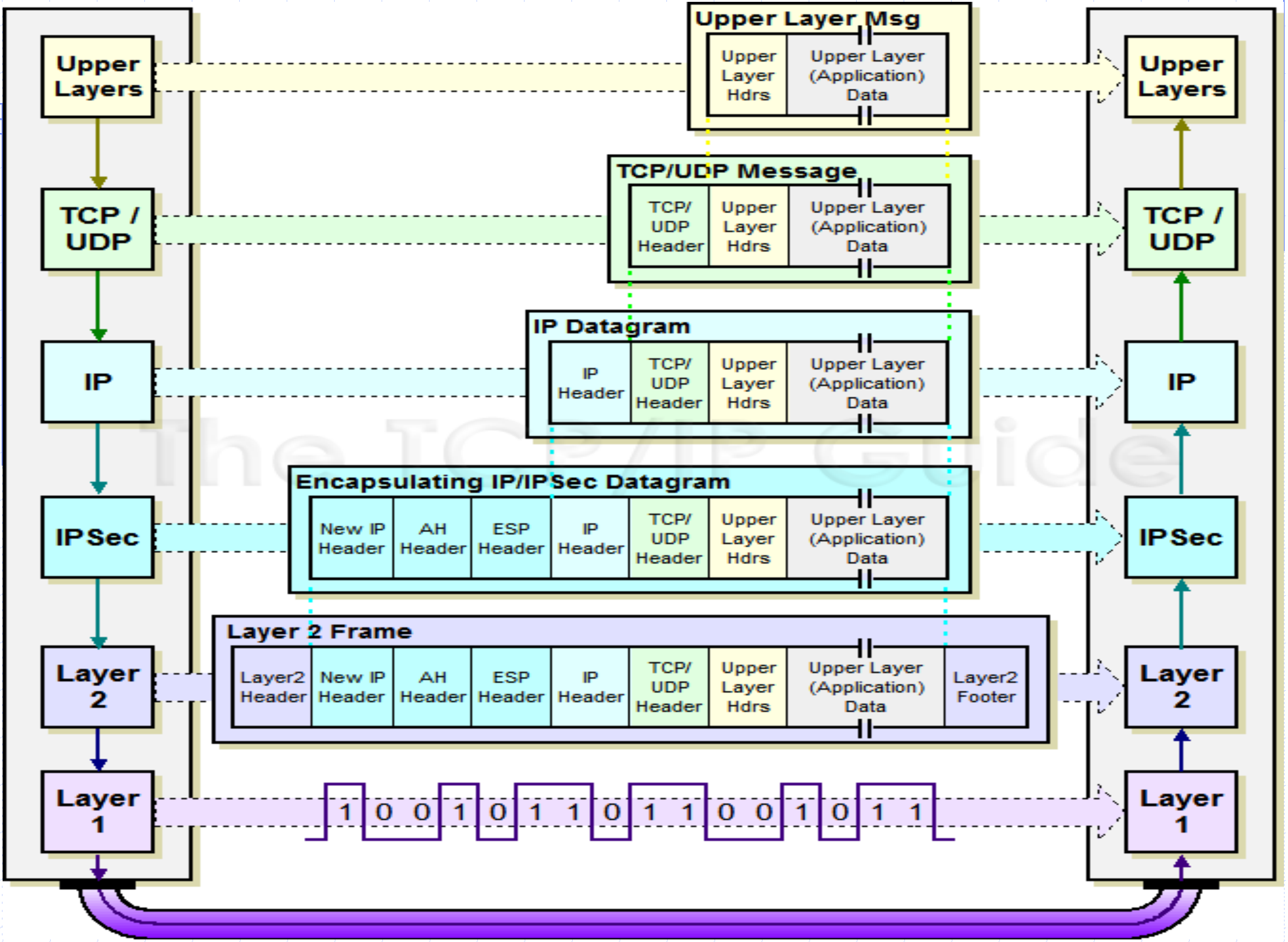


http://www.tcpipguide.com/free/t_IPSecModesTransportandTunnel.htm

IPSEC Tunnel Mode



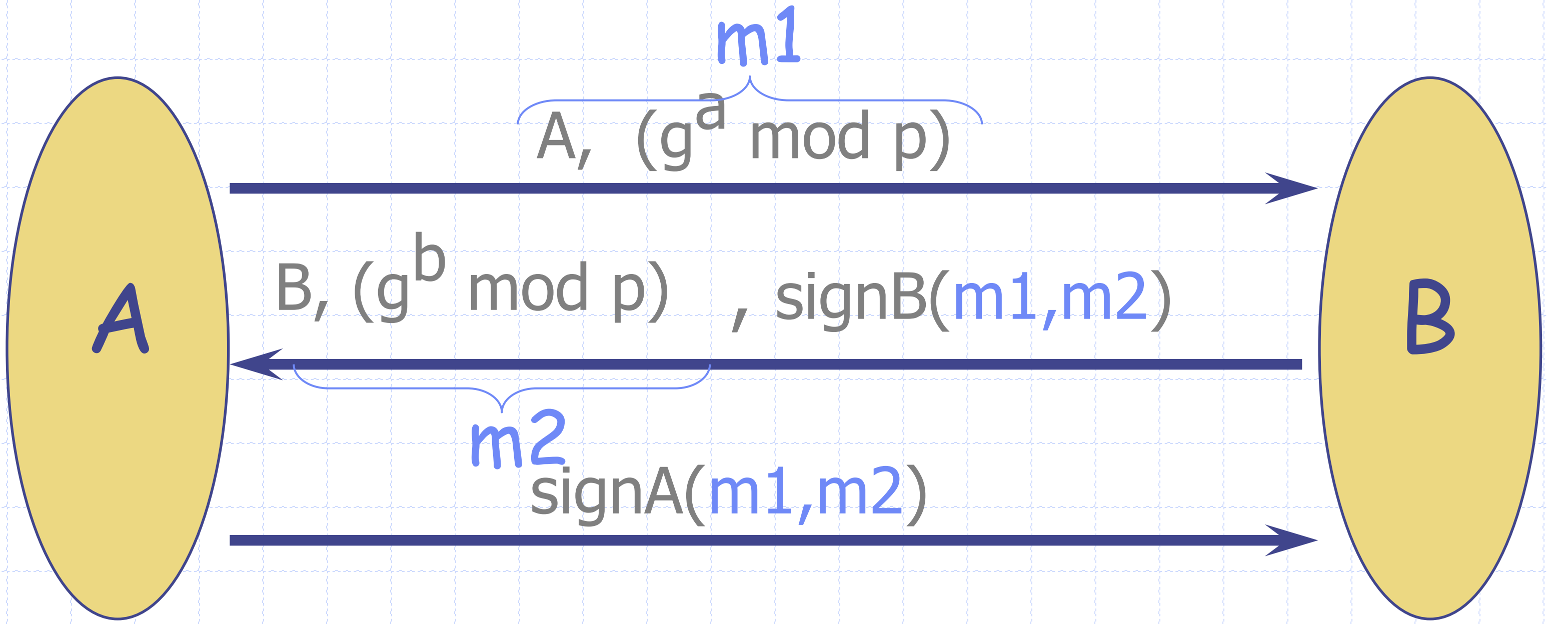
IPSec Tunnel Mode: IPSEC header + IP header



Diffie-Hellman (DH) Algorithm

- Key exchange algorithm based on modulo arithmetic to securely exchange keys without sharing mutual keys
  - Both agree on a large modulus (m) and a common base (x)
    - Each side picks random number s, r
    - Each side computes **x^s mod m, x^r mod m** and transmits this value
    - Each side, raise the received value to its secrete number
      - **x^{rs} mod m, x^{sr} mod m**
    - Both have the same secrete number

IKE subprotocol from IPSEC

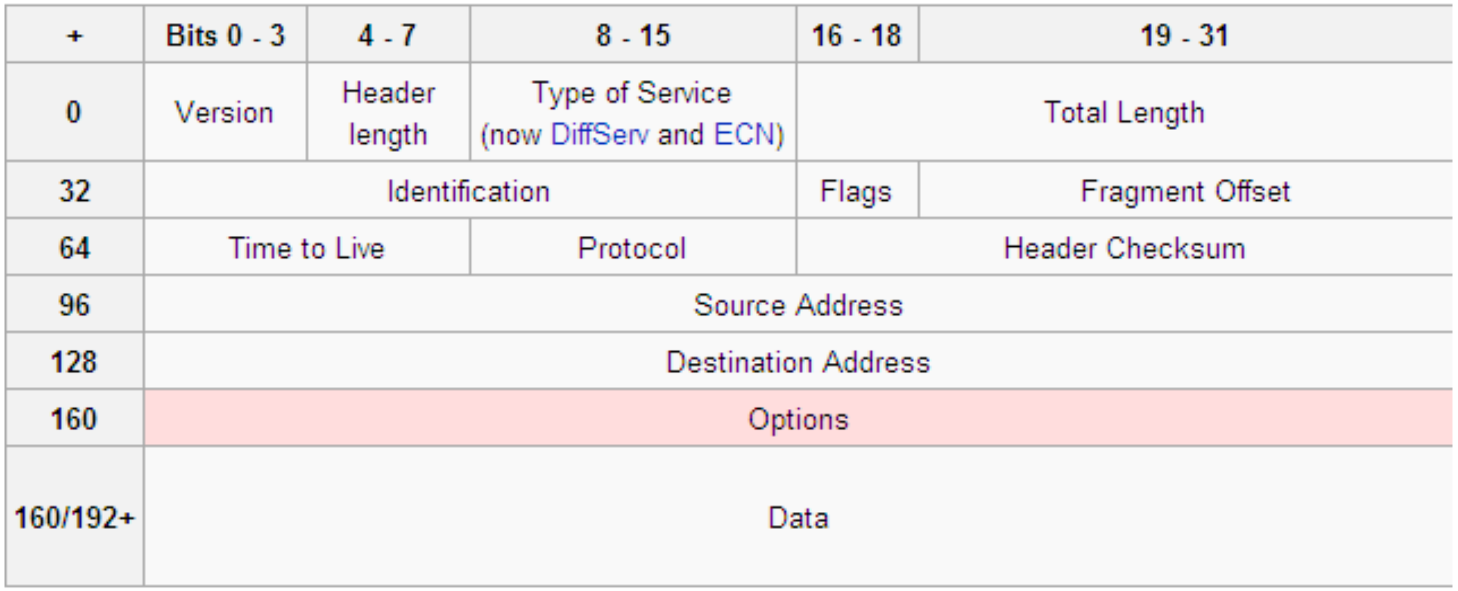

A
B
$\overbrace{A,\ (g^a \bmod p)}^{m1}$
$\underbrace{B,\ (g^b \bmod p)}_{m2}$, signB(m1,m2)
signA(m1,m2)

Result: A and B share secret $g^{ab} \bmod p$

Filtering network traffic

(starting at IP, transport layer …)

IP Header

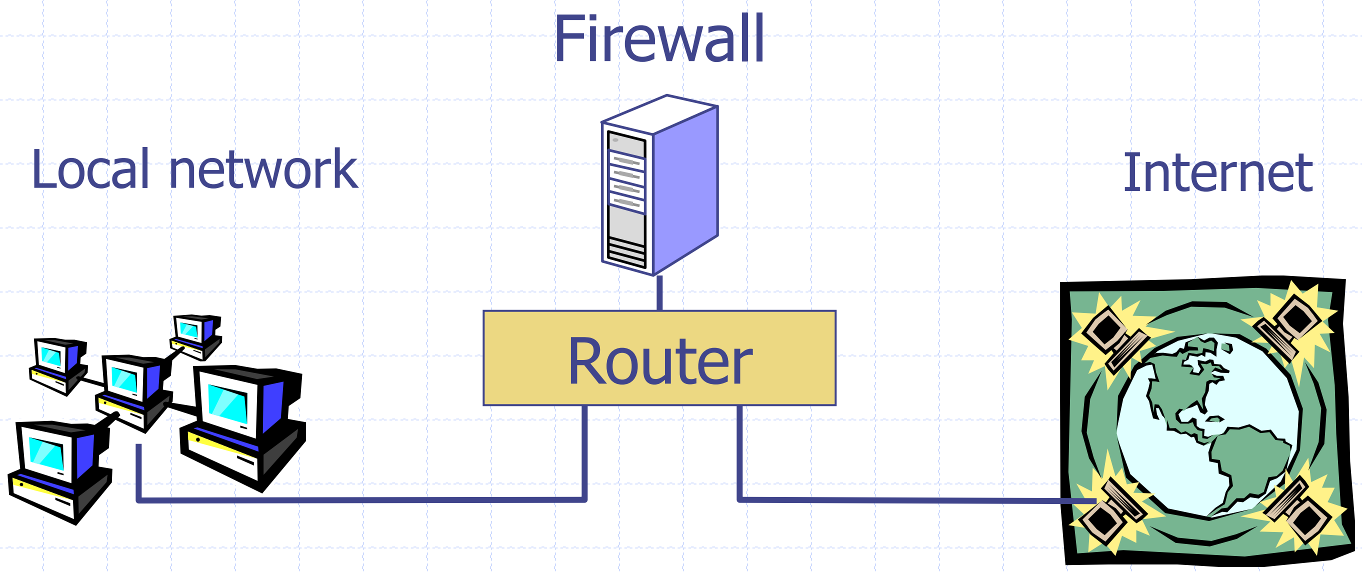


+	Bits 0 - 3	4 - 7	8 - 15	16 - 18	19 - 31
0	Version	Header length	Type of Service (now DiffServ and ECN)	Total Length	
32	Identification			Flags	Fragment Offset
64	Time to Live		Protocol	Header Checksum	
96	Source Address				
128	Destination Address				
160	Options				
160/192+	Data				

Source [http://en.wikipedia.org/wiki/IPv4_header#Header]

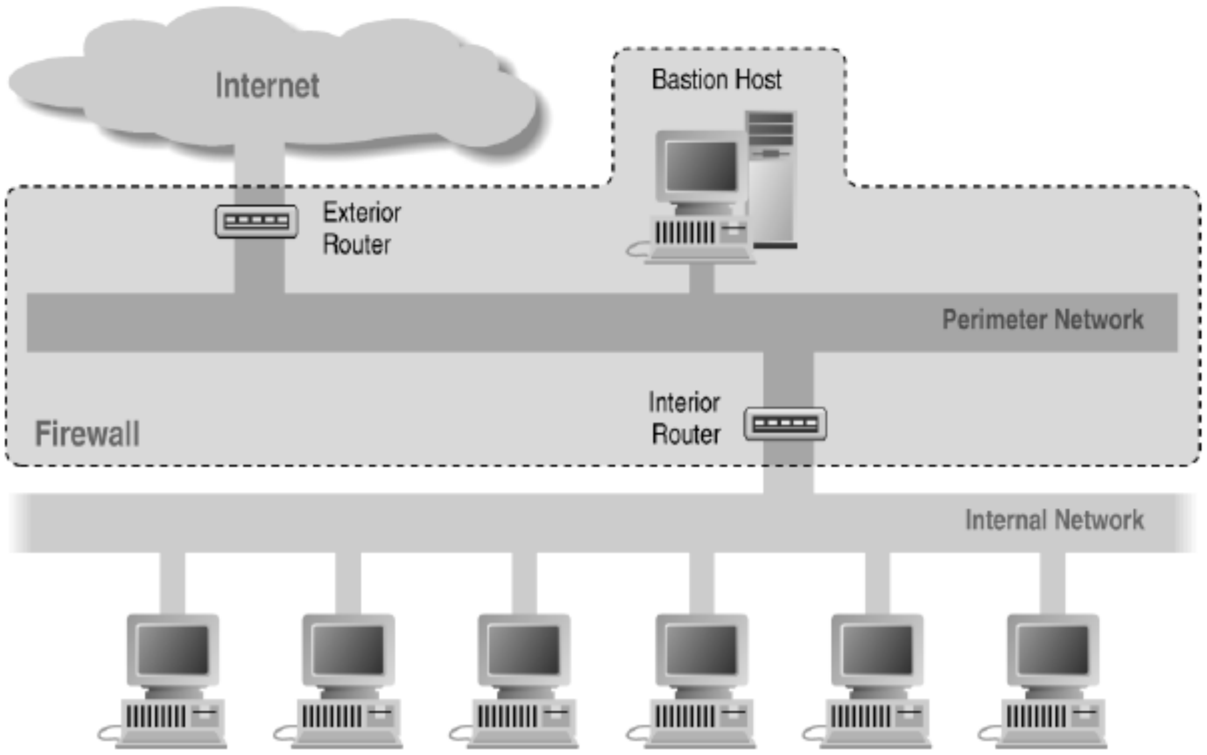
Basic Firewall Concept

- Separate local area net from internet

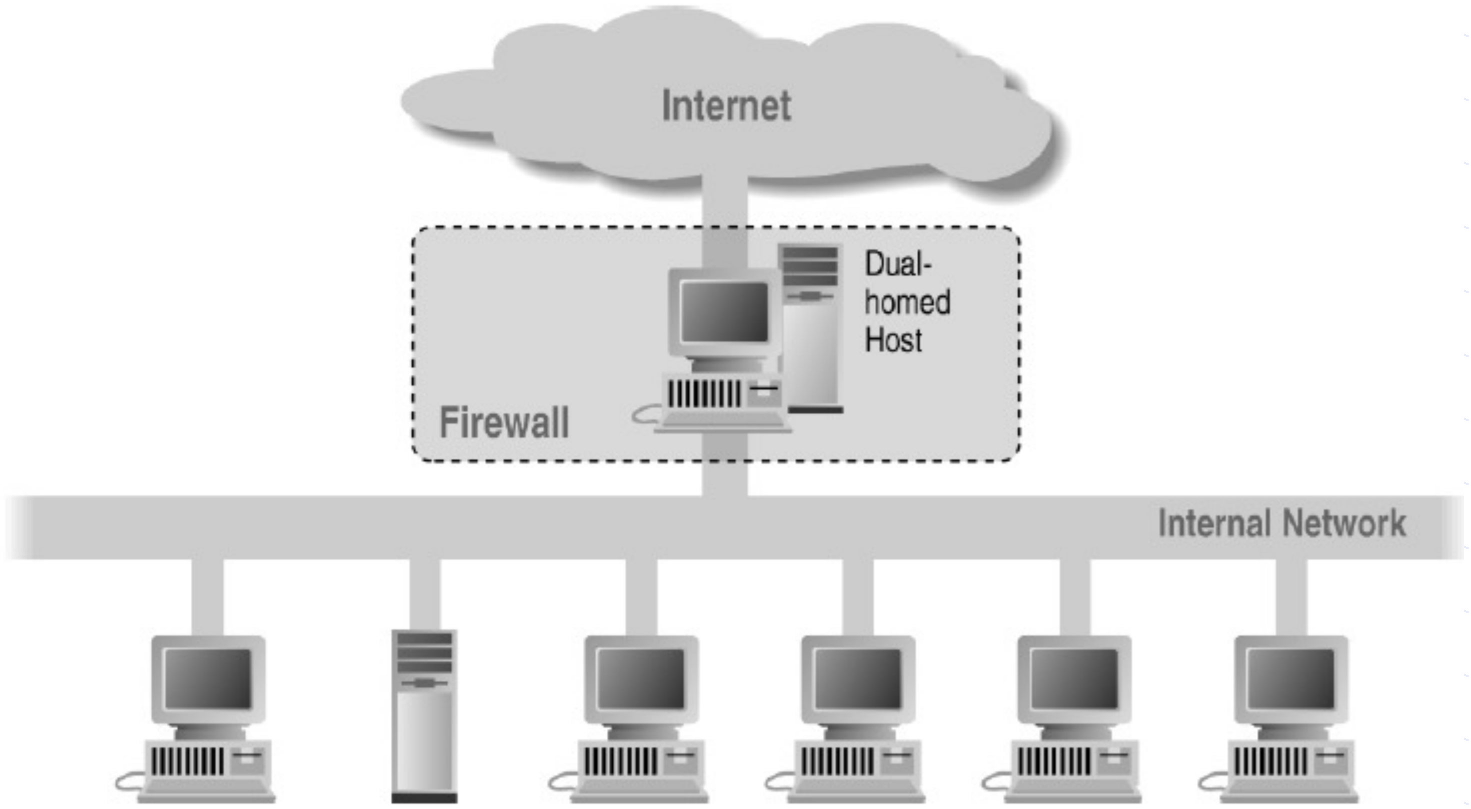


All packets between LAN and internet routed through firewall

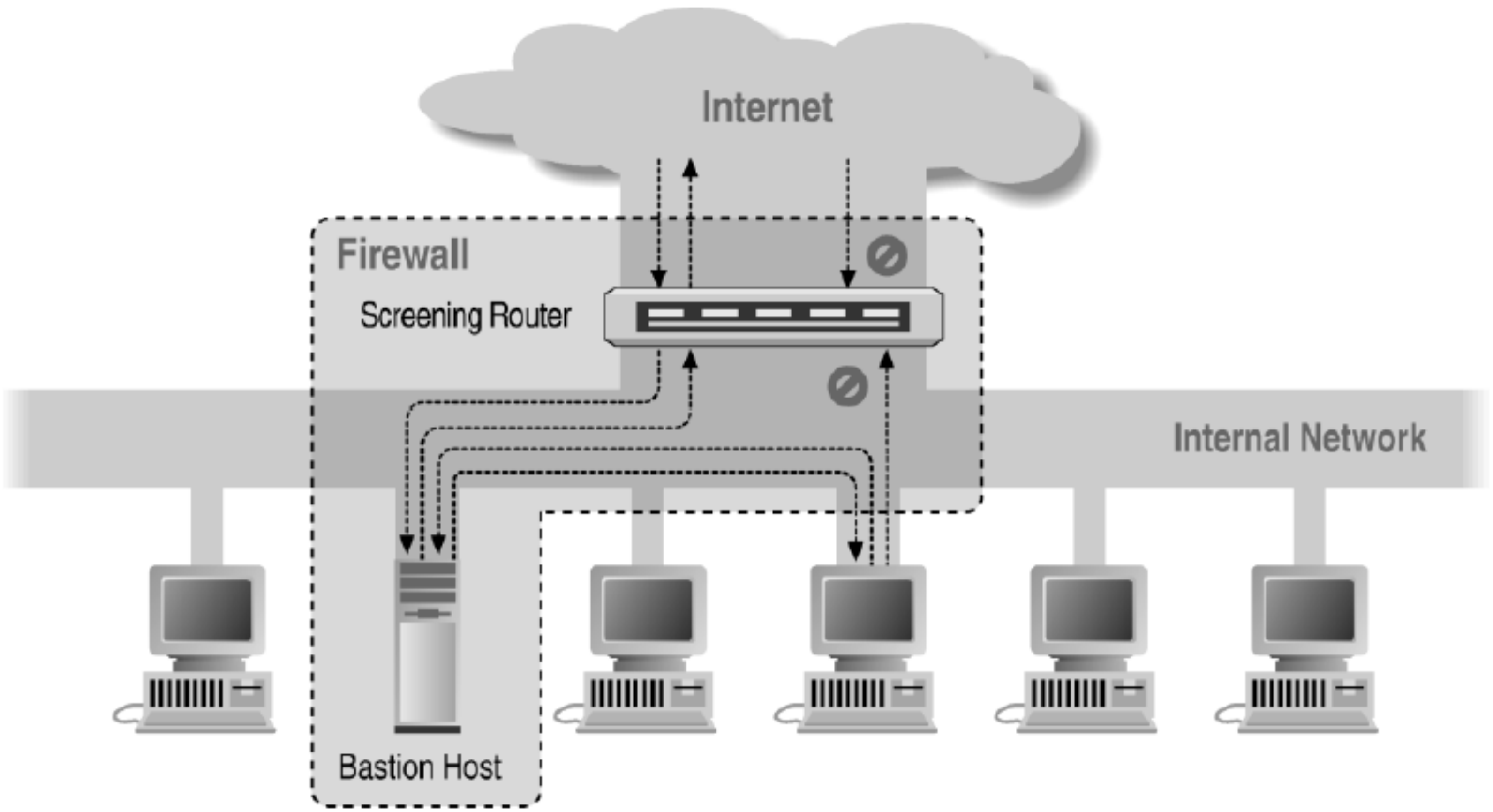
Screened Subnet Using Two Routers



Alternate 1: Dual-Homed Host



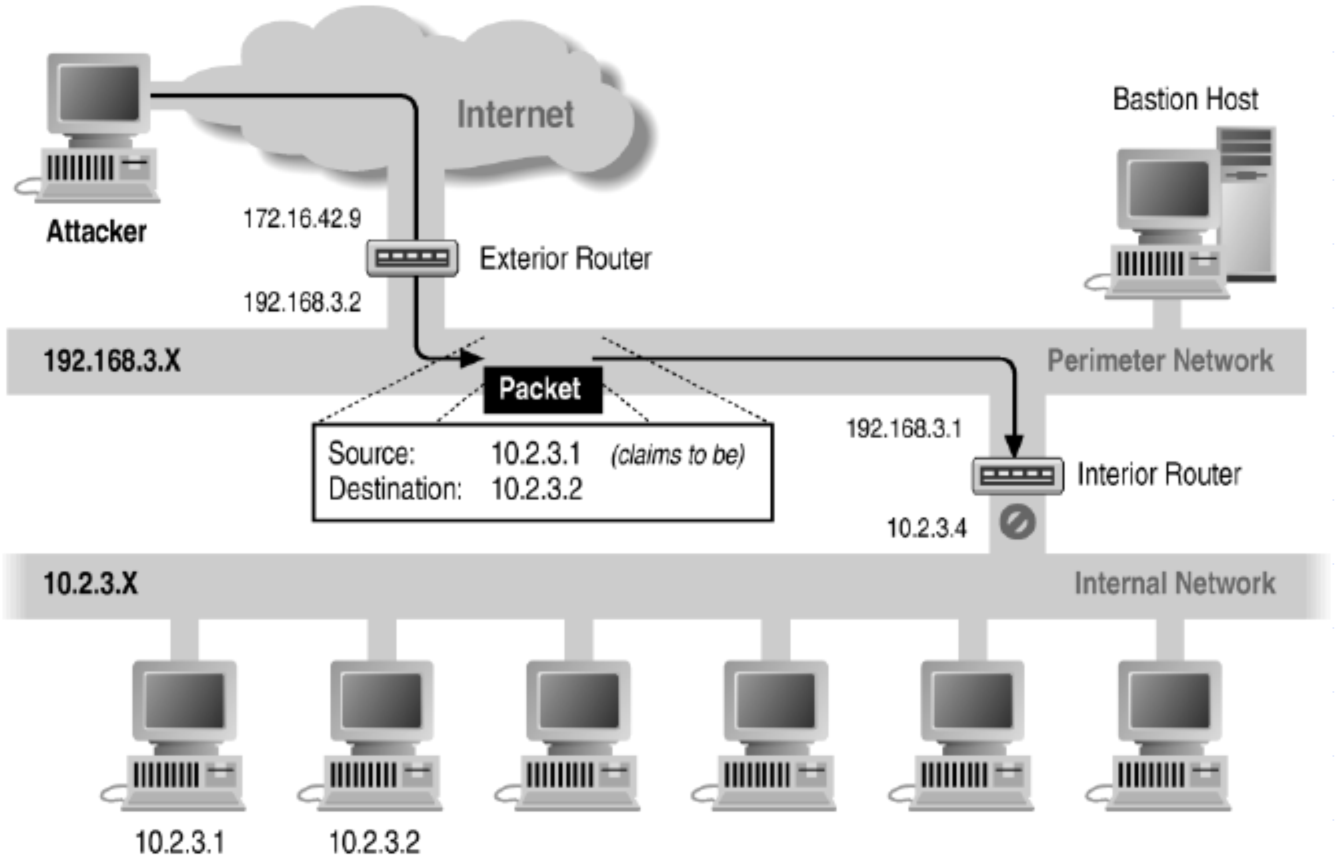
Alternate 2: Screened Host



Basic Packet Filtering

- Uses transport-layer information only
  - IP Source Address, Destination Address
  - Protocol (TCP, UDP, ICMP, etc)
  - TCP or UDP source & destination ports
  - TCP Flags (SYN, ACK, FIN, RST, PSH, etc)
  - ICMP message type
- Examples
  - DNS uses port 53
    - Block incoming port 53 packets except known trusted servers
- Issues
  - Stateful filtering
  - Encapsulation: address translation, other complications
  - Fragmentation

Source/Destination Address Forgery



Address spoofing

- Sender can put any source address in packets he sends:
    - Can be used to send unwelcome return traffic to the spoofed address
    - Can be used to bypass filters to get unwelcome traffic to the destination
- Reverse Path verification can be used by routers to broadly catch some spoofers

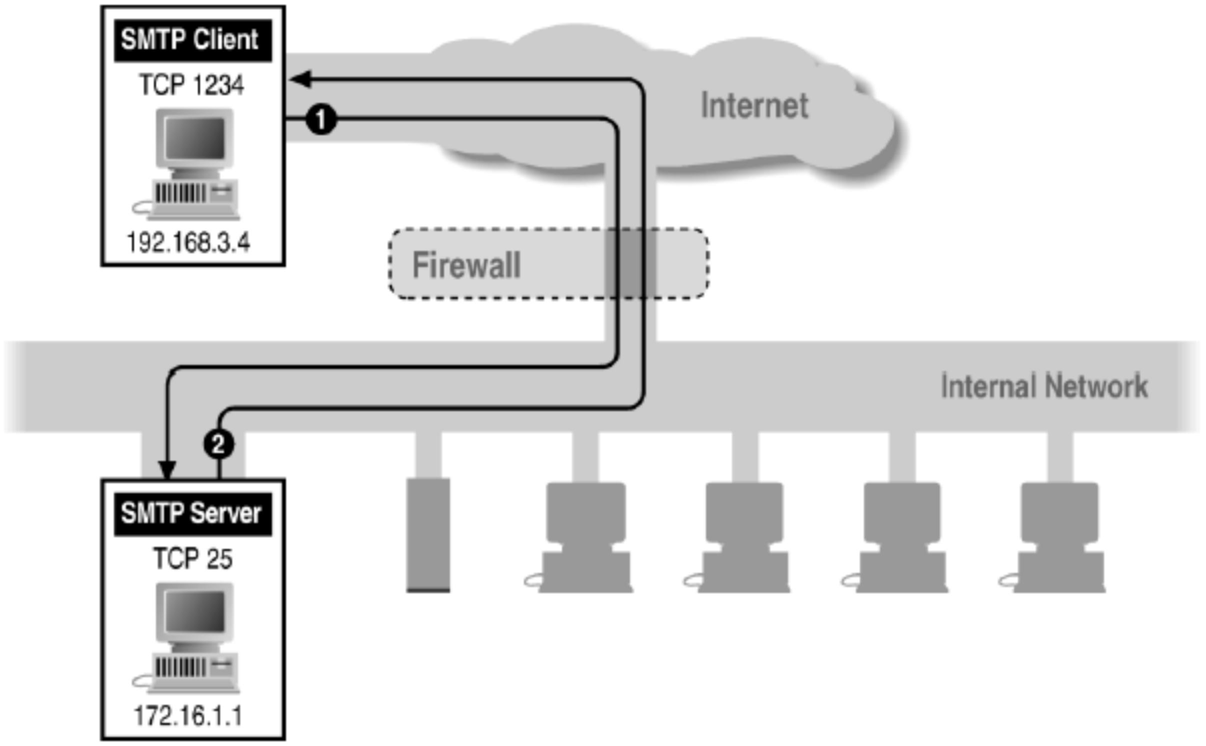
Defending Against IP Spoofing

- Ingress filtering
  - Forbid inbound broadcasts from the Internet into your networks
  - Forbid inbound packets from non-routable networks
- Egress filtering
  - Prevent stations in networks you control from spoofing IPs from other networks by dropping their outbound packets
    - Make your network a less attractive and useful target for attackers that want to launch other attacks
    - Be a good Internet citizen (reputation is important)
  - Drop outbound broadcasts
- Reference
  - RFC 2267 - "Network Ingress Filtering: Defeating Denial of Service Attacks which Employ IP Source Address Spoofing".

More about networking: port numbering

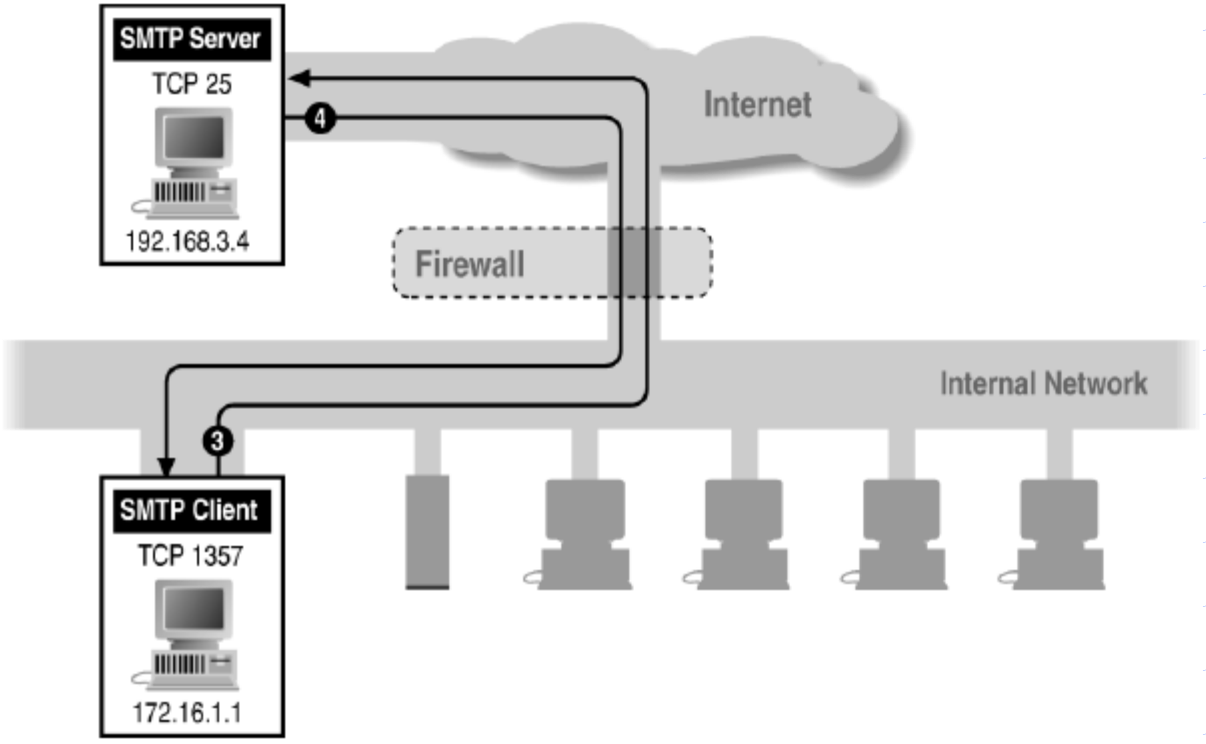
- TCP connection
  - Server port uses number less than 1024
  - Client port uses number between 1024 and 16383
- Permanent assignment
  - Ports <1024 assigned permanently
    - 20,21 for FTP 23 for Telnet
    - 25 for server SMTP 80 for HTTP
- Variable use
  - Ports >1024 must be available for client to make connection
  - Limitation for stateless packet filtering
    - If client wants port 2048, firewall must allow incoming traffic
  - Better: stateful filtering knows outgoing requests
    - Only allow incoming traffic on high port to a machine that has initiated an outgoing request on low port

Filtering Example: Inbound SMTP



Can block external request to internal server based on port number

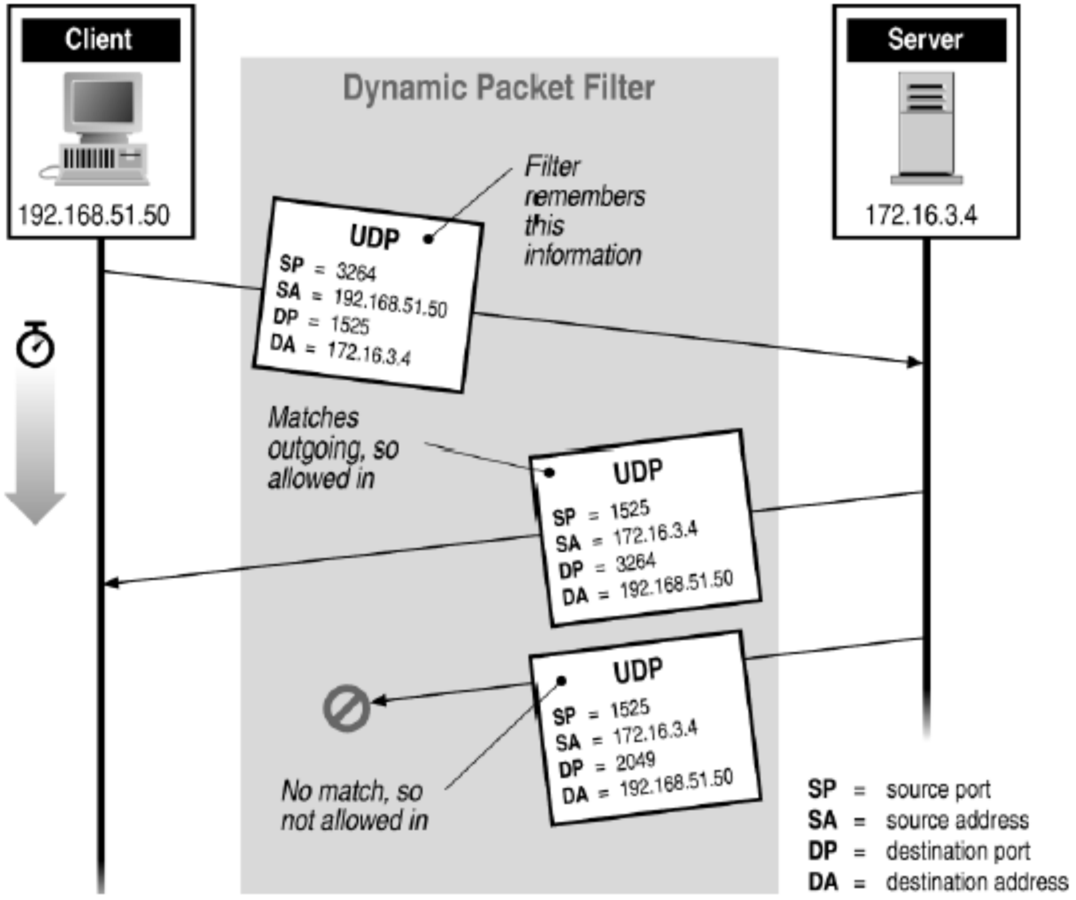
Filtering Example: Outbound SMTP



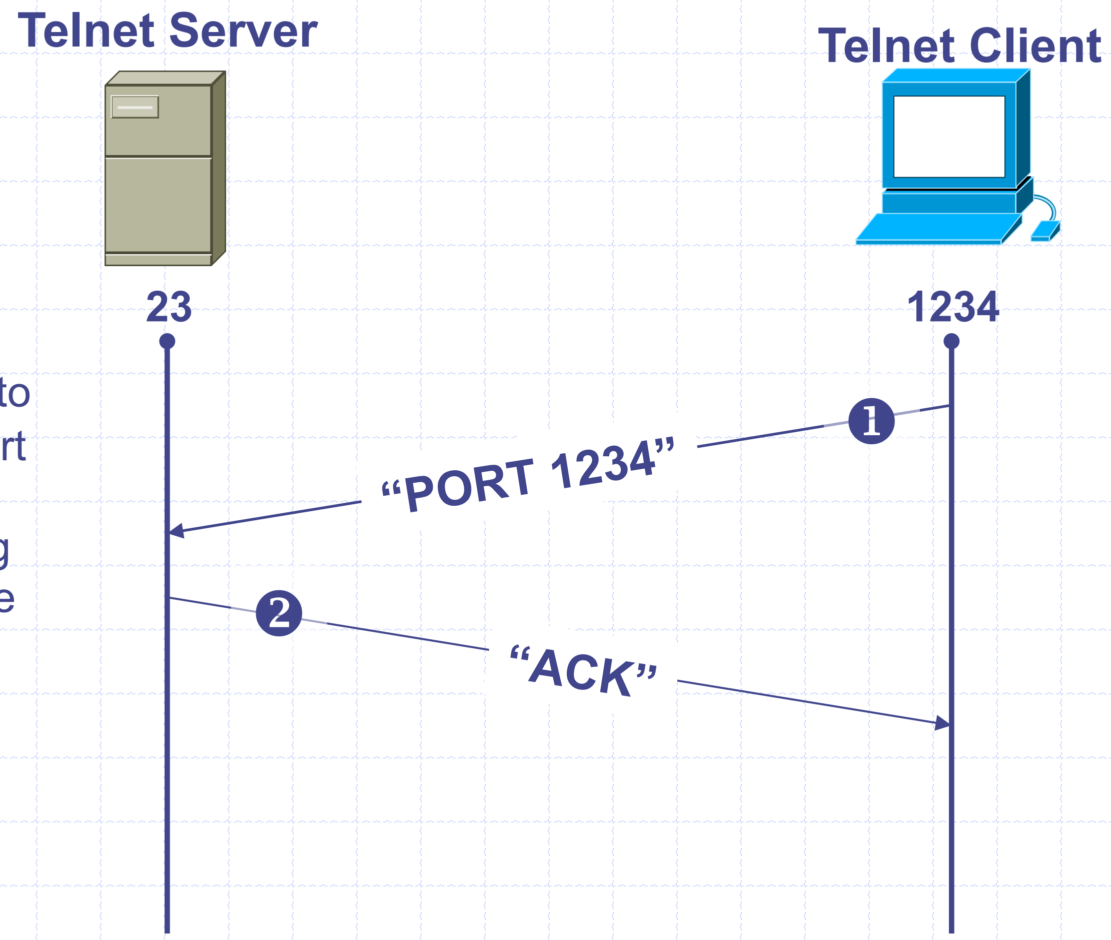
Known low port out, arbitrary high port in

If firewall blocks incoming port 1357 traffic then connection fails

Stateful or Dynamic Packet Filtering



Telnet

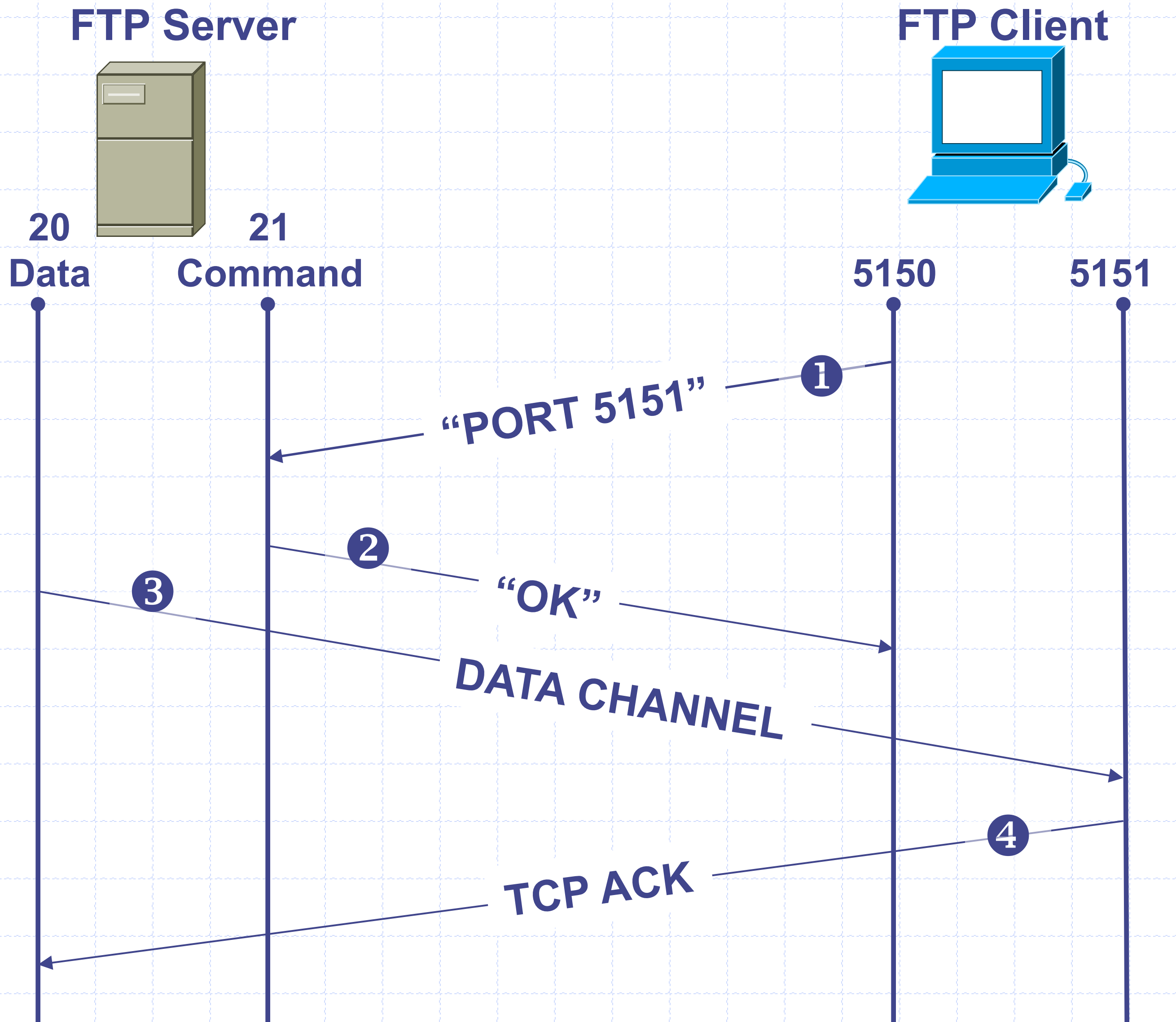


❶ Client opens channel to server; tells server its port number. The ACK bit is not set while establishing the connection but will be set on the remaining packets

❷ Server acknowledges

Stateful filtering can use this pattern to identify legitimate sessions

FTP



❶ Client opens command channel to server; tells server second port number

❷ Server acknowledges

❸ Server opens data channel to client's second port

❹ Client acknowledges

Fragmentation

- May need to fragment an IP packet if one data link along the way cannot handle the packet size
  - Perhaps path is a mix of different types of communication media
  - Perhaps unexpected encapsulation makes the packet larger than the source expected
  - Hosts try to understand Maximum Transmission Unit (MTU) to avoid the need for fragmentation (which causes a performance hit)
  - MTU on Ethernet 1500bytes
- Any device along the way can fragment
  - ***Identification*** field identifies all elements of the same fragment
  - Fragmentation stored in the ***MF*** (more fragments) and fragment offset fields
  - Devices can reassemble too
  - But generally the destination does the reassembly

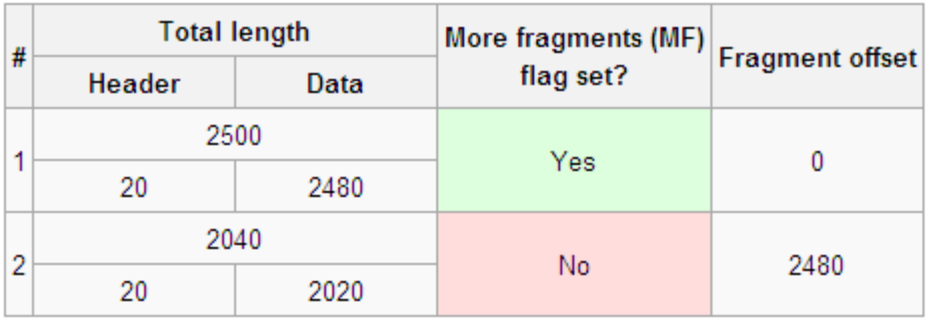
IP Fragmentation Rules

In order for a fragmented packet to be successfully reassembled at the destination each fragment must obey the following rules:

- Must share a common fragment identification number. Also known as fragment Id.
- Each fragment must say what its place or offset is in the original unfragmented packet.
- Each fragment must tell the length of the data carried in the fragment.
- Finally the fragment must know whether more fragments follow this one.

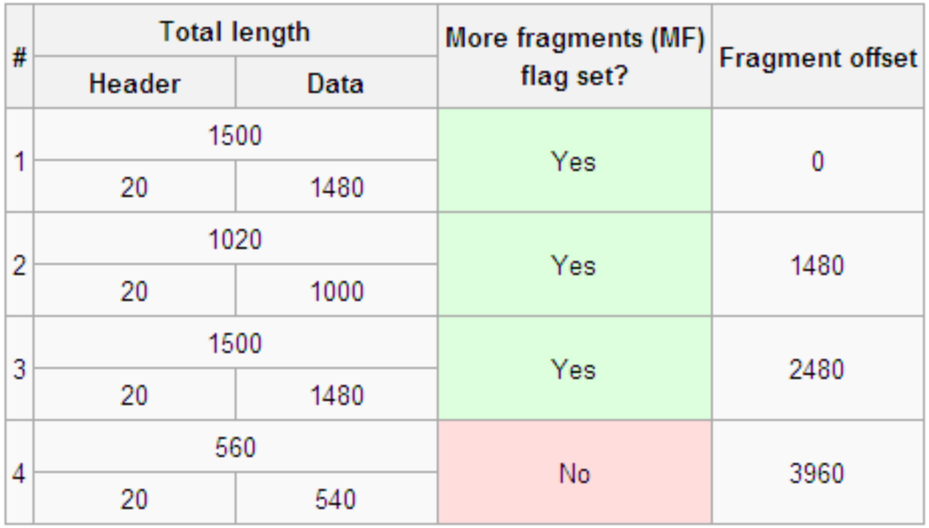
Example of Fragmentation

- For example, if a 4,500 byte data payload is inserted into an IP packet with no options (thus total length is 4,520 bytes) and is transmitted over a link with an MTU of 2,500 bytes then it will be broken up into two fragments:



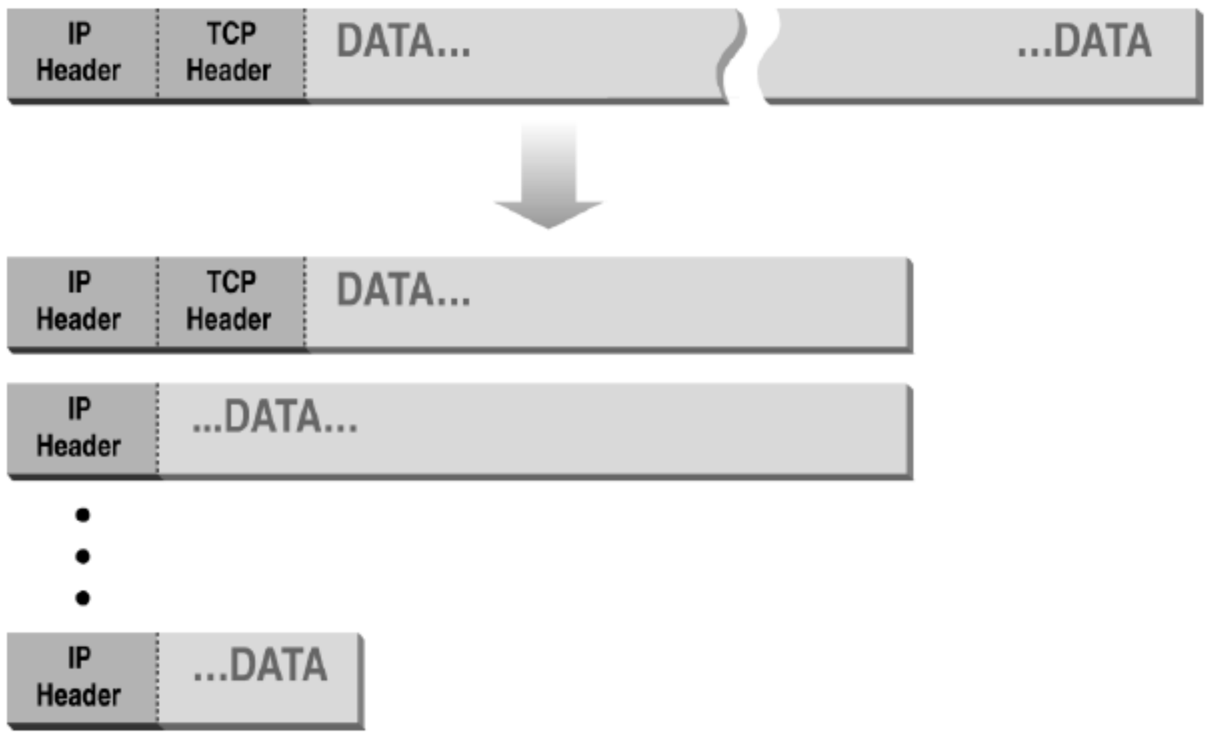
#	Total length		More fragments (MF) flag set?	Fragment offset
	Header	Data		
1	2500		Yes	0
	20	2480		
2	2040		No	2480
	20	2020		

- Now, let's say the MTU drops to 1,500 bytes.



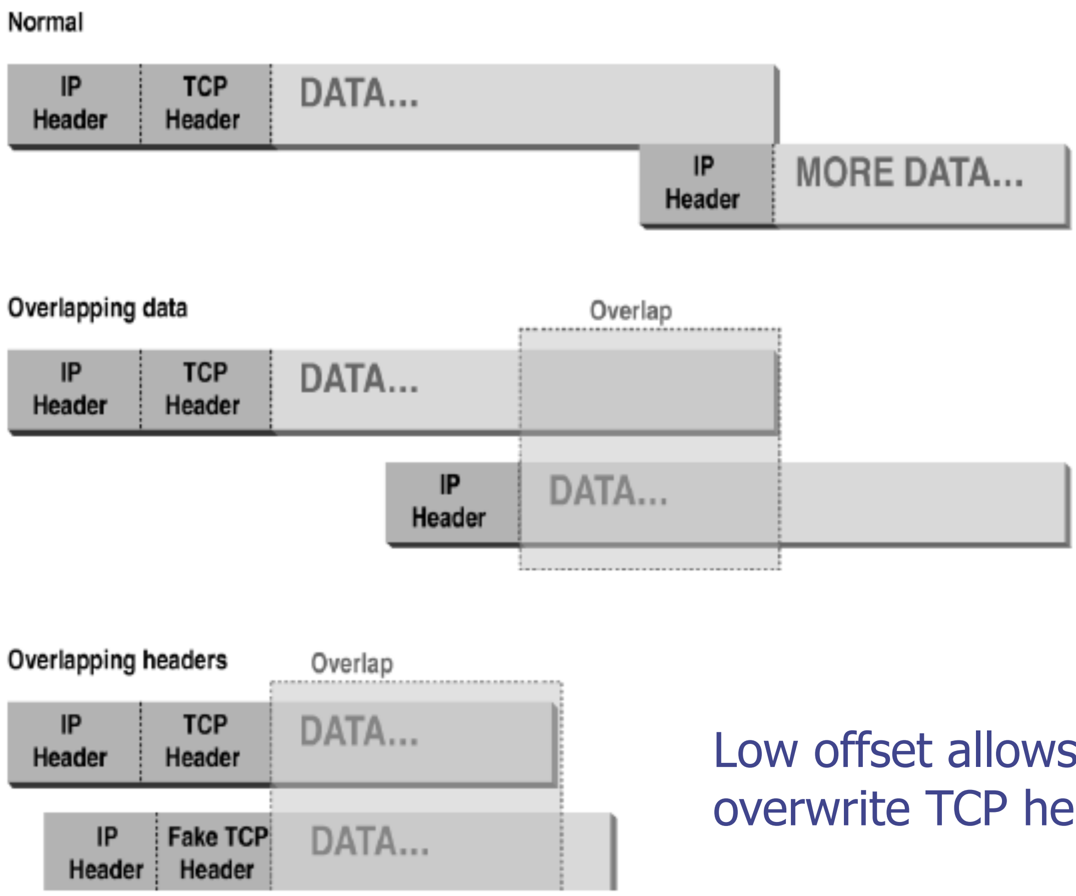
#	Total length		More fragments (MF) flag set?	Fragment offset
	Header	Data		
1	1500		Yes	0
	20	1480		
2	1020		Yes	1480
	20	1000		
3	1500		Yes	2480
	20	1480		
4	560		No	3960
	20	540		

Normal IP Fragmentation



Flags and offset inside IP header indicate packet fragmentation

Abnormal Fragmentation



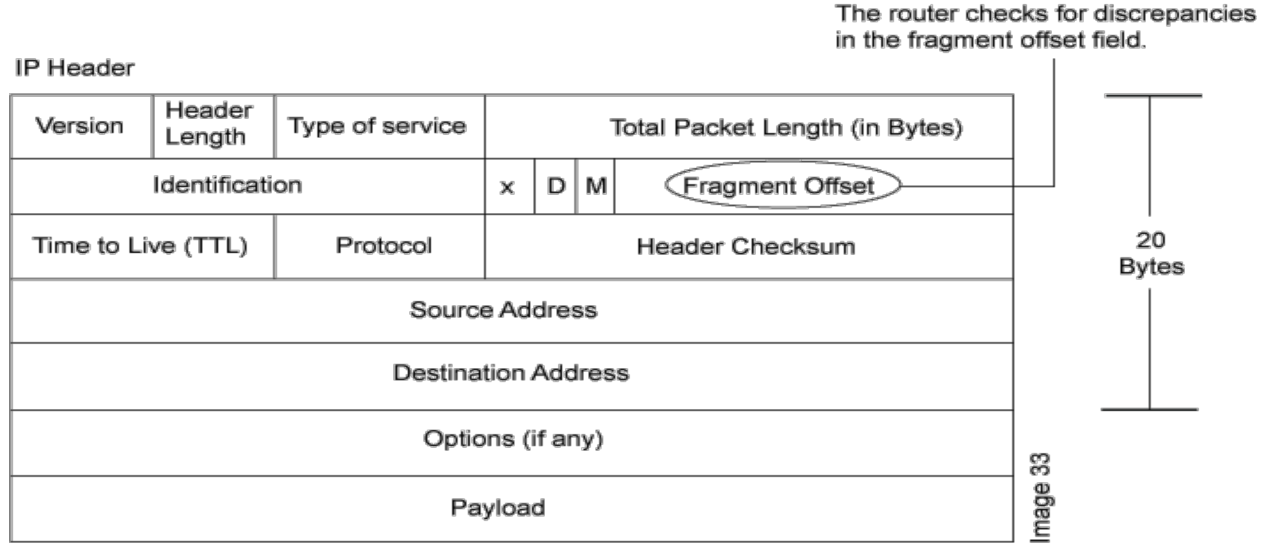
Low offset allows second packet to overwrite TCP header at receiving host

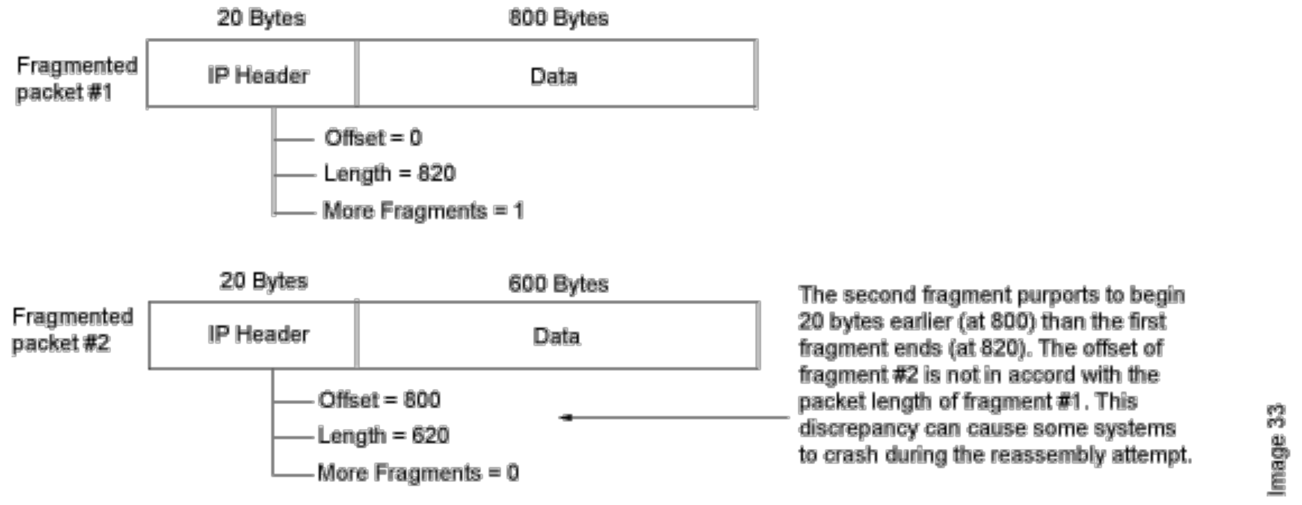
Fragmentation Flaws

- Split packet to fool simple firewall and IDS
  - Intermediate content observers must do reassembly
- Overlapping fragments
  - Can be used to trick IDS by hiding, e.g. a "get /etc/password" request
  - Different clients reassemble overlapping fragments differently
  - Just drop overlapping fragments
- Bad fragment offsets exploit poor stack implementations
  - E.g. Teardrop attack, negative offsets or overlarge offsets cause buffer overflows
  - Firewalls can check for well formed packets.
- Resource attacks on re-assemblers
  - Send all but one fragment for many packets
- More: **An Analysis of Fragmentation Attacks**

  http://www.ouah.org/fragma.html

How TearDrop Works?


The router checks for discrepancies in the fragment offset field.


The second fragment purports to begin 20 bytes earlier (at 800) than the first fragment ends (at 820). The offset of fragment #2 is not in accord with the packet length of fragment #1. This discrepancy can cause some systems to crash during the reassembly attempt.

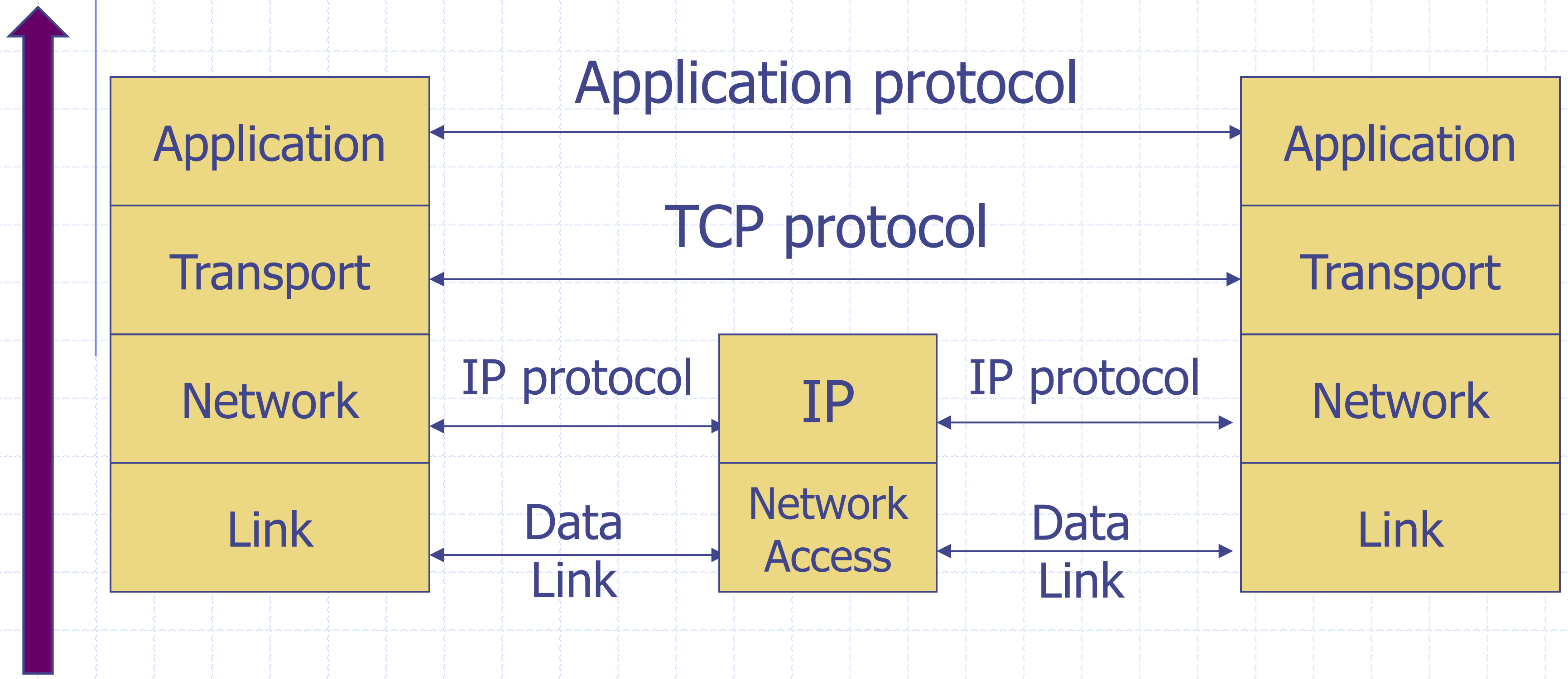
Types of Fragmentation Attacks

- *Ping O' Death Fragmentation Attack*
  - The Ping O' Death fragmentation attack is a denial of service attack, which utilizes a ping system utility to create an IP packet, which exceeds the maximum allowable size for an IP datagram of 65535 bytes.
  - This attack uses many small fragmented ICMP packets which when reassembled at the destination exceed the maximum allowable size for an IP datagram. This can cause the victim host to crash, hang or even reboot.
- *The Overlapping Fragment Attack*
  - Another variation on the teardrop attack that also uses overlapping fragments is the Overlapping Fragment Attack. This attack however is not a denial of service attack but it is used in an attempt to bypass firewalls to gain access to the victim host.
  - This attack can be used to overwrite part of the TCP header information of the first fragment, which contained data that was allowed to pass through the firewall, with malicious data in subsequent fragments. A common example of this is to overwrite the destination port number to change the type of service i.e. change from port 80 (HTTP) to port 23 (Telnet) which would not be allowed to pass the router in normal circumstances.
  - Ensuring a minimum fragment offset is specified in the router's IP filtering code can prevent this attack.

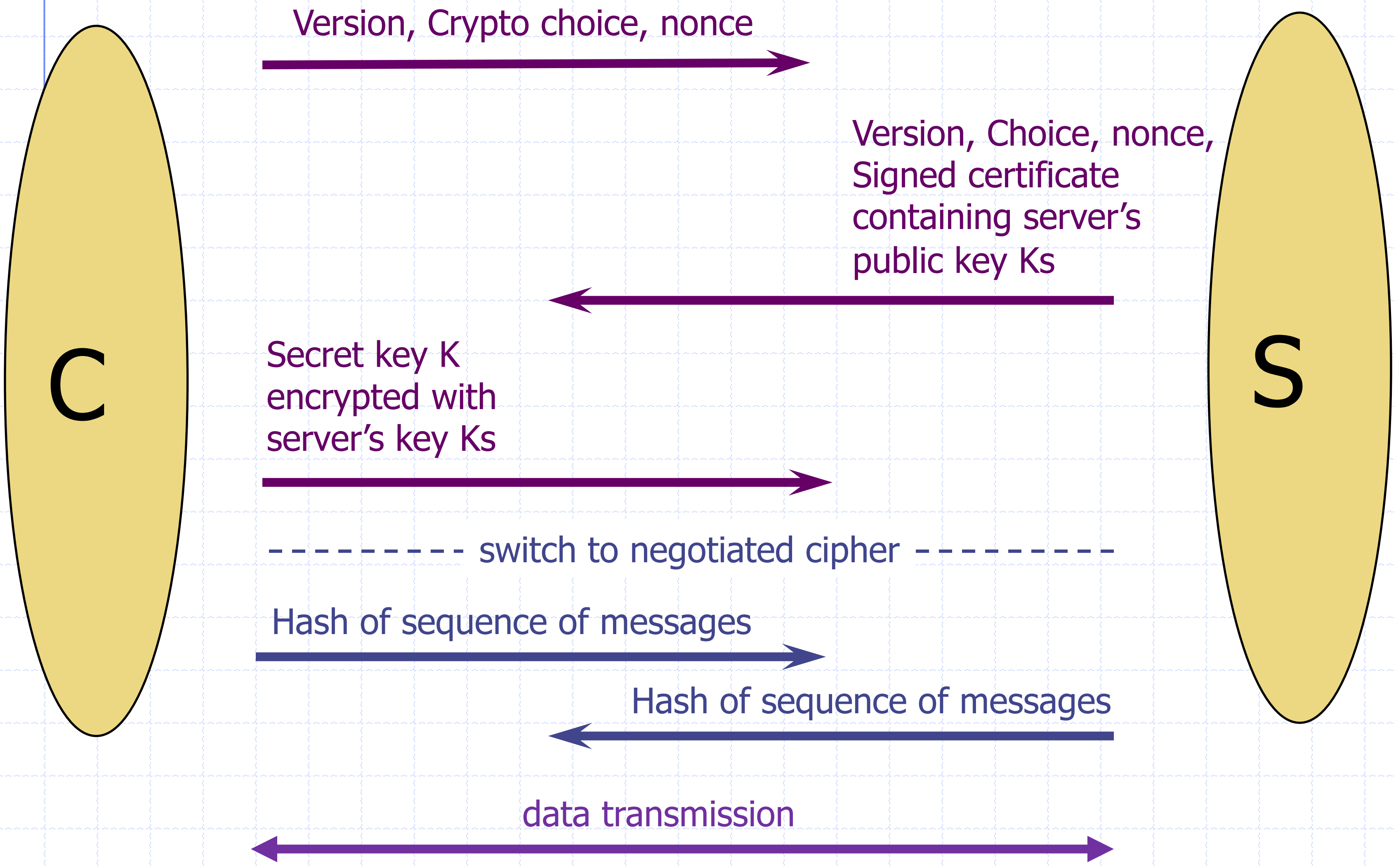
Packet Fragmentation Attack

- Firewall configuration
  - TCP port 23 is blocked but SMTP port 25 is allowed
- First packet
  - Fragmentation Offset = 0.
  - DF bit = 0 : "May Fragment"
  - MF bit = 1 : "More Fragments"
  - Destination Port = 25. TCP port 25 is allowed, so firewall allows packet
- Second packet
  - Fragmentation Offset = 1: second packet overwrites all but first 8 bits of the first packet
  - DF bit = 0 : "May Fragment"
  - MF bit = 0 : "Last Fragment."
  - Destination Port = 23. Normally be blocked, but sneaks by!
- What happens
  - Firewall ignores second packet “TCP header” because it is fragment of first
  - At host, packet reassembled and received at port 23

TCP Protocol Stack



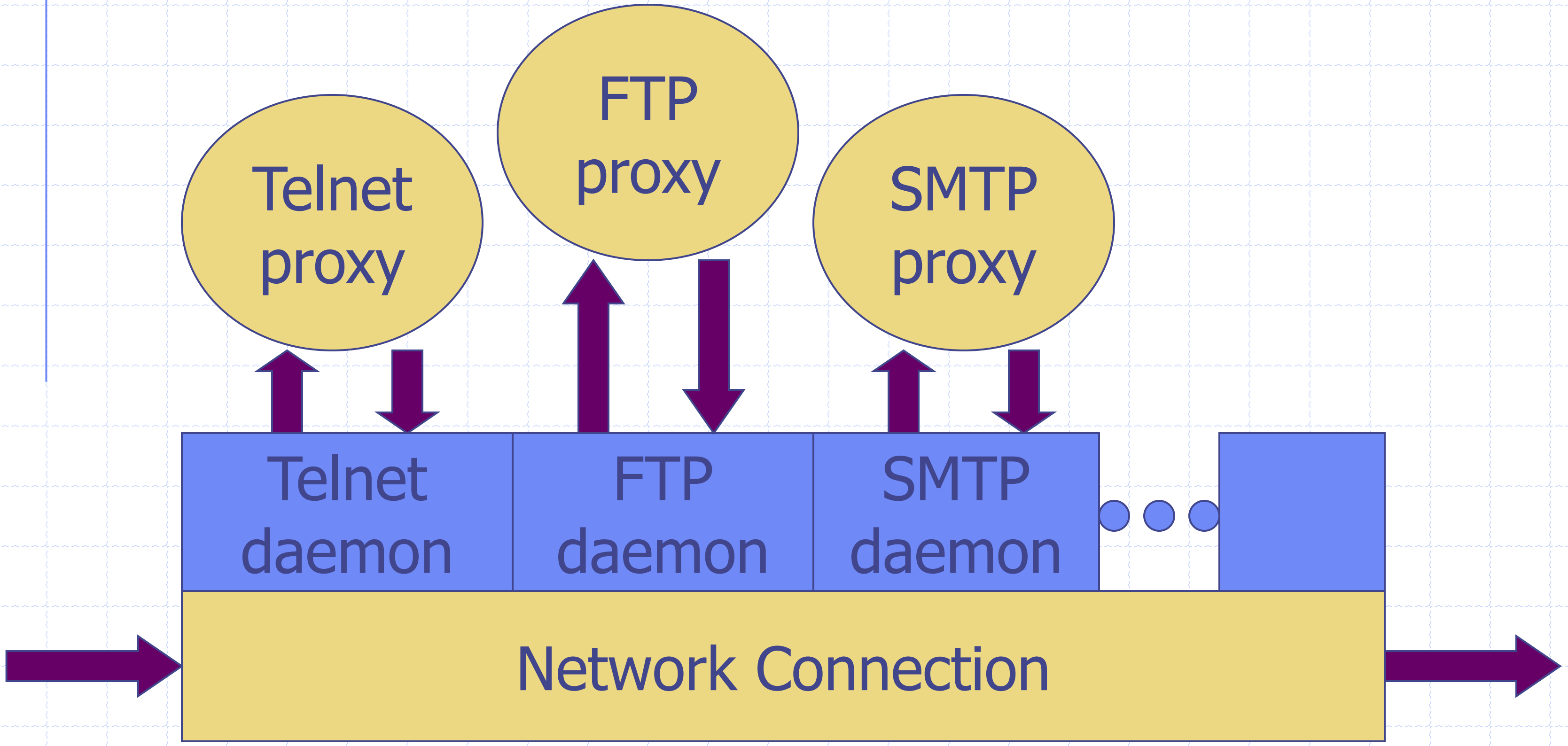
Remember SSL/TLS


C
S
Version, Crypto choice, nonce
Version, Choice, nonce, Signed certificate containing server's public key Ks
Secret key K encrypted with server's key Ks
---------- switch to negotiated cipher ----------
Hash of sequence of messages
Hash of sequence of messages
data transmission

Proxying Firewall

- Application-level proxies
  - Tailored to http, ftp, smtp, etc.
  - Some protocols easier to proxy than others
- Policy embedded in proxy programs
  - Proxies filter incoming, outgoing packets
  - Reconstruct application-layer messages
  - Can filter specific application-layer commands, etc.
    - Example: only allow specific ftp commands
    - Other examples: ?
- Several network locations – see next slides

Firewall with application proxies



Daemon spawns proxy when communication detected …

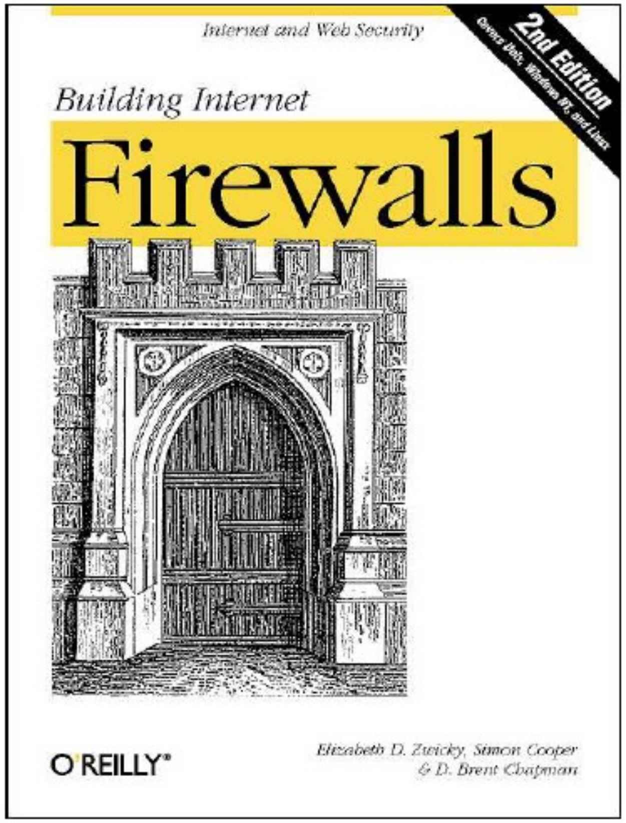
Application-level proxies

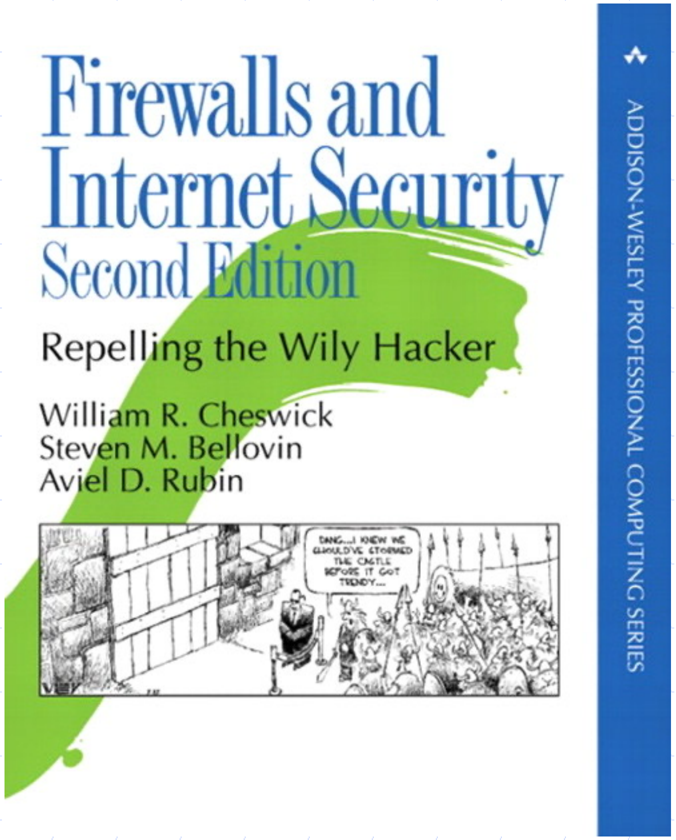
- Enforce policy for specific protocols
  - E.g., Virus scanning for SMTP
    - Need to understand MIME, encoding, Zip archives
  - Flexible approach, but may introduce network delays
- "Batch" protocols are natural to proxy
  - SMTP (E-Mail) NNTP (Net news)
  - DNS (Domain Name System) NTP (Network Time Protocol
- Must protect host running protocol stack
  - Disable all non-required services; keep it simple
  - Install/modify services you want
  - Run security audit to establish baseline
  - Be prepared for the system to be compromised

Web traffic scanning

- Intercept and proxy web traffic
  - Can be host-based
  - Usually at enterprise gateway
- Block known bad sites
- Block pages with known attacks
- Scan attachments
  - Usually traditional virus scanning methods

Firewall references


Internet and Web Security
2nd Edition
Covers Unix, Windows NT, and Linux
Building Internet
Firewalls
O'REILLY
Elizabeth D. Zwicky, Simon Cooper & D. Brent Chapman


Firewalls and Internet Security
Second Edition
Repelling the Wily Hacker
William R. Cheswick
Steven M. Bellovin
Aviel D. Rubin
DANG... I KNEW WE SHOULD'VE STORMED THE CASTLE BEFORE IT GOT TRENDY...
ADDISON-WESLEY PROFESSIONAL COMPUTING SERIES

Elizabeth D. Zwicky
Simon Cooper
D. Brent Chapman

William R Cheswick
Steven M Bellovin
Aviel D Rubin