Network Security Defense Tools

Firewalls and Intrusion Detection

Christoph Schuba
Senior Research Staff
Sun Microsystems, Inc.

Slides: John Mitchell

Security Posture

- Prevention

  vs.

- Detection, Recovery, and Response

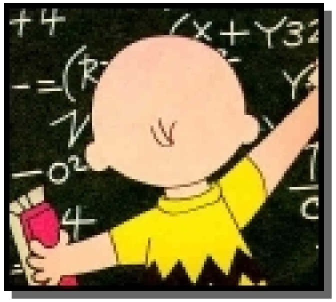
Security Posture (cont.)

This lecture

- Standard perimeter defense mechanisms (Bag of tricks)
  - Firewall
    - Packet filter (stateless, stateful)
    - Application layer proxies
  - Intrusion detection
    - Anomaly and misuse detection
    - Methods applicable to network or host

Perimeter and Internal Defenses

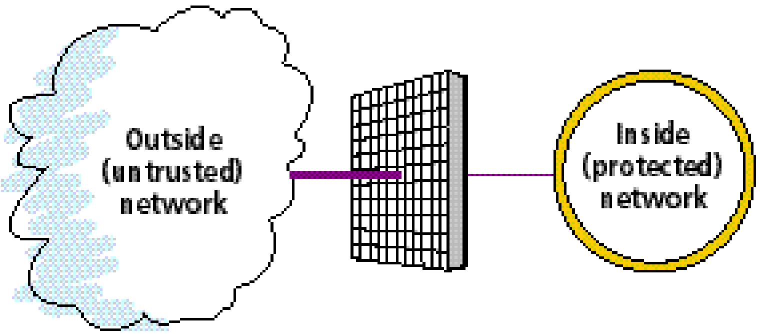
(bag of tricks)



- Commonly deployed defenses
  - Perimeter defenses – Firewall, IDS
    - Protect local area network and hosts
    - Keep external threats from internal network
  - Internal defenses – Virus scanning
    - Protect hosts from threats that get through the perimeter defenses
  - Extend the "perimeter" – VPN
- Common practices, but could be improved
  - Internal threats are significant
    - Unhappy employees
    - Compromised hosts

Firewall Technology - A Definition

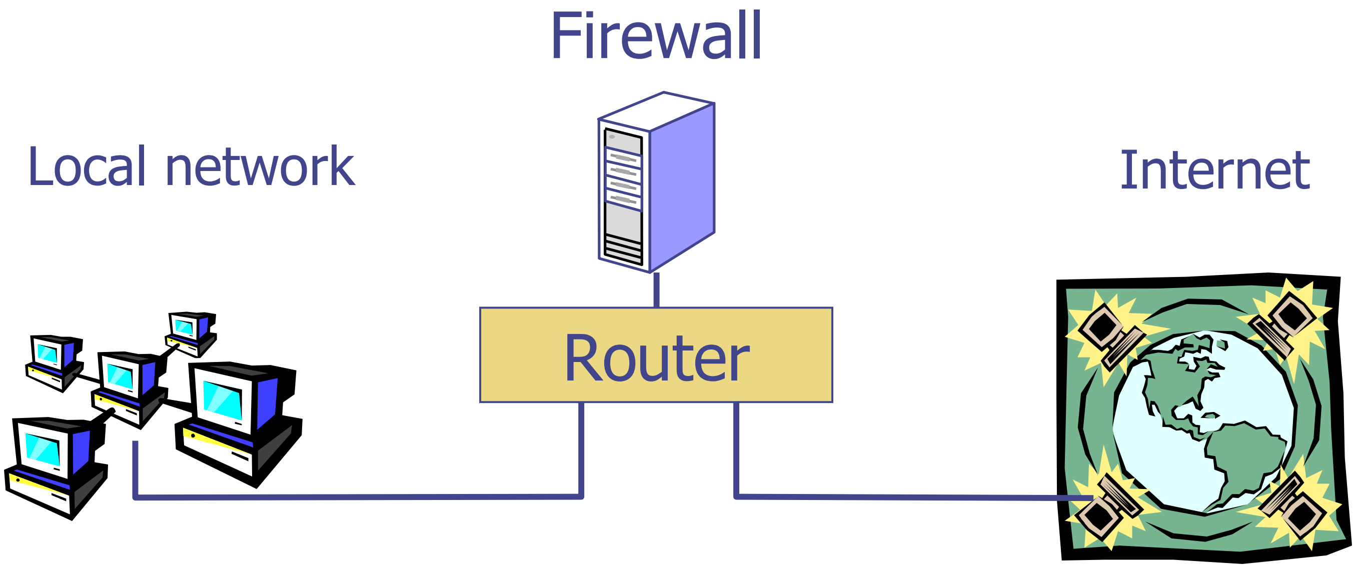
We define firewall technology as a set of mechanisms that collectively enforce a network domain security policy on communication traffic entering or leaving a guarded network policy domain.



A firewall system, or firewall is an instantiation of firewall technology.

Basic Firewall Concept

- Separate local area net from internet



All packets between LAN and internet routed through firewall

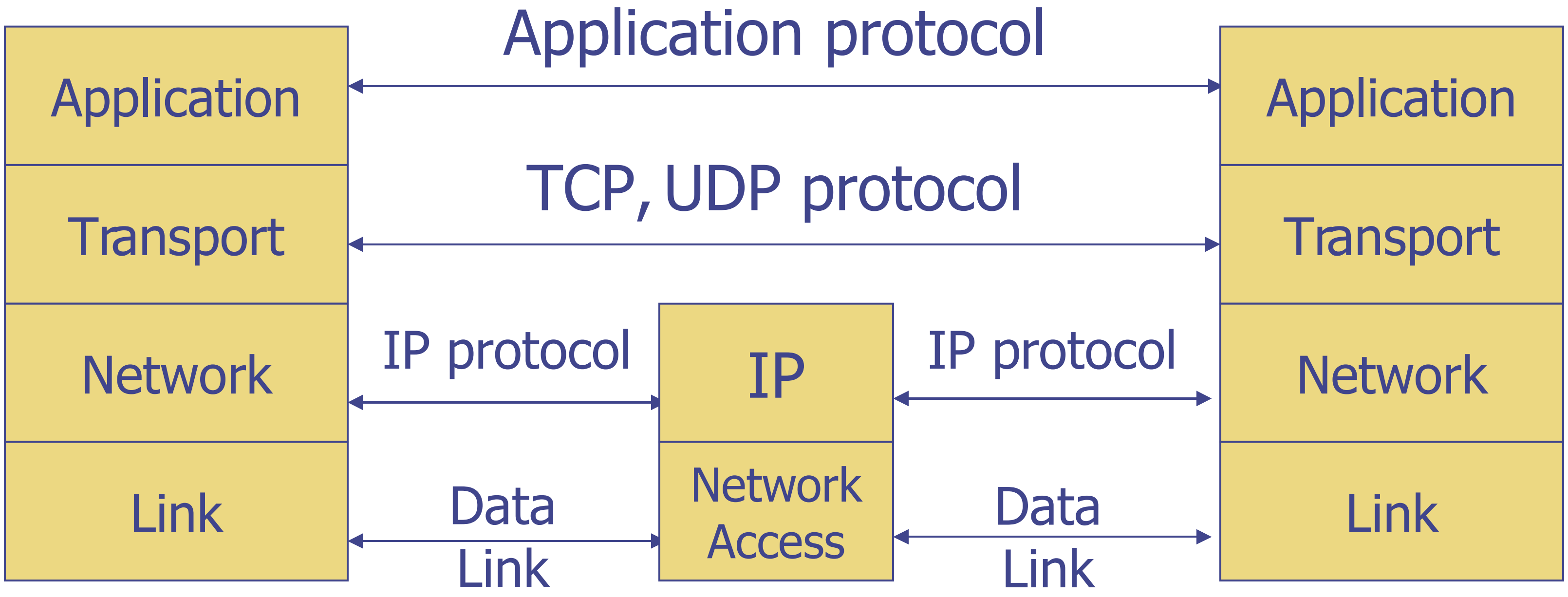
Firewall goals

- Prevent malicious attacks on hosts
  - Port sweeps, ICMP echo to broadcast addr, syn flooding, …
  - Worm propagation
    - Exploit buffer overflow in program listening on network
- Prevent general disruption of internal network
  - External SMNP packets
- Provide defense in depth
  - Programs contain bugs and are vulnerable to attack
  - Network protocols may contain;
    - Design weaknesses (SSH CRC)
    - Implementation flaws (SSL, NTP, FTP, SMTP...)
- Control traffic between "zones of trusts"
  - Can control traffic between separate local networks, etc

Two Separable Topics

- Arrangement of firewall and routers
    - Several different network configurations
        - Separate internal LAN from external Internet
        - Wall off subnetwork within an organization
        - Intermediate zone for web server, etc.
    - Personal firewall on end-user machine
- How the firewall processes data
    - Packet filtering router
    - Application-level gateway
        - Proxy for protocols such as ftp, smtp, http, etc.
    - Personal firewall
        - E.g., disallow telnet connection from email client

Review: TCP Protocol Stack



Application protocol

TCP, UDP protocol

IP protocol

IP protocol

Data Link

Data Link

Application

Transport

Network

Link

IP

Network Access

Application

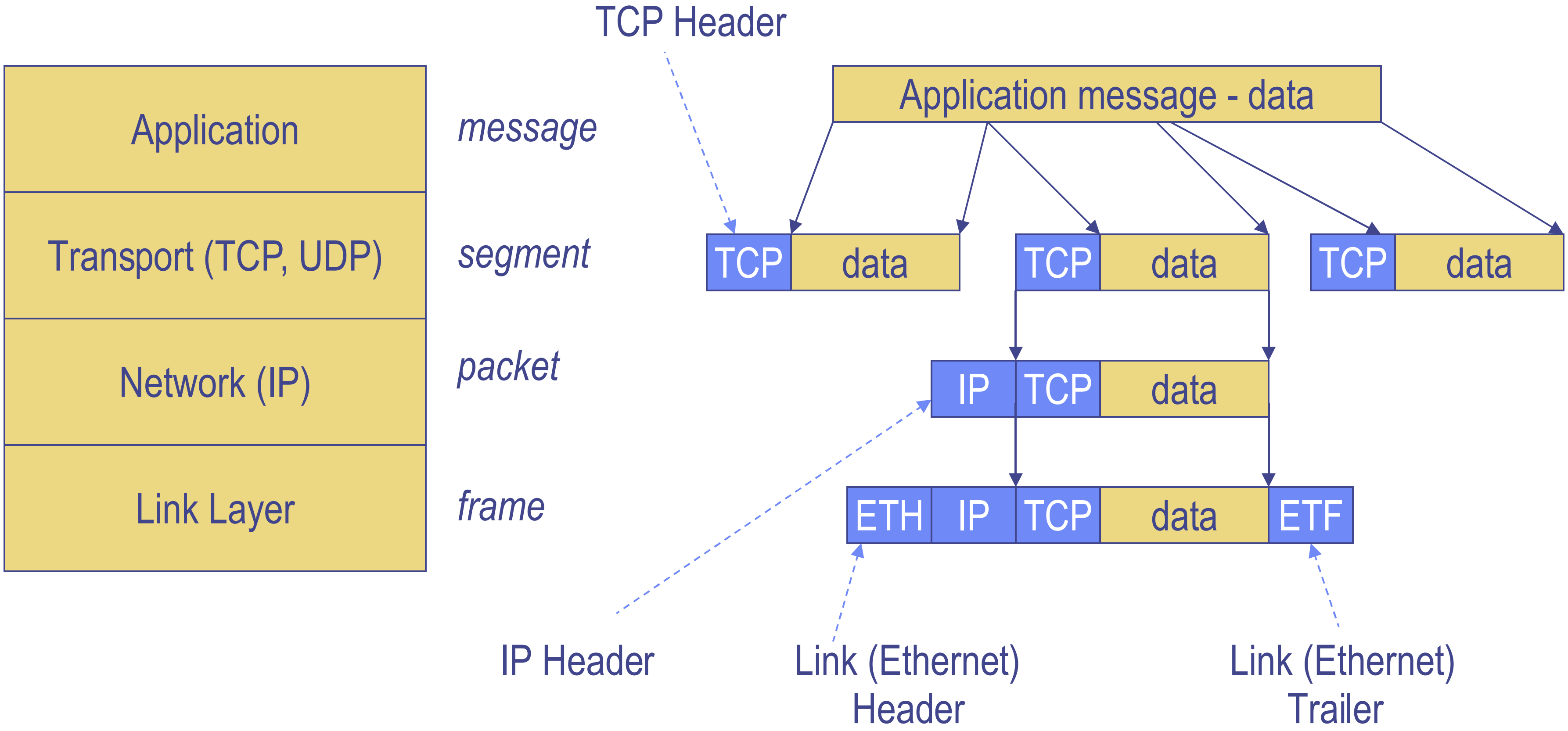
Transport

Network

Link

Transport layer provides *ports*, logical channels identified by number

Review: Data Formats



TCP Header

Application

Transport (TCP, UDP)

Network (IP)

Link Layer

message

segment

packet

frame

Application message - data

TCP data TCP data TCP data

IP TCP data

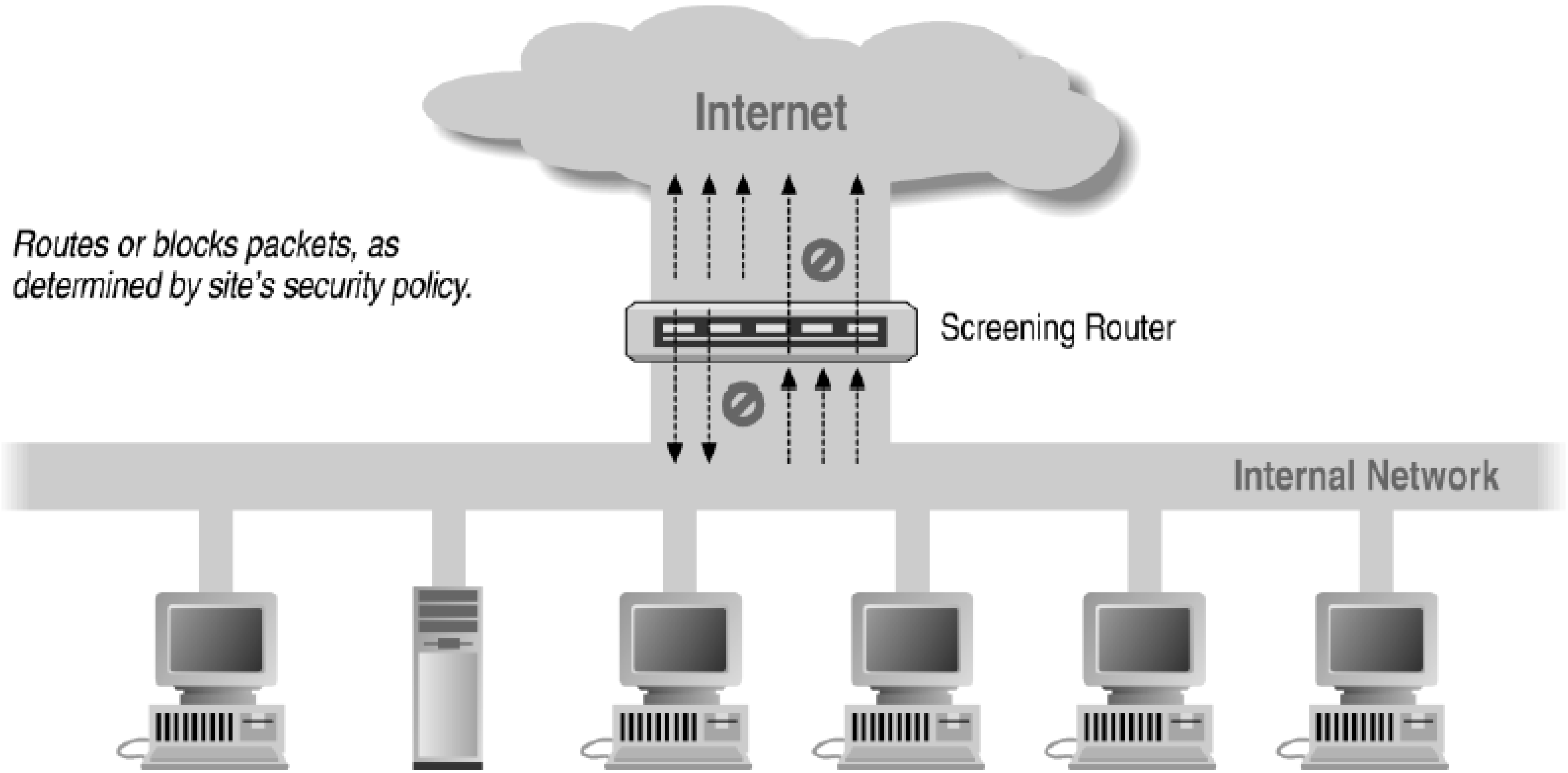
ETH IP TCP data ETF

IP Header

Link (Ethernet)
Header

Link (Ethernet)
Trailer

Screening router for packet filtering

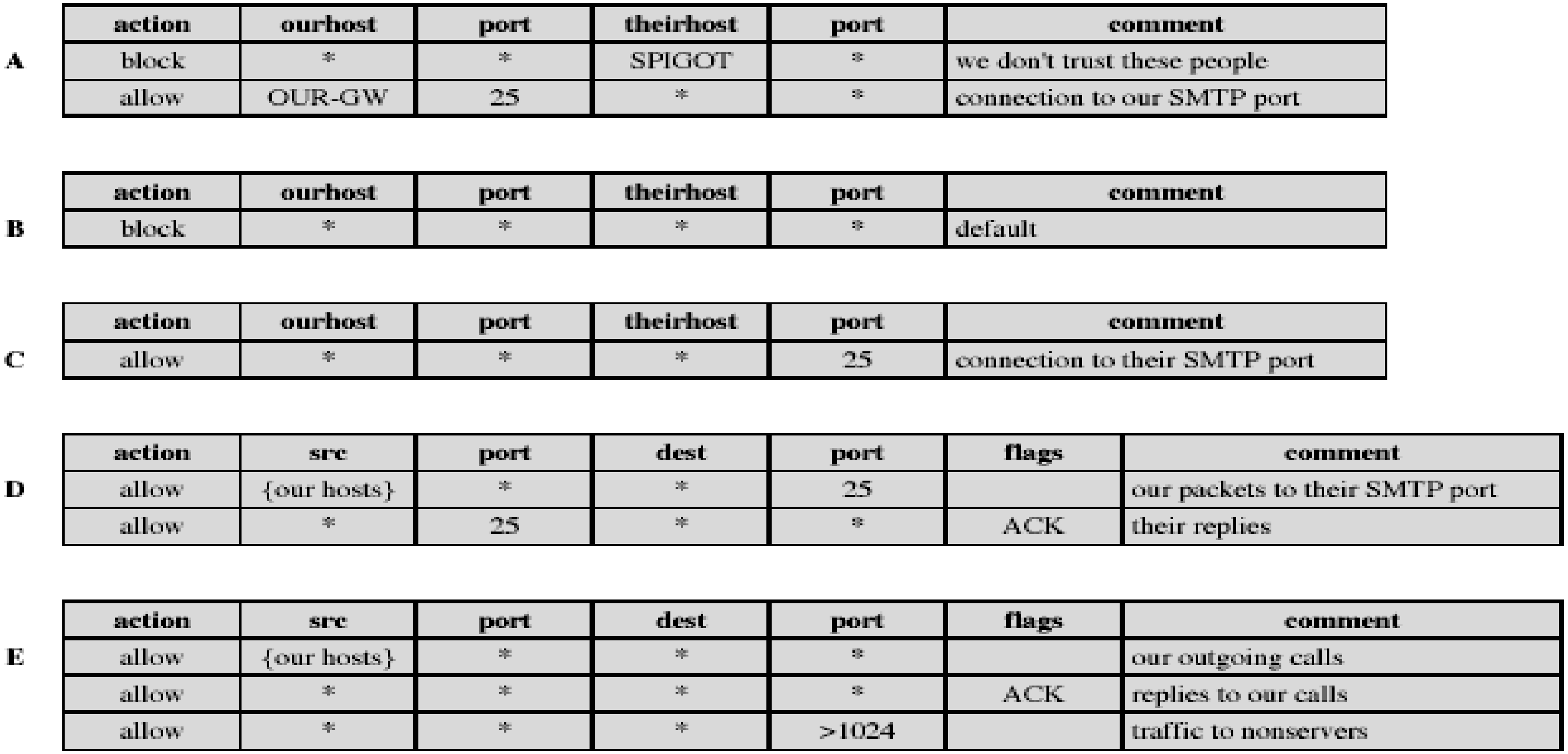

Internet
Routes or blocks packets, as determined by site's security policy.
Screening Router
Internal Network

Illustrations: Simon Cooper

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

Packet filtering examples



A

action	ourhost	port	theirhost	port	comment
block	*	*	SPIGOT	*	we don't trust these people
allow	OUR-GW	25	*	*	connection to our SMTP port

B

action	ourhost	port	theirhost	port	comment
block	*	*	*	*	default

C

action	ourhost	port	theirhost	port	comment
allow	*	*	*	25	connection to their SMTP port

D

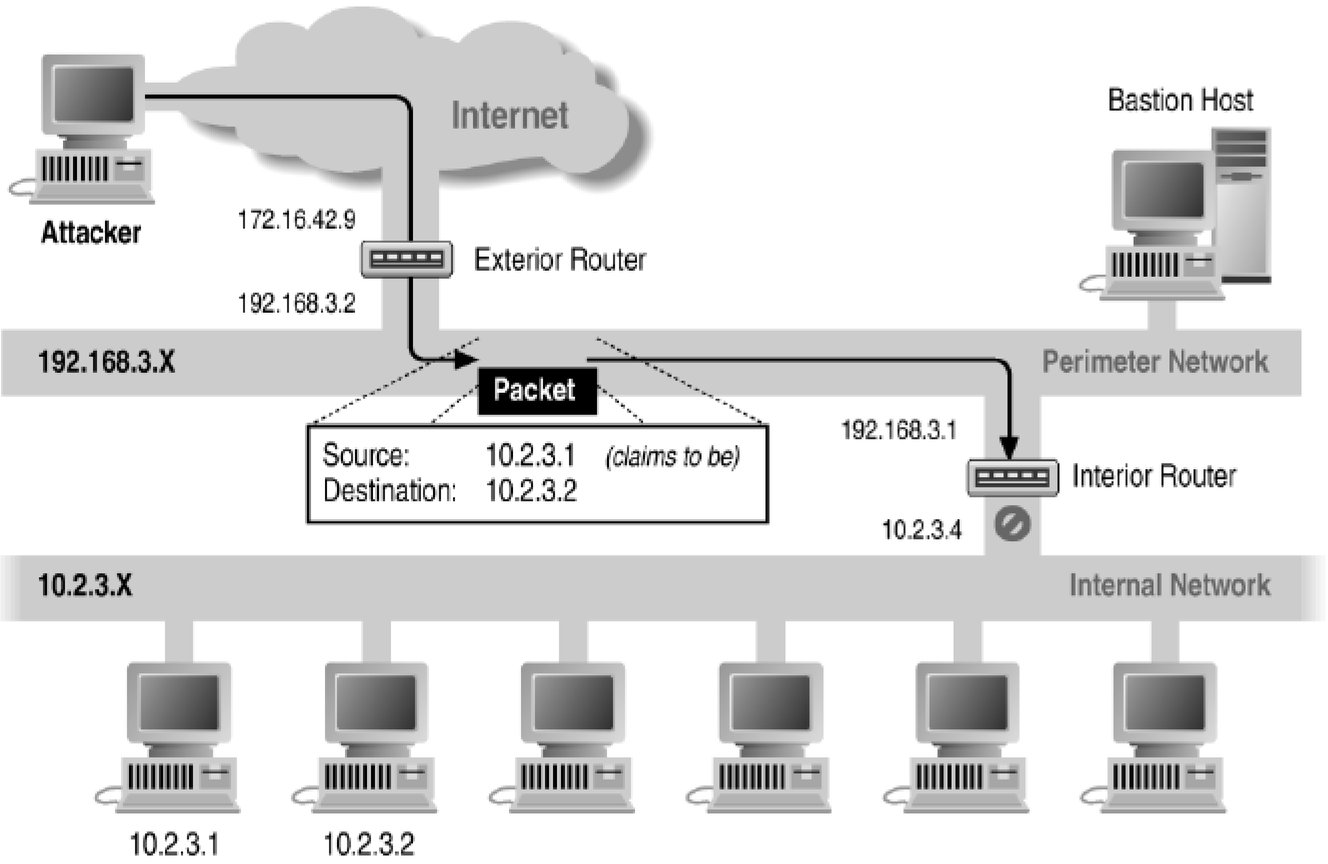
action	src	port	dest	port	flags	comment
allow	{our hosts}	*	*	25		our packets to their SMTP port
allow	*	25	*	*	ACK	their replies

E

action	src	port	dest	port	flags	comment
allow	{our hosts}	*	*	*		our outgoing calls
allow	*	*	*	*	ACK	replies to our calls
allow	*	*	*	>1024		traffic to nonservers

Compare: Tiny Personal Firewall, ZoneAlarm

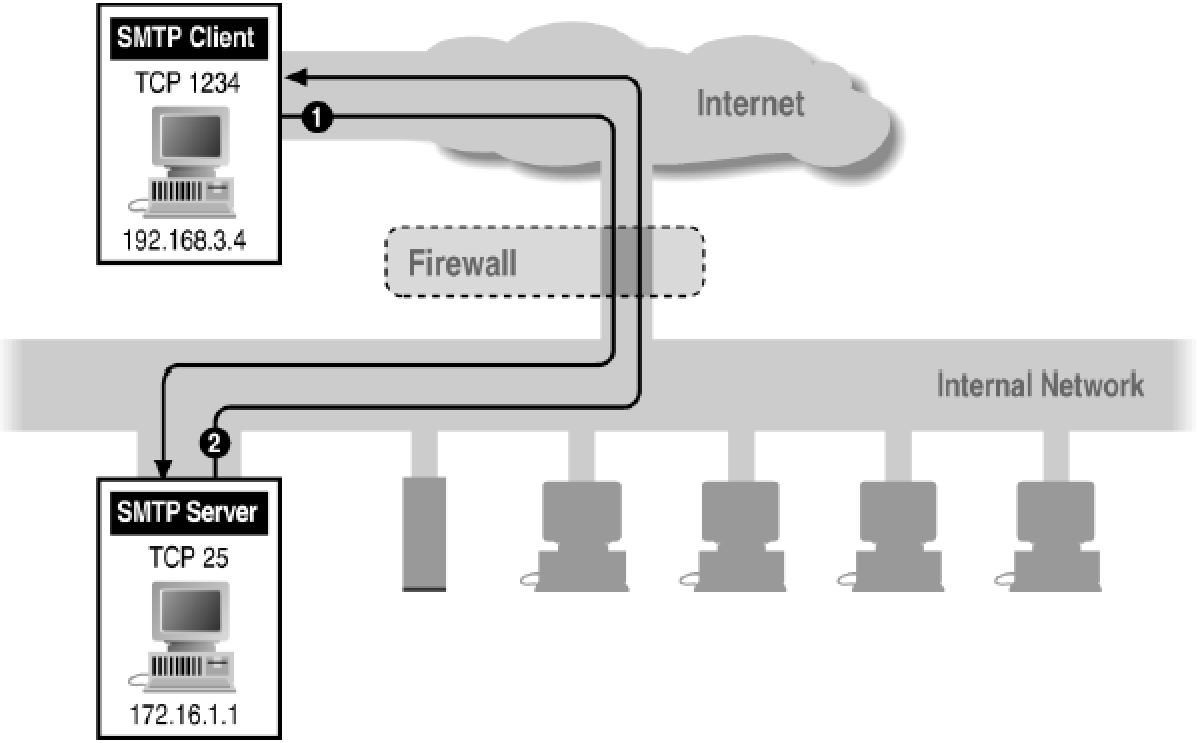
Source/Destination Address Forgery



More about networking: port numbering

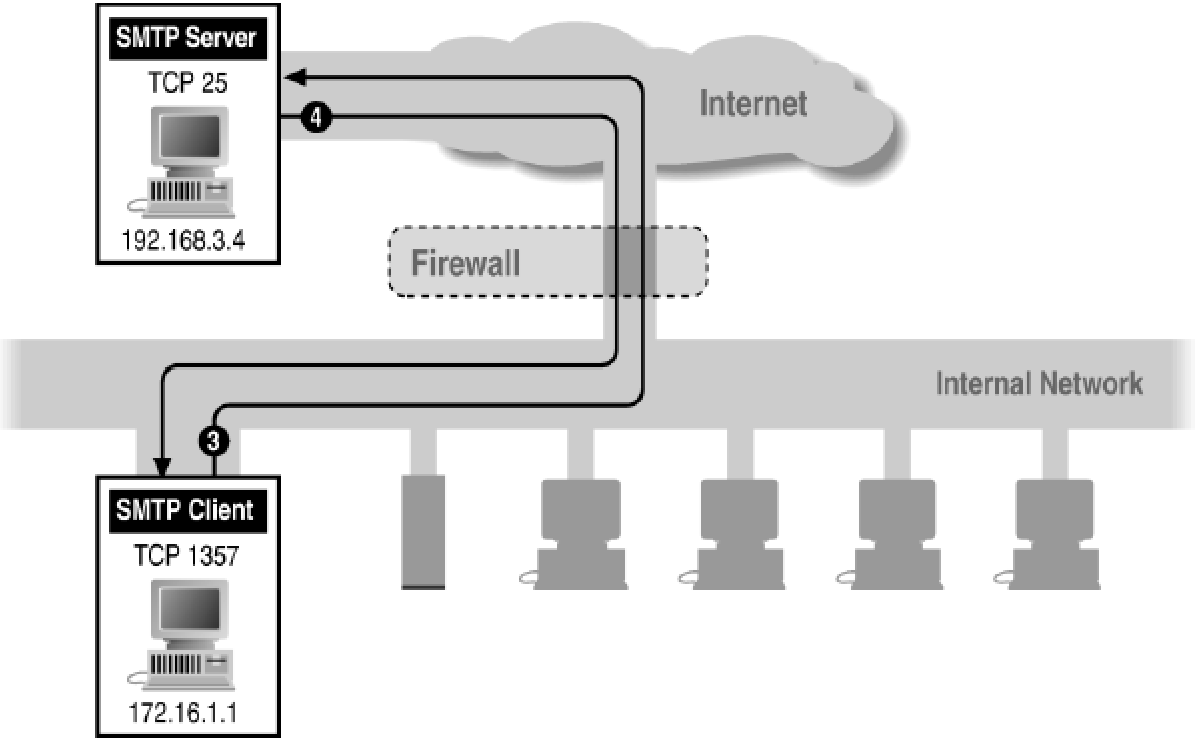
- Port numbers (http://www.iana.org/assignments/port-numbers)
  - Well known ports 0 .. 1023
  - DCCP registered ports: 1024 .. 49151
  - Dynamic/private ports: 49152 .. 65535
- Permanent assignment examples
  - Ports <1024 assigned permanently
    - 20,21 for FTP 23 for Telnet
    - 25 for server SMTP 80 for HTTP
- Variable use
  - 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

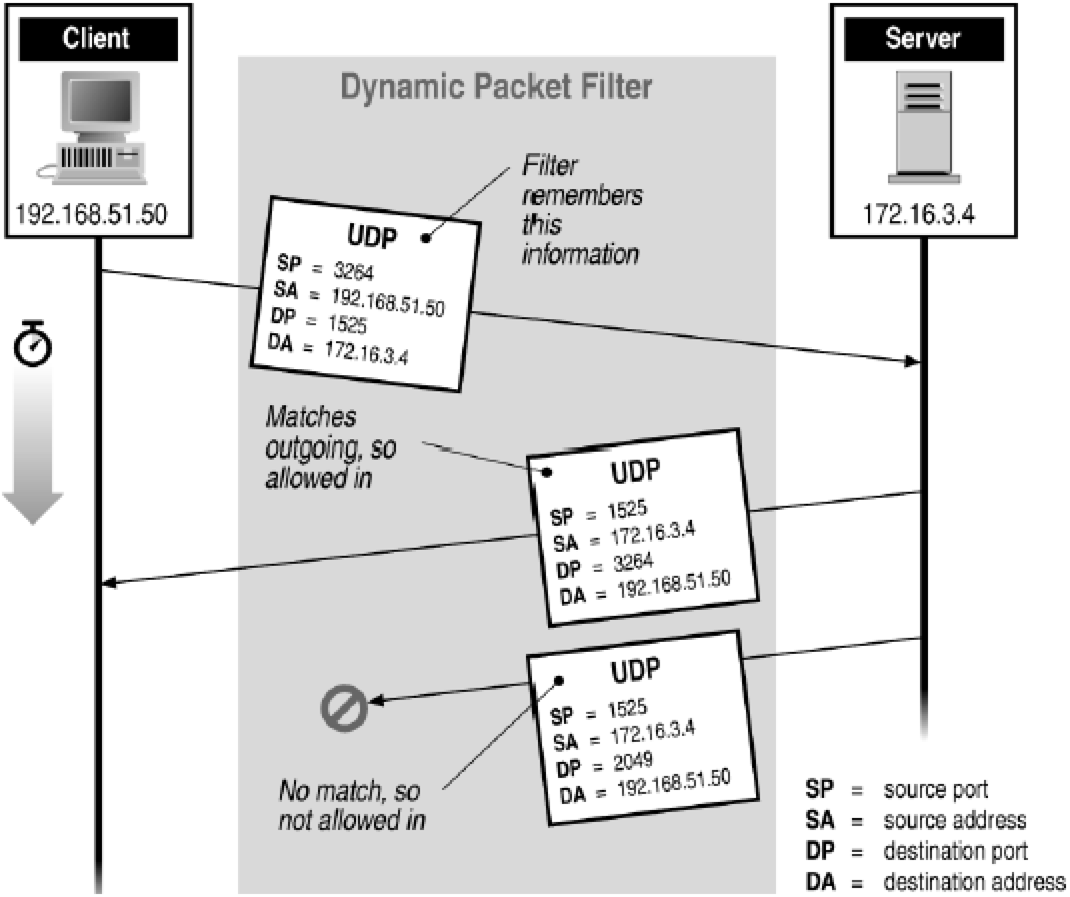
Filtering Example: Outbound SMTP



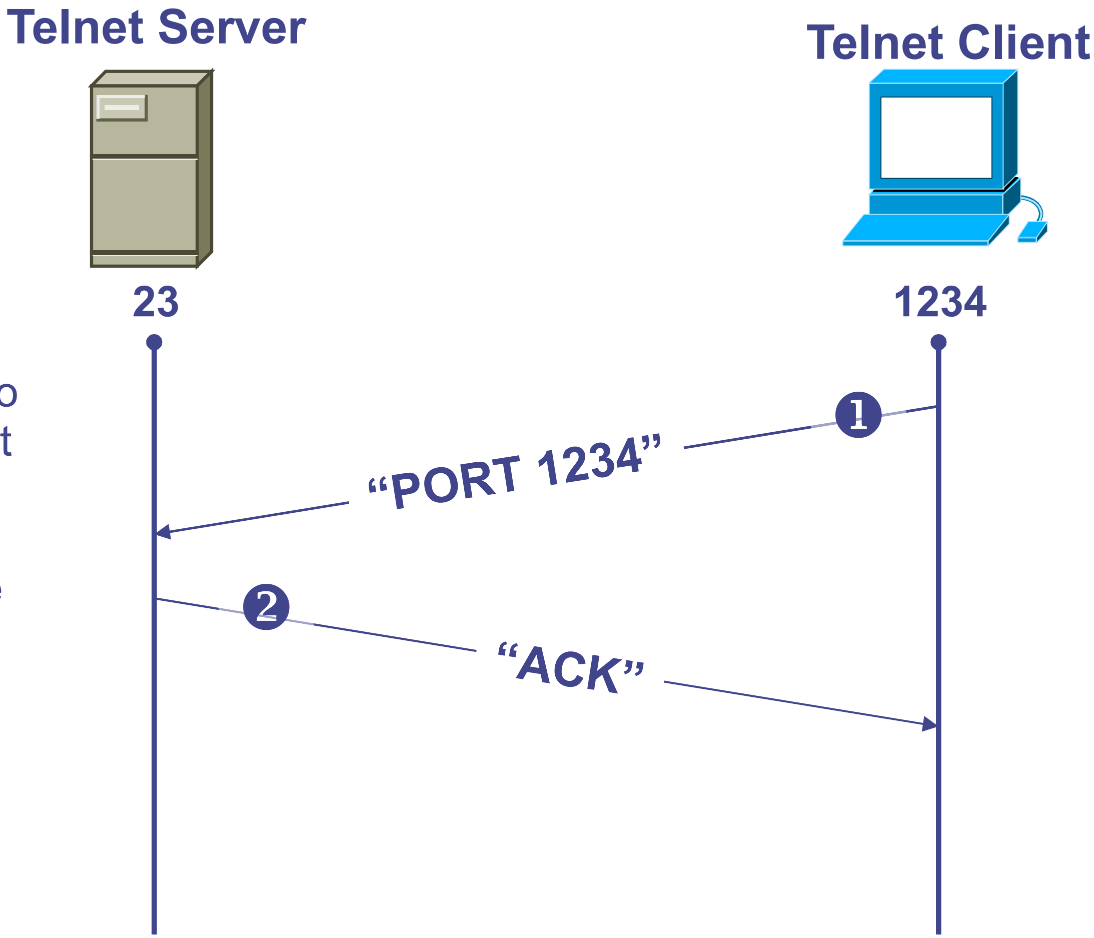
Known low port out, arbitrary high port in

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

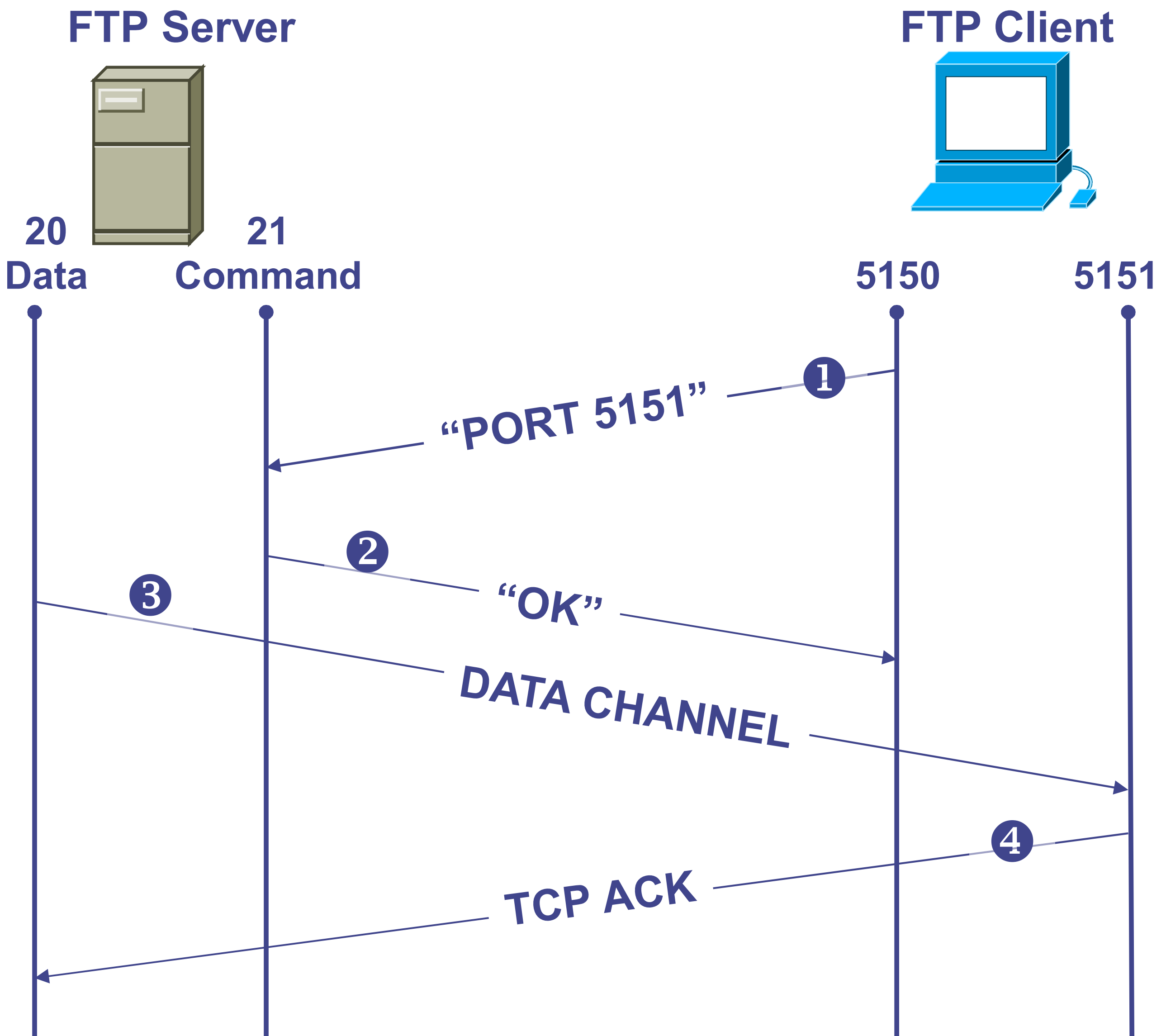
FTP

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

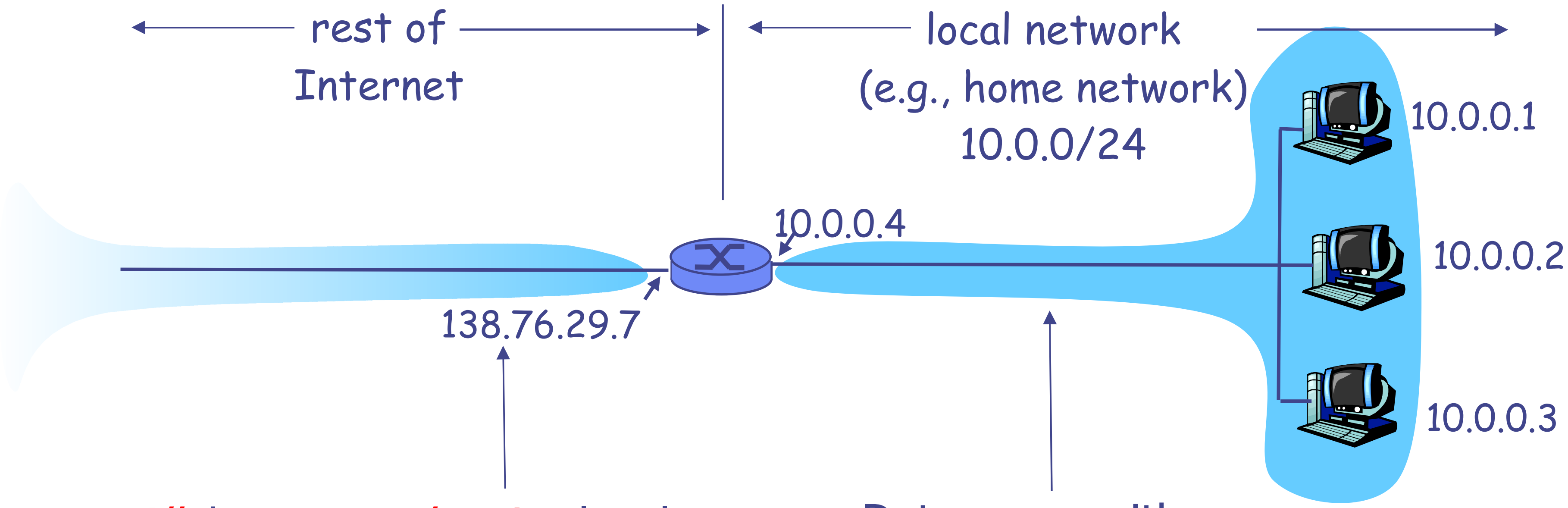
❷ Server acknowledges

❸ Server opens data channel to client's second port

❹ Client acknowledges



NAT: Network Address Translation



All datagrams *leaving* local network have same single source NAT IP address: 138.76.29.7, different source port numbers

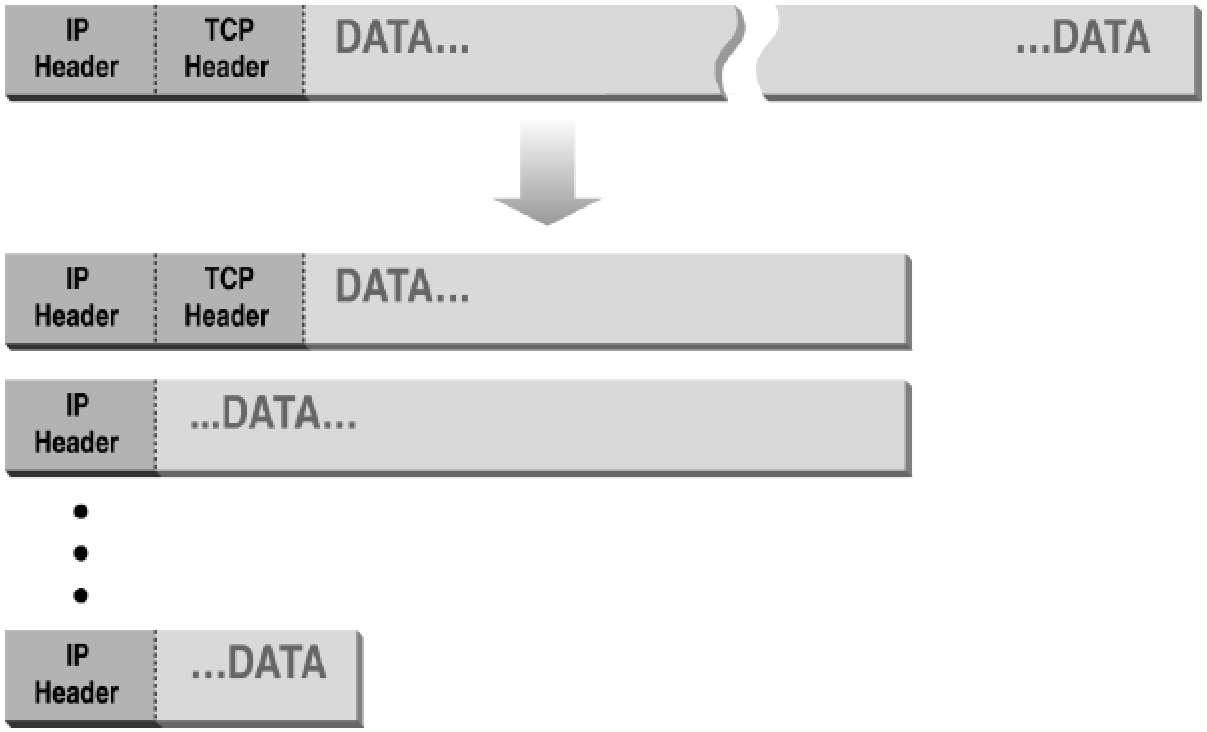
Datagrams with source or destination in this network have 10.0.0/24 address for source, destination (as usual)

Illustration: Kurose and Ross

Advantages of NAT

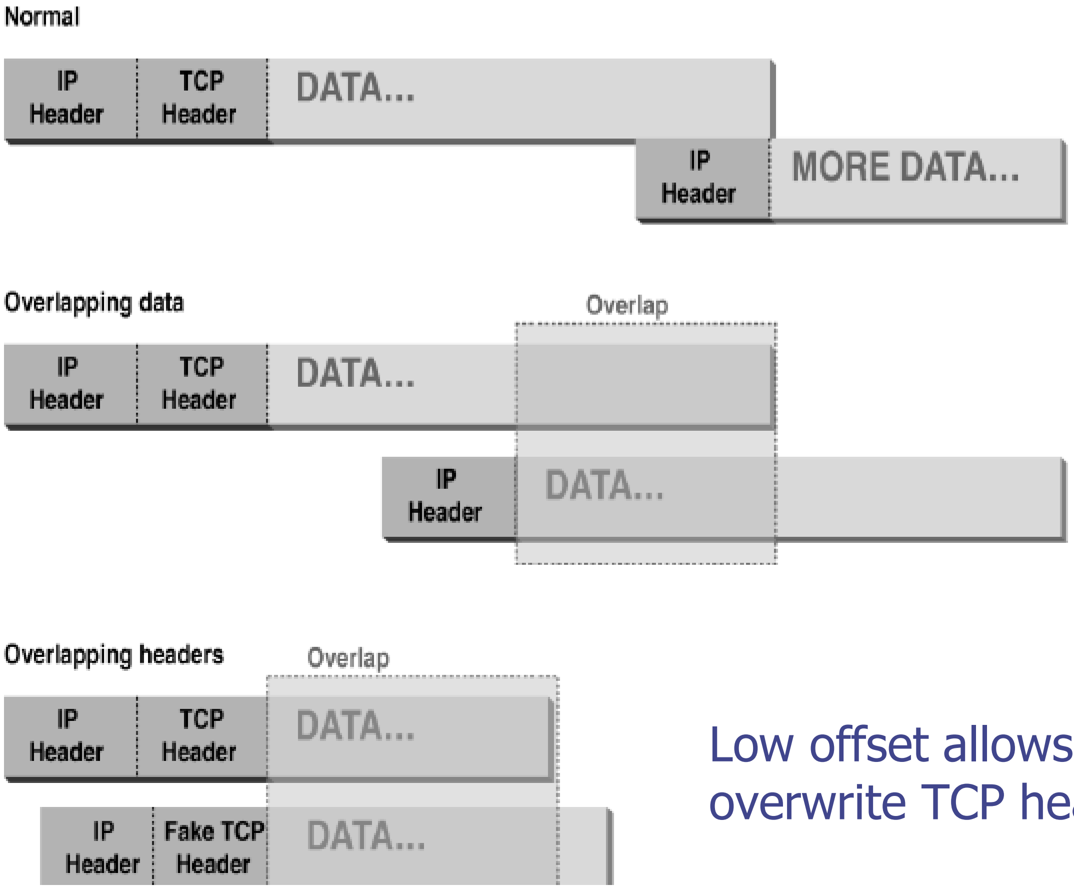
- Motivations for NAT
  - Limited address space
  - Prevent unsolicited inbound requests
    - Port numbering: host behind NAT not reachable as server
  - Avoid renumbering if provider changes
    - Small/mid-sized LANs inherit address space from ISP
- Addresses hidden by NAT
  - Normal routing
    - Outgoing msg from 171.64.78.90 contains sending address
    - Recipient or observer can access 171.64.78.90
  - Addressing with NAT
    - NAT rewrites outgoing packet so recipient sees public addr only
    - An outside computer cannot see 171.64.78.90

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

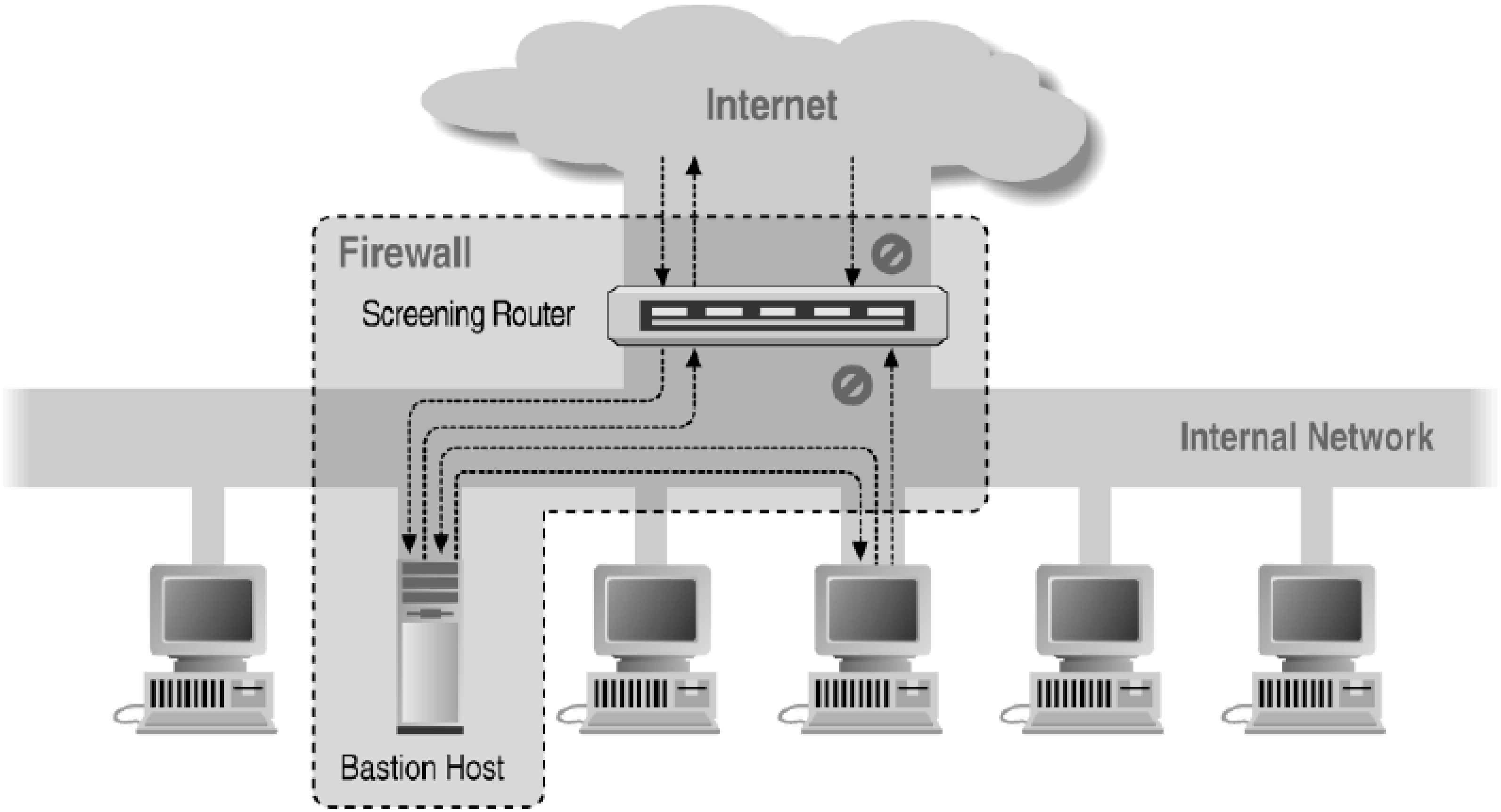
Packet Fragmentation Attack

- 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

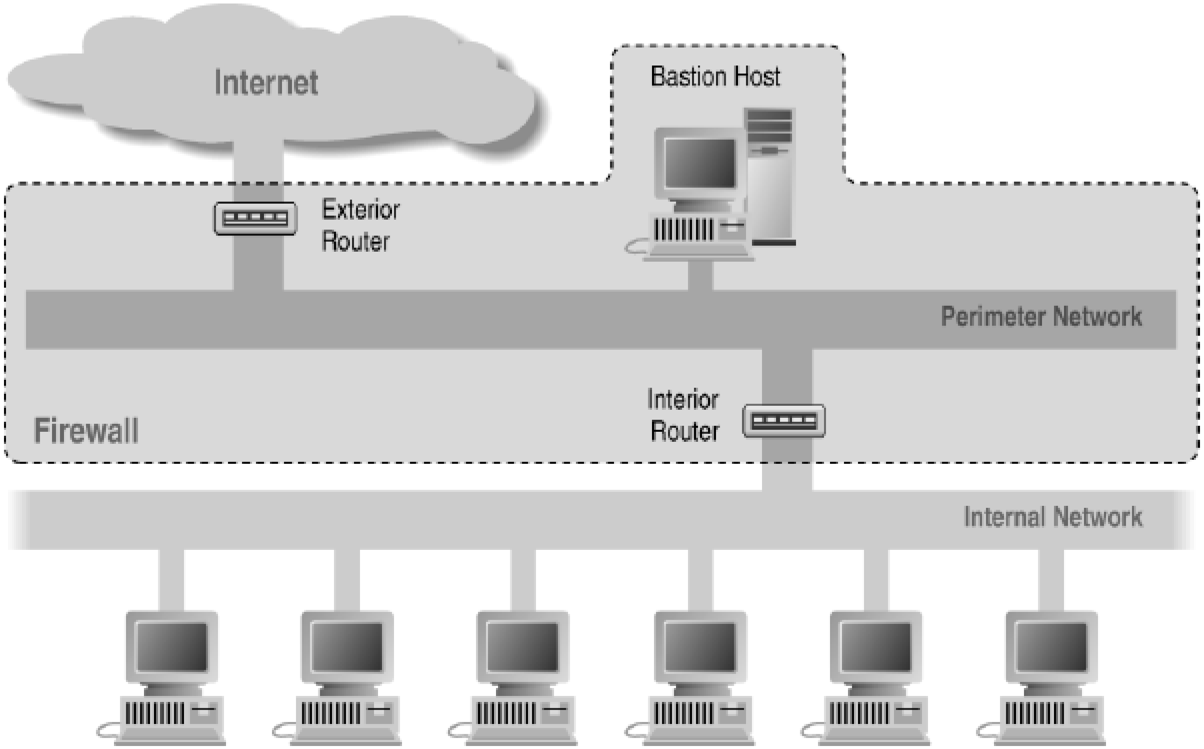
Proxying Firewall

- Several network locations – see next slides
- Two kinds of proxies
  - Circuit-level proxies
    - Works at session layer (which I omitted from OSI diagram)
  - 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: ?

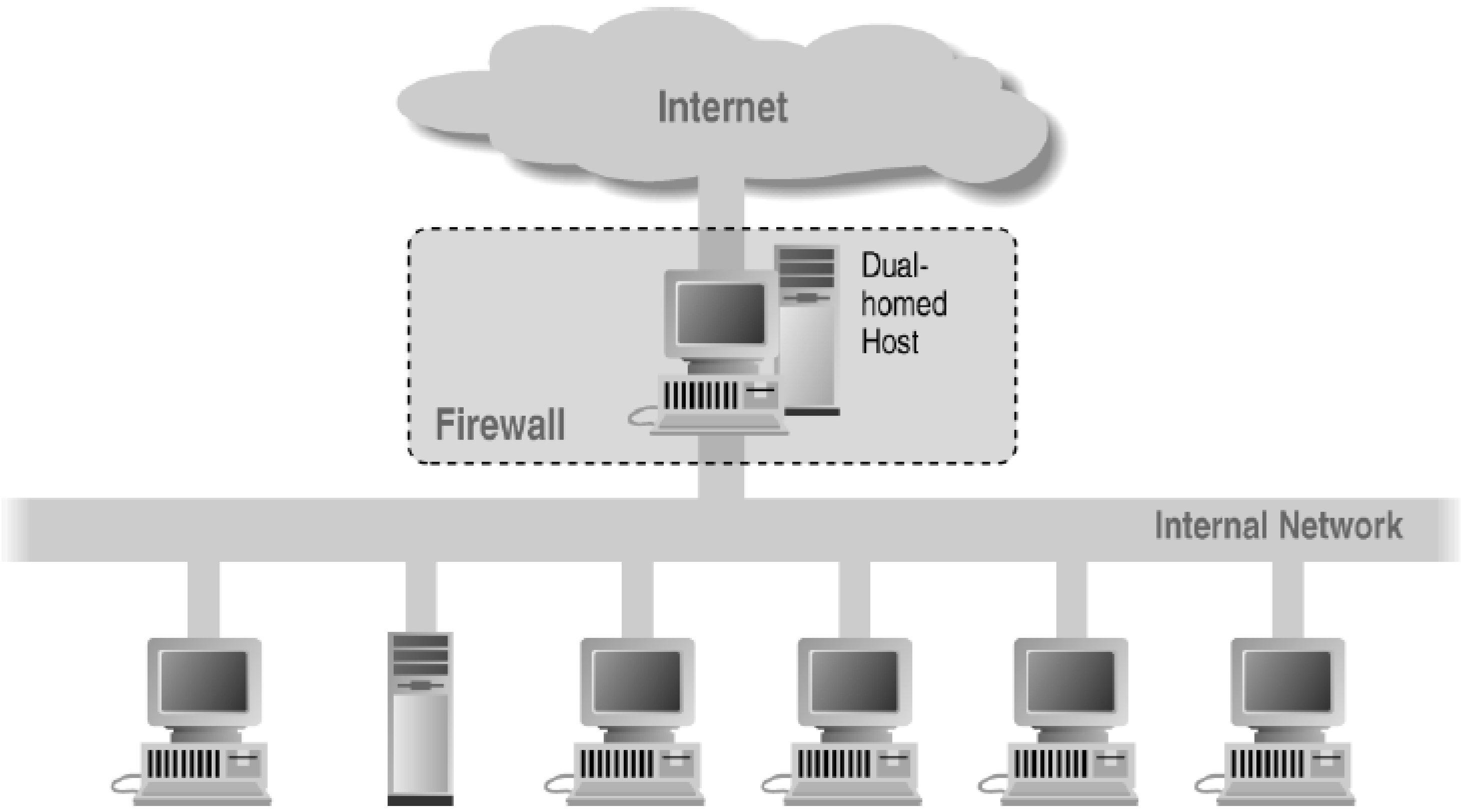
Screened Host Architecture



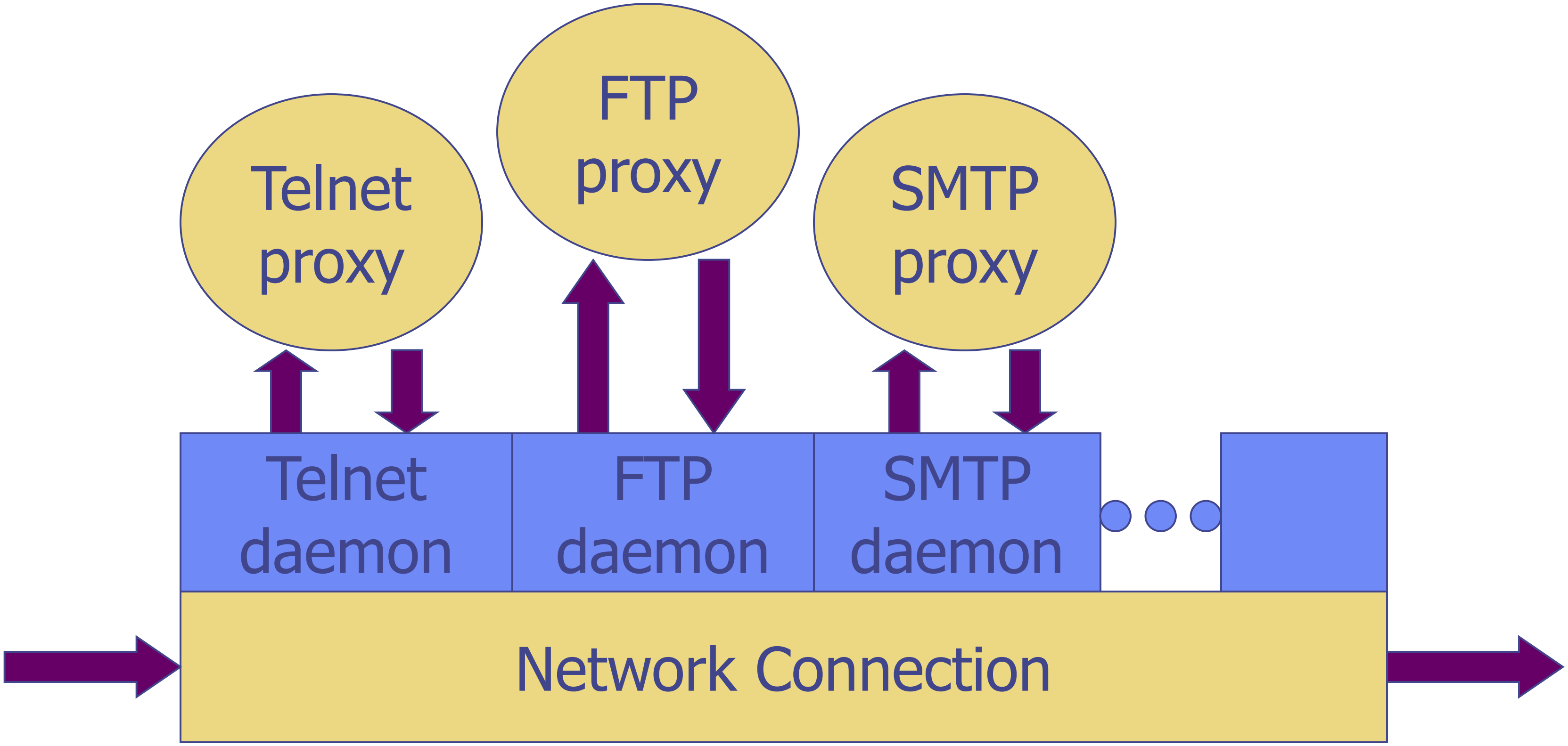
Screened Subnet Using Two Routers



Dual Homed Host Architecture



Firewall with application proxies

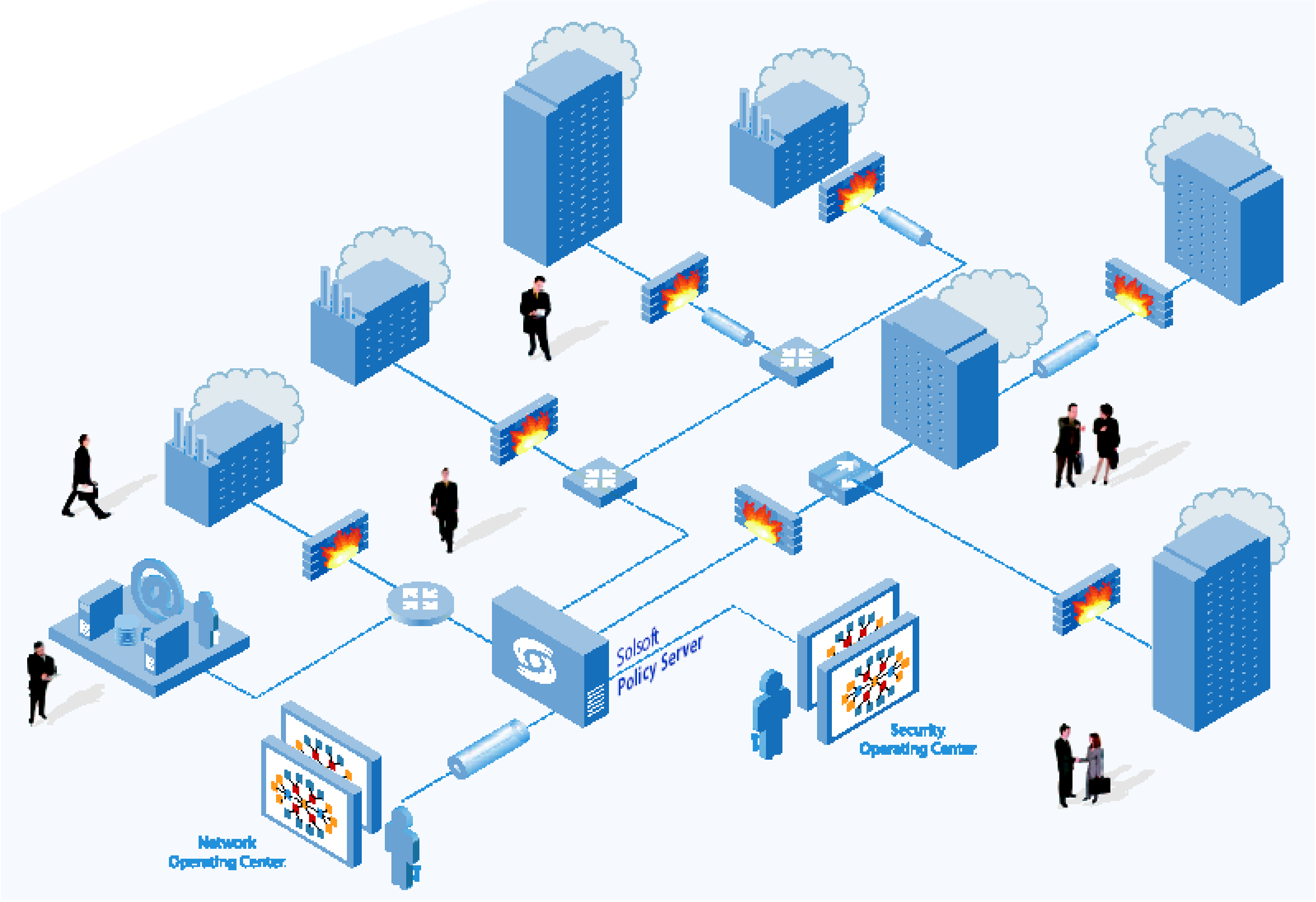


Daemon spawns proxy when communication detected …

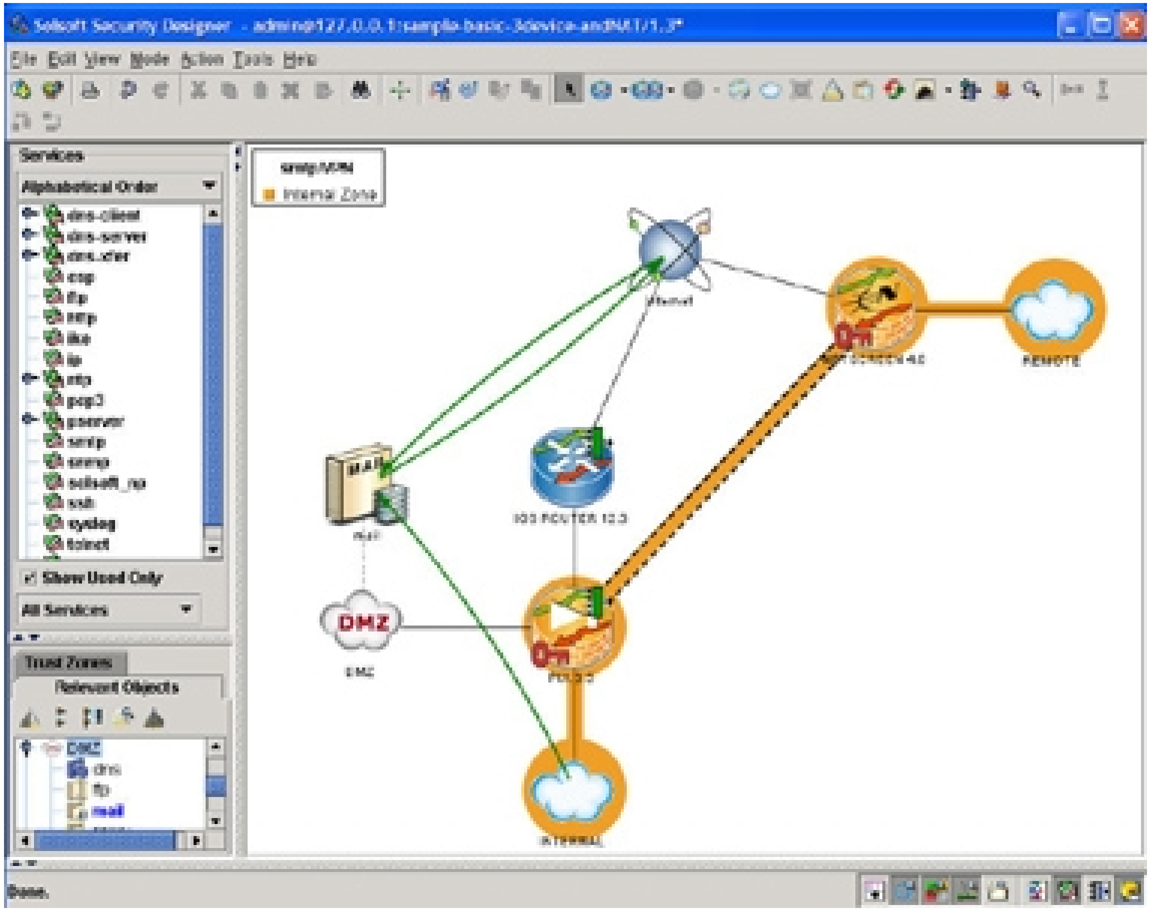
Application-level proxies

- 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

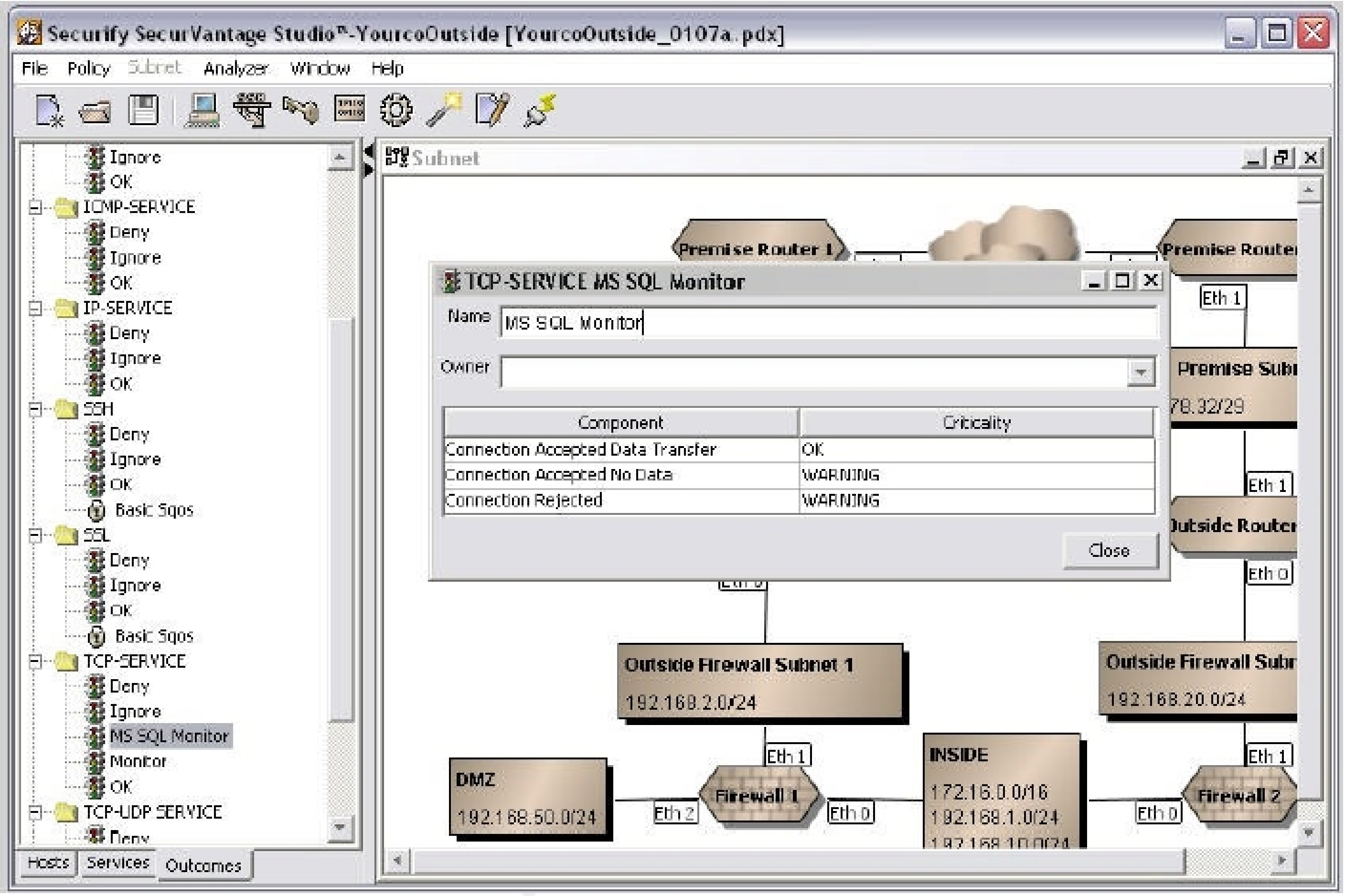
Configuration issues



Solsoft


Solsoft Security Designer - admin@127.0.0.1:sample-basic-3device-andNAT/1.3*
File Edit View Node Action Tools Help
Services
Alphabetical Order
dns-client
dns-server
dns_xfer
esp
ftp
http
ike
ip
ntp
pop3
pserver
smtp
snmp
solsoft_np
ssh
syslog
telnet
Show Used Only
All Services
Trust Zones
Relevant Objects
DMZ
dns
ftp
mail
smtp/VPN
Internal Zone
Internet
REMOTE
IOS ROUTER 12.3
mail
DMZ
DMZ
INTERNAL
Done.

Securify



Name: MS SQL Monitor

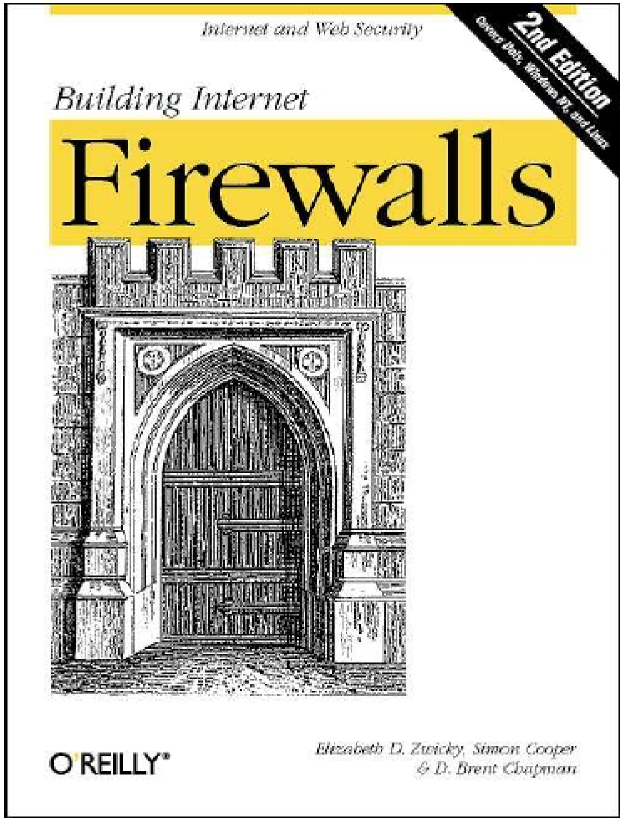
Owner:

Component	Criticality
Connection Accepted Data Transfer	OK
Connection Accepted No Data	WARNING
Connection Rejected	WARNING

Problems with Firewalls

- Performance
  - Firewalls may interfere with network use
- Limitations
  - They don't solve deeper problems
    - Buggy software
    - Bad protocols
  - Generally cannot prevent Denial of Service
  - Ineffective against insider attacks
- Administration
  - Many commercial firewalls permit very complex configurations

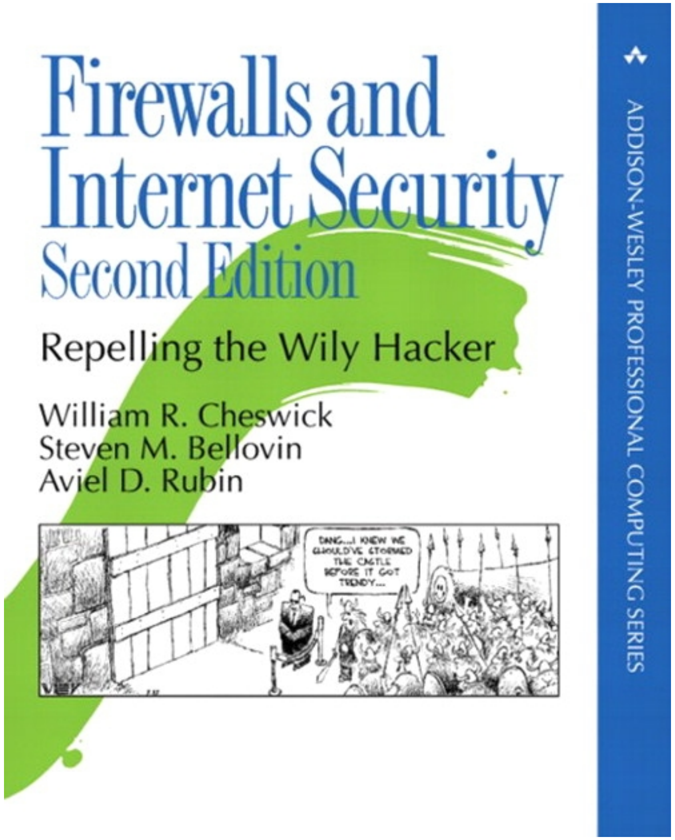
References


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

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

William R Cheswick

Steven M Bellovin

Aviel D Rubin

Host and network intrusion detection

- Intrusion prevention
    - Network firewall
        - Restrict flow of packets
    - System security
        - Find buffer overflow vulnerabilities and remove them!
- Intrusion detection
    - Discover system modifications
        - Tripwire
    - Look for attack in progress
        - Network traffic patterns
        - System calls, other system events

Tripwire

- Outline of standard attack
  - Gain user access to system
  - Gain root access
  - Replace system binaries to set up backdoor
  - Use backdoor for future activities
- Tripwire detection point: system binaries
  - Compute hash of key system binaries
  - Compare current hash to hash stored earlier
  - Report problem if hash is different
  - Store reference hash codes on read-only medium

Is Tripwire too late?

- Typical attack on server
  - Gain access
  - Install backdoor
    - This can be in memory, not on disk!!
  - Use it
- Tripwire
  - Is a good idea
  - Wont catch attacks that don’t change system files
  - Detects a compromise that *has happened*

Remember: Defense in depth

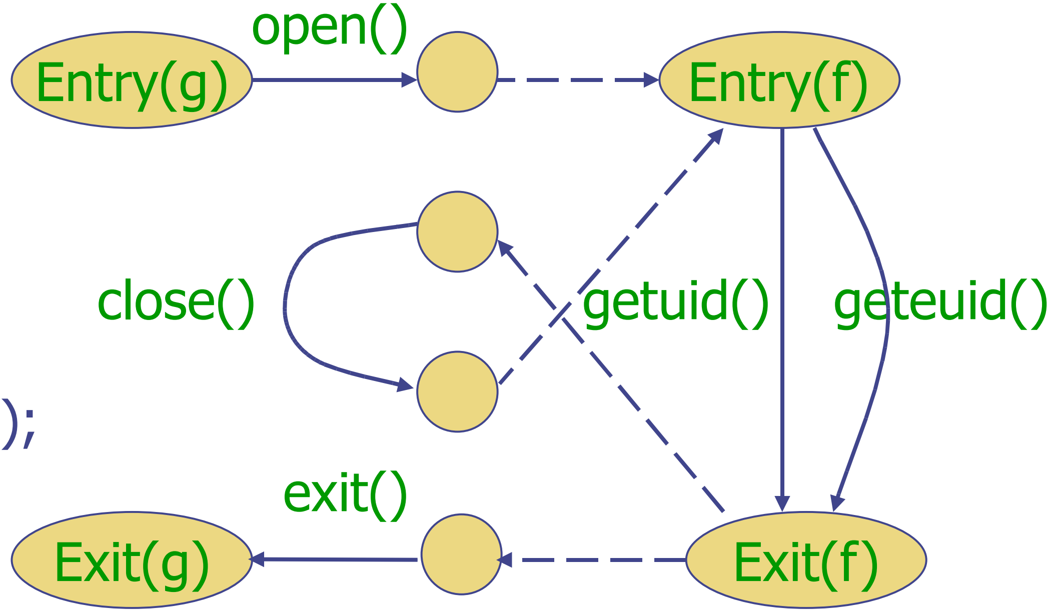
Detect modified binary in memory?

- Can use system-call monitoring techniques
- For example [Wagner, Dean IEEE S&P '01]
  - Build automaton of expected system calls
    - Can be done automatically from source code
  - Monitor system calls from each program
  - Catch violation

Results so far: lots better than not using source code!

Example code and automaton

```
f(int x) {
   x ? getuid() : geteuid();
   x++
}
g() {
   fd = open("foo", O_RDONLY);
   f(0); close(fd); f(1);
   exit(0);
}
```



If code behavior is inconsistent with automaton, something is wrong

General intrusion detection

http://www.snort.org/

- Many intrusion detection systems
  - Close to 100 systems with current web pages
  - Network-based, host-based, or combination
- Two basic models
  - Misuse detection model
    - Maintain data on known attacks
    - Look for activity with corresponding signatures
  - Anomaly detection model
    - Try to figure out what is “normal”
    - Report anomalous behavior
- Fundamental problem: too many false alarms

Misuse example - rootkit

- Rootkit sniffs network for passwords
  - Collection of programs that allow attacker to install and operate a packet sniffer (on Unix machines)
  - Emerged in 1994, has evolved since then
  - 1994 estimate: 100,000 systems compromised
- Rootkit attack
  - Use stolen password or dictionary attack to get user access
  - Get root access using vulnerabilities in rdist, sendmail, /bin/mail, loadmodule, rpc.ypupdated, lpr, or passwd
  - Ftp Rootkit to the host, unpack, compile, and install it
  - Collect more username/password pairs and move on

Rootkit covers its tracks

- Modifies netstat, ps, ls, du, ifconfig, login
  - Modified binaries hide new files used by rootkit
  - Modified login allows attacker to return for passwords
- Rootkit fools simple Tripwire checksum
  - Modified binaries have same checksum
  - But a better hash would be able to detect rootkit

Detecting rootkit on system

- Sad way to find out
  - Disk is full of sniffer logs
- Manual confirmation
  - Reinstall clean ps and see what processes are running
- Automatic detection
  - Rootkit does not alter the data structures normally used by netstat, ps, ls, du, ifconfig
  - Host-based intrusion detection can find rootkit files
    - As long as an update version of Rootkit does not disable your intrusion detection system …

Detecting network attack (Sept 2003)

- Symantec honeypot running Red Hat Linux 9
- Attack
  - Samba ‘call_trans2open’ Remote Buffer Overflow (BID 7294)
  - Attacker installed a copy of the SHV4 Rootkit
- Snort NIDS generated alerts, from this signature

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 139 \
(msg:"NETBIOS SMB trans2open buffer overflow attempt"; \
flow:to_server,established; \
content:"|00|"; offset:0; depth:1; \
content:"|ff|SMB|32|"; offset:4; depth:5;
content:"|00 14|"; offset:60; depth:2; \
…
```

More info: https://tms.symantec.com/members/
AnalystReports/030929-Analysis-SHV4Rootkit.pdf

Misuse example - port sweep

- Attacks can be OS specific
  - Bugs in specific implementations
  - Oversights in default configuration
- Attacker sweeps net to find vulnerabilities
  - Port sweep tries many ports on many IP addresses
  - If characteristic behavior detected, mount attack
    - SGI IRIX responds TCPMUX port (TCP port 1)
    - If machine responds, SGI IRIX vulnerabilities can be tested and used to break in
- Port sweep activity can be detected

Anomaly Detection

- Basic idea
  - Monitor network traffic, system calls
  - Compute statistical properties
  - Report errors if statistics outside established range
- Example – IDES (Denning, SRI)
  - For each user, store daily count of certain activities
    - E.g., Fraction of hours spent reading email
  - Maintain list of counts for several days
  - Report anomaly if count is outside weighted norm

Big problem: most unpredictable user is the most important

[Hofmeyr, Somayaji, Forrest]

Anomaly – sys call sequences

- Build traces during normal run of program
  - Example program behavior (sys calls)

    open read write open mmap write fchmod close
  - Sample traces stored in file (4-call sequences)

    open read write open

    read write open mmap

    write open mmap write

    open mmap write fchmod

    mmap write fchmod close
  - Report anomaly if following sequence observed

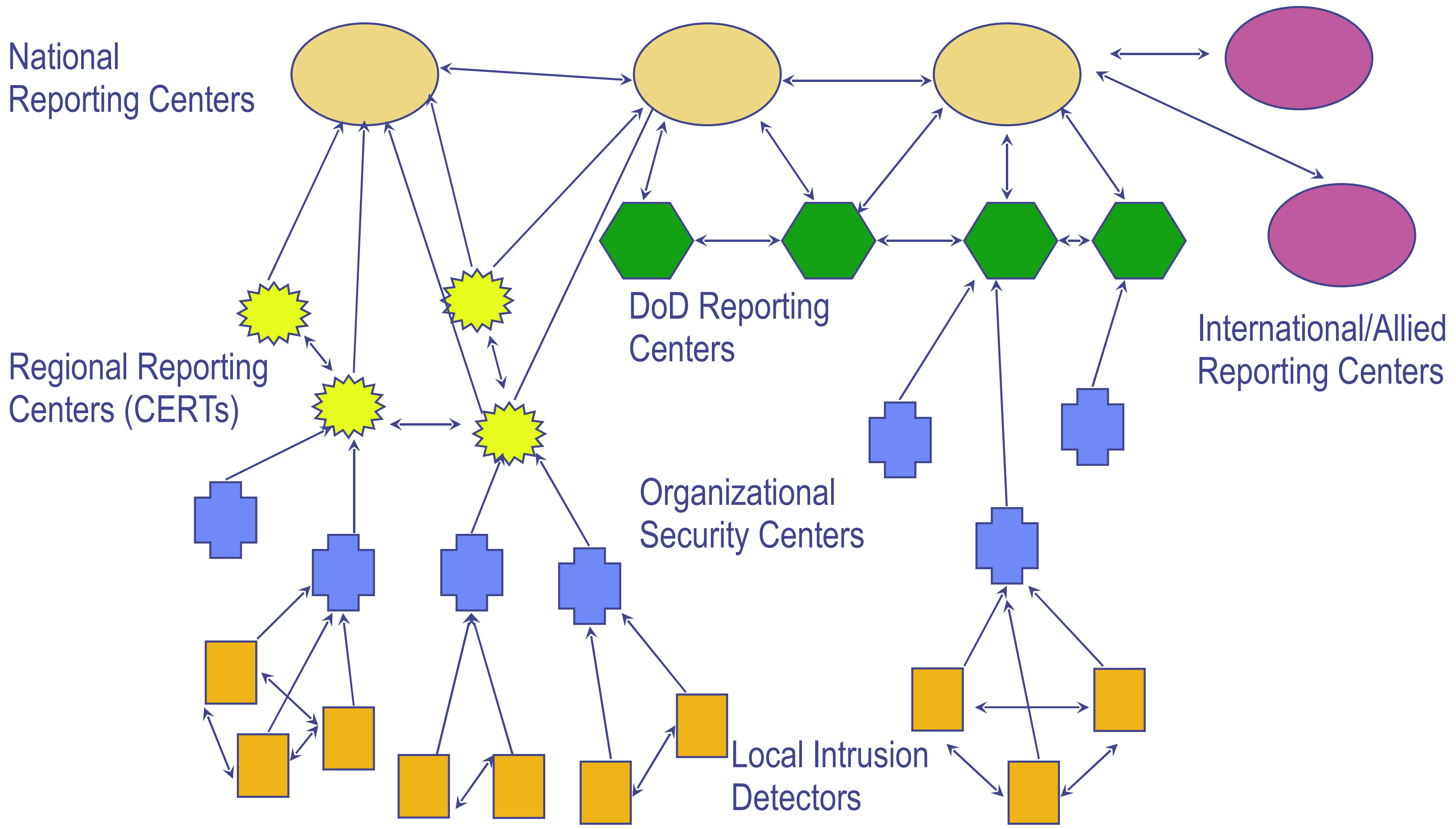
    open read read open mmap write fchmod close

    Compute # of mismatches to get mismatch rate

Difficulties in intrusion detection

- Lack of training data
  - Lots of “normal” network, system call data
  - Little data containing realistic attacks, anomalies
- Data drift
  - Statistical methods detect changes in behavior
  - Attacker can attack gradually and incrementally
- Main characteristics not well understood
  - By many measures, attack may be within bounds of “normal” range of activities
- False identifications are very costly
  - Sys Admin spend many hours examining evidence

Strategic Intrusion Assessment [Lunt]



National
Reporting Centers

Regional Reporting
Centers (CERTs)

DoD Reporting
Centers

International/Allied
Reporting Centers

Organizational
Security Centers

Local Intrusion
Detectors

www.blackhat.com/presentations/bh-usa-99/teresa-lunt/tutorial.ppt

Strategic Intrusion Assessment [Lunt]

- Test over two-week period
  - AFIWC's intrusion detectors at 100 AFBs alarmed on 2 million sessions
  - Manual review identified 12,000 suspicious events
  - Further manual review => four actual incidents
- Conclusion
  - Most alarms are false positives
  - Most true positives are trivial incidents
  - Of the significant incidents, most are isolated attacks to be dealt with locally

Lecture Review

- Firewalls
  - Packet filter (stateless, stateful)
  - Application-layer proxies
- Intrusion detection
  - Anomaly and misuse detection
  - Host and network intrusion detection
- Questions?