Firewalls

Introduction

- seen evolution of information systems
- now everyone want to be on the Internet
- and to interconnect networks
- has persistent security concerns
  - can’t easily secure every system in org
- typically use a **Firewall**
- to provide **perimeter defence**
- as part of comprehensive security strategy

What is a Firewall?

- a **choke point** of control and monitoring
- interconnects networks with differing trust
- imposes restrictions on network services
  - only authorized traffic is allowed
- auditing and controlling access
  - can implement alarms for abnormal behavior
- provide NAT & usage monitoring
- implement VPNs using IPSec
- must be immune to penetration

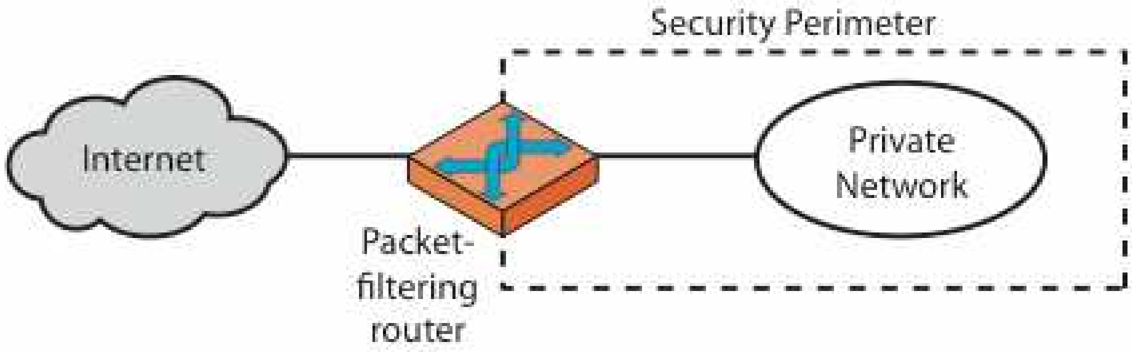
Firewall Limitations

- cannot protect from attacks bypassing it
  - eg sneaker net, utility modems, trusted organisations, trusted services (eg SSL/SSH)
- cannot protect against internal threats
  - eg disgruntled or colluding employees
- cannot protect against transfer of all virus infected programs or files
  - because of huge range of O/S & file types

Firewalls – Packet Filters

- simplest, fastest firewall component
- foundation of any firewall system
- examine each IP packet (no context) and permit or deny according to rules
- hence restrict access to services (ports)
- possible default policies
  - that not expressly permitted is prohibited
  - that not expressly prohibited is permitted

Firewalls – Packet Filters



(a) Packet-filtering router

Screeing policy actions

- Forward
  - The package is forwarded to the intended recipient
- Drop
  - The packages is dropped (without notification)
- Reject
  - The package is rejected (with notification)
- Log
  - The packages appearance is logged (to be combined)
- Alarm
  - The packages appearance triggers an alarm (to be combined)

Screening policies

- There should always be some default rules
  - The last rule should be „Drop everything from everyone‟ which enforce a defensive strategy
  - Network monitoring and control messages should be considered

Firewalls – Packet Filters

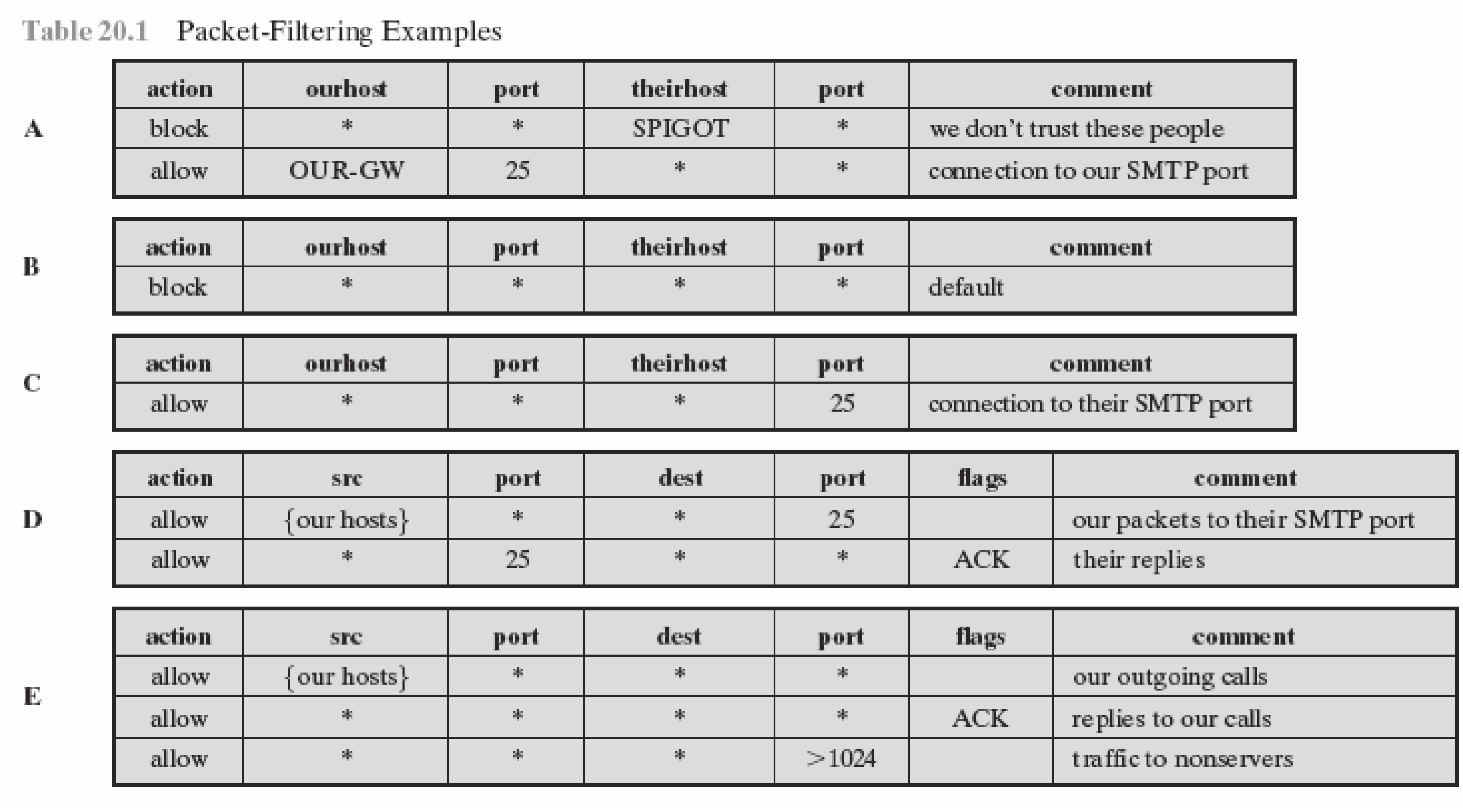


Table 20.1 Packet-Filtering Examples

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

Attacks on Packet Filters

- IP address spoofing
  - fake source address to be trusted
  - add filters on router to block
- source routing attacks
  - attacker sets a route other than default
  - block source routed packets
- tiny fragment attacks
  - split header info over several tiny packets
  - either discard or reassemble before check

Firewalls – Stateful Packet Filters

- traditional packet filters do not examine higher layer context
  - ie matching return packets with outgoing flow
- stateful packet filters address this need
- they examine each IP packet in context
  - keep track of client-server sessions
  - check each packet validly belongs to one
- hence are better able to detect bogus packets out of context

Advantage/Disadvantage

+

- One screening router can protect a whole network
- Packet filtering is extremely efficient
- Packet filtering is widely available

–

- Current filtering tools are not perfect
- Some policies are difficult to enforce
- Packet filtering generates extra load for the router

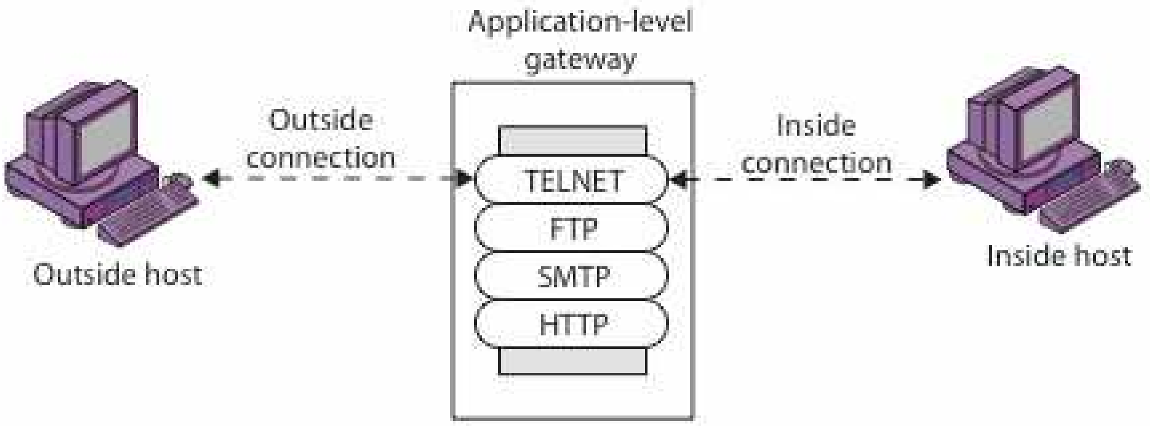
Firewalls - Application Level Gateway (or Proxy)

- have application specific gateway / proxy
- has full access to protocol
  - user requests service from proxy
  - proxy validates request as legal
  - then actions request and returns result to user
  - can log / audit traffic at application level
- need separate proxies for each service
  - some services naturally support proxying
  - others are more problematic

Different modes

- Proxy-aware application software
  - The application software knows how to connect to the proxy and forward the final destination
- Proxy-aware operating system software
  - The operating system checks and eventually modify the IP addresses to use the proxy
- Proxy-aware user procedures
  - The user has to follow some procedures. He tells the client software where to connect and also the proxy the destination address
- Proxy-aware router
  - The client attempts to make connections as usual and the router intercepts and redirects packages to the proxy

Firewalls - Application Level Gateway (or Proxy)

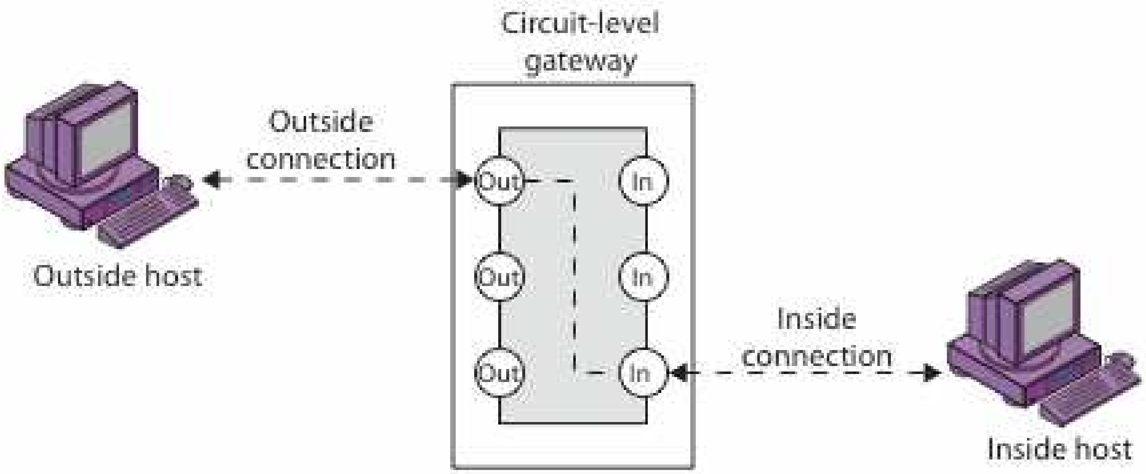


(b) Application-level gateway

Firewalls - Circuit Level Gateway

- relays two TCP connections
- imposes security by limiting which such connections are allowed
- once created usually relays traffic without examining contents
- typically used when trust internal users by allowing general outbound connections
- SOCKS is commonly used

Firewalls - Circuit Level Gateway



(c) Circuit-level gateway

Advantage/Disadvantage

+

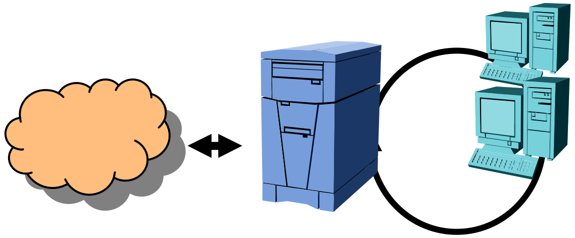
- Proxies can do intelligent filtering
- Proxies can provide logging and caching
- Proxies can provide user-level authentication

-

- Proxies cause a delay
- Proxies can require modifications to clients
- Proxies may require a different server for each service

Network Adress Transalation

- NAT allows to use a set of network addresses internally and a different set externally
- Do not generate security itself but force connection over one point



Modes

- Static allocation
  - The translation scheme is static
- Dynamic allocation of addresses
  - The connection addresses are determined on a per session base
- Dynamic allocation of addresses and ports
  - Both addresses and ports are dynamic

Advantage/Disadvantage

+

- NAT helps to enforce the firewalls control over outbound traffic
- NAT helps to restrict incoming traffic
- NAT hides the internal network configuration

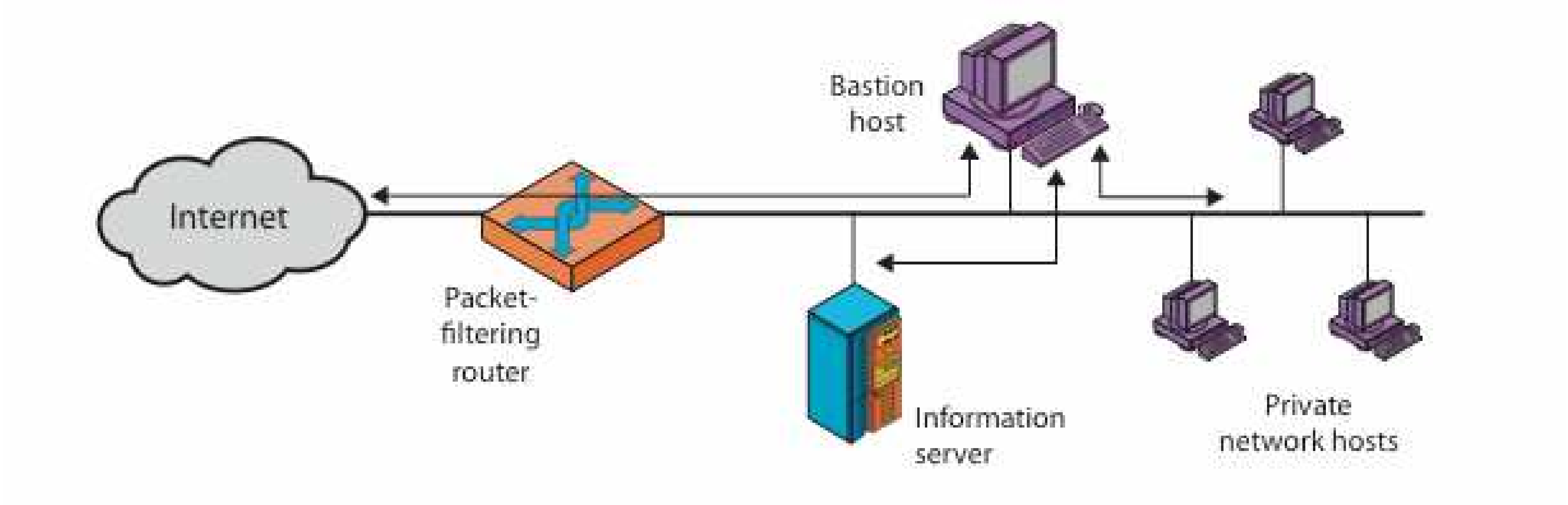
−

- Embedded IP can become a problem
- Dynamic allocation may interfere with encryption and authentication
- Dynamic allocation of port may interfere with package filters

Bastion Host

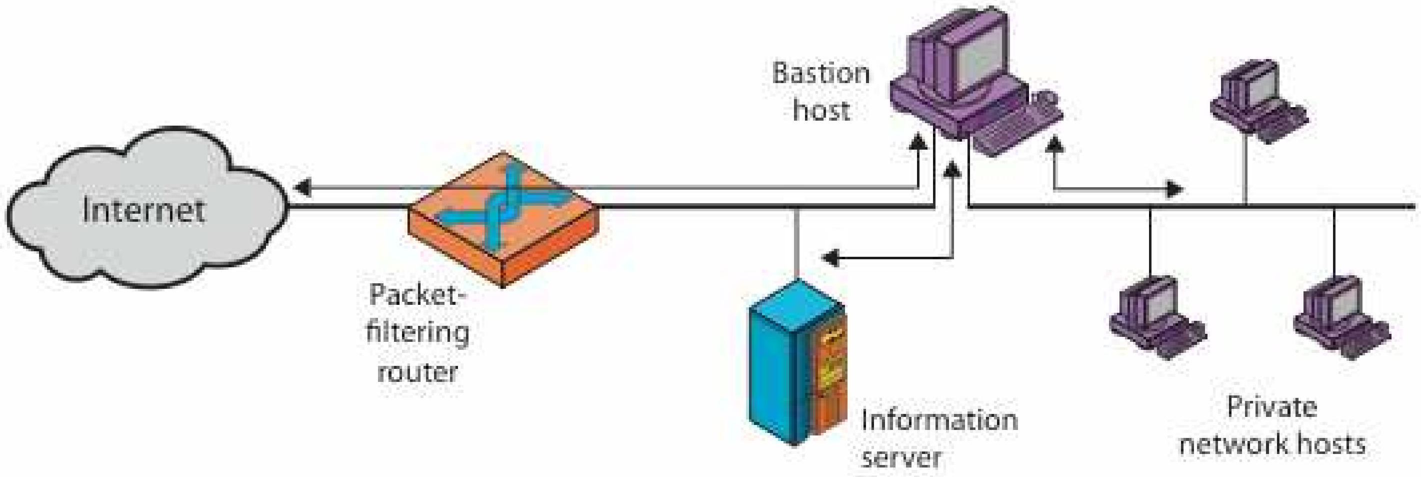
- highly secure host system
- runs circuit / application level gateways
- or provides externally accessible services
- potentially exposed to "hostile" elements
- hence is secured to withstand this
  - hardened O/S, essential services, extra auth
  - proxies small, secure, independent, non-privileged
- may support 2 or more net connections
- may be trusted to enforce policy of trusted separation between these net connections

Firewall Configurations



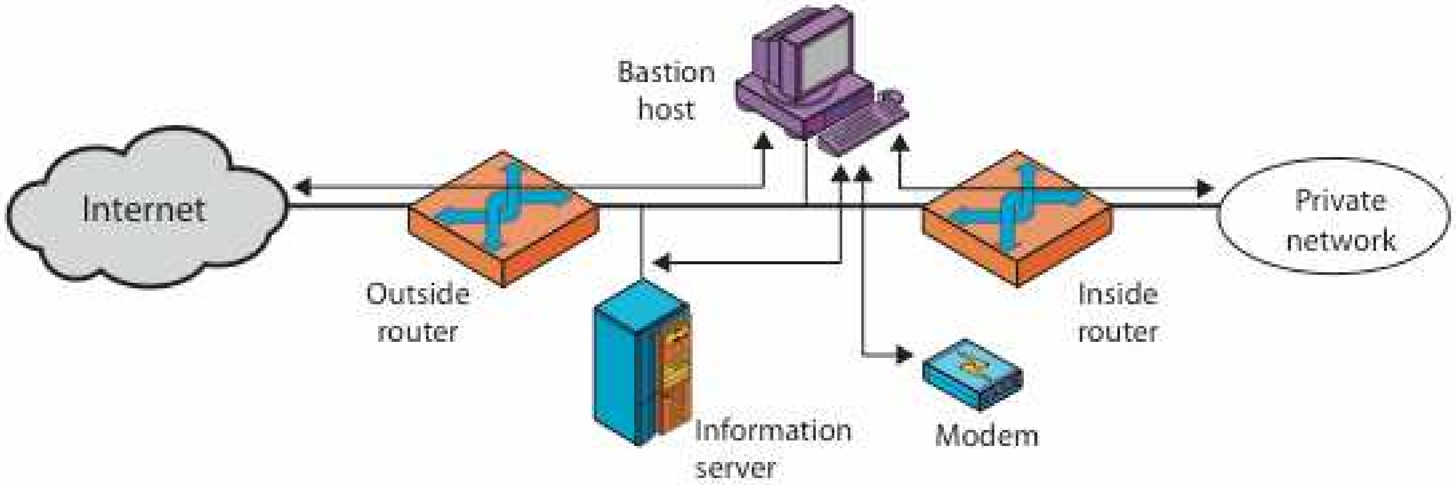
(a) Screened host firewall system (single-homed bastion host)

Firewall Configurations



(b) Screened host firewall system (dual-homed bastion host)

Firewall Configurations

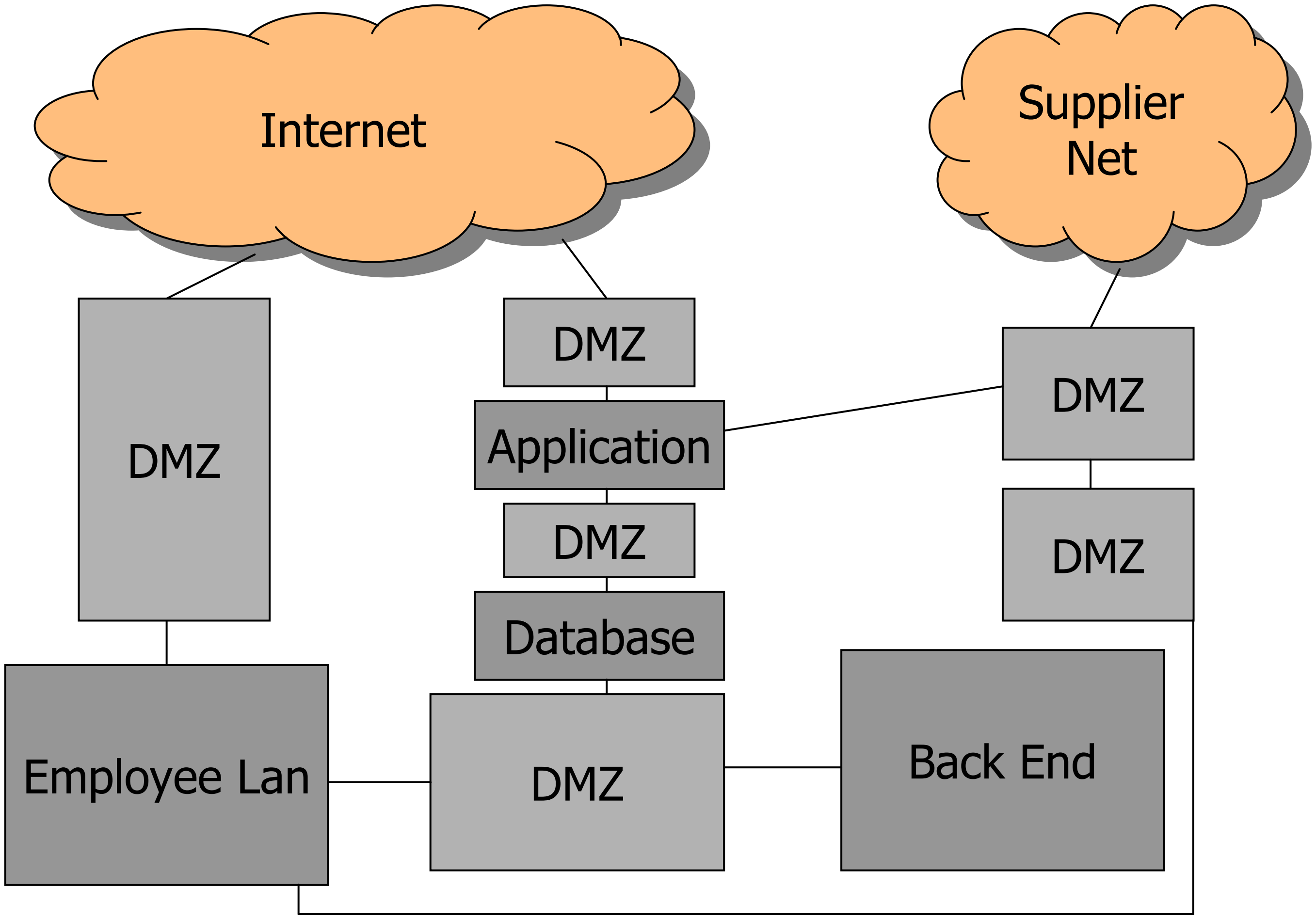


(c) Screened-subnet firewall system

Mulitple Screened Subnets

- Split-Screened subnet
  - Multiple networks between the exterior and interior router. The networks are usually connected by dual-homed hosts.
- Independent Screened Subnets
  - n Screened Subnets

Hybrid - Example Structure

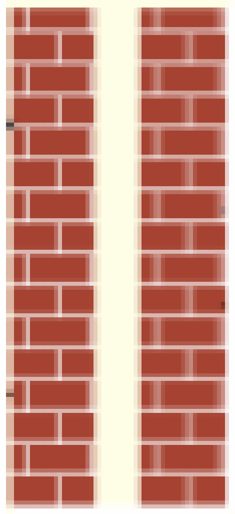


Evaluating a Firewall

- Scalability
- Reliability and Redundancy
- Auditability
- Price (Hardware, Software, Setup, Maintenance)
- Management and Configuration

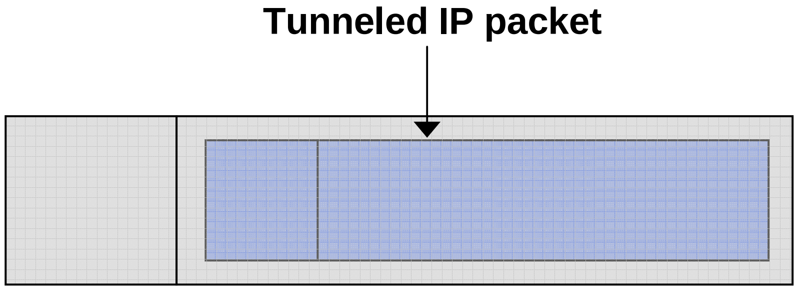
Firewalls and Malware

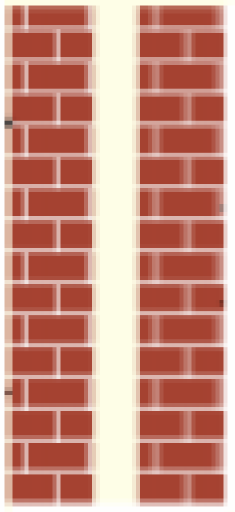
- Should preferably control both ingoing and outgoing traffic
  - Windows XP firewall controls only ingoing traffic
  - Trojans can start up servers on the inside
- Firewall should preferable inspect packets on the application layer
  - Network layer based packet filters do not provide adequate protection



Firewalls and Malware

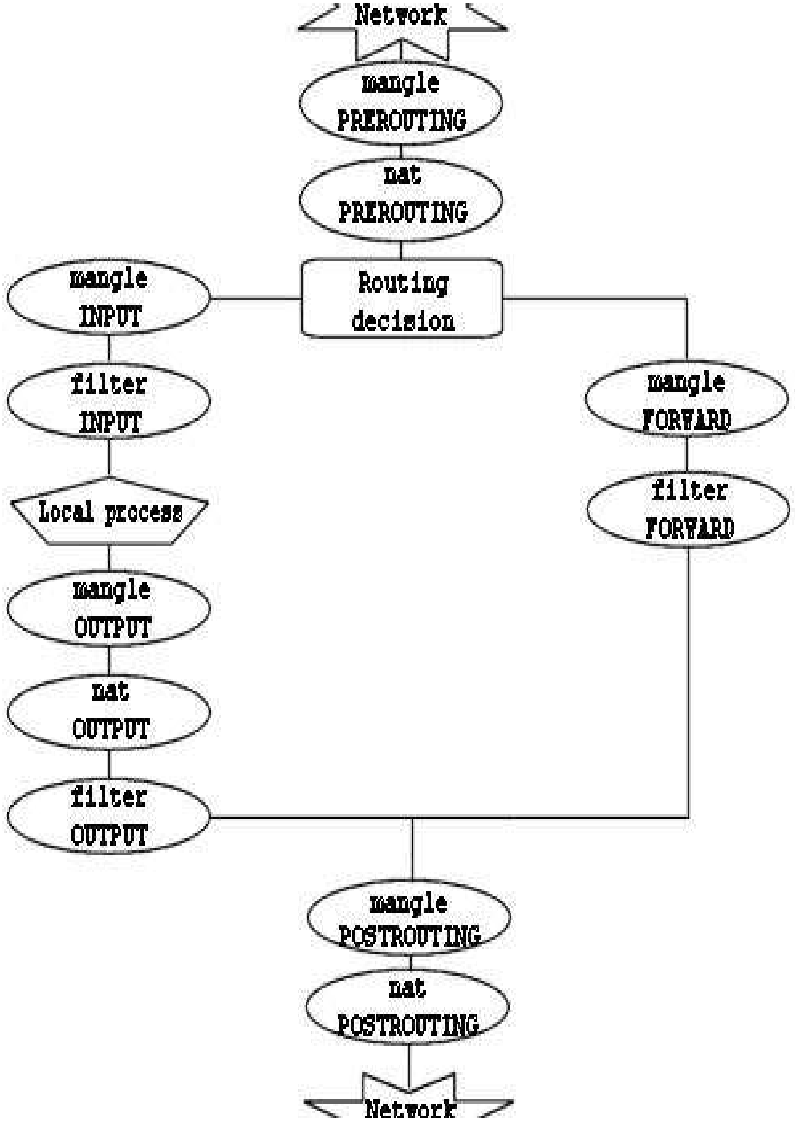
- New worms/viruses often tries to kill firewall and anti virus processes
- “Tunneled Worms”
  - Tunnel IP packet within other IP packet to hide real IP header
  - Tunneling program can be built in in Trojans





IP- Tables

- IP Tables is the standard kernel firewall system for Linux since Kernel 2.4.x
- Packet Filtering and NAT for linux



Rule

iptables [-t table] command [match] [traget/jump]

- -t table
  - Nat (PREROUTING, POSTROUTING)
  - Mangle (PREROUTING, POSTROUTING)
  - Filter (default) (FORWARD, INPUT, OUTPUT)

Rule

iptables [-t table] command [match] [traget/jump]

- Command
  - -P, --policy
  - -A, --append
  - -D, --delete
  - -R, --replace
  - -L, --list
  - ...

Rule

iptables [-t table] command [match] [traget/jump]

- Match (generic)
  - -p, --protocoll (TCP, UDP, ICMP)
  - -s, --source (IP Adresse/port)
  - -d, --destination (IP Adresse/port)
  - -i, --in-interface (eth0, eth1, ppp1)
  - -o, --out-interface (eth0, eth1, ppp1)
  - -m, --match (special commands)

Rule

iptables [-t table] command [match] [traget/jump]

- Target/jump
  - -j ACCEPT
  - -j DROP
  - -j LOG
  - -j MAQUERADE
  - …

Example Rules

- iptable –P FORWARD DROP
  - Introduce the general policy to drop all packages
- Iptable –t nat –P PREROUTING ACCEPT
  - Accept prerouting nat traffic
- iptable –A FORWARD -i eth1 –p TCP –d 193.10.221.184 -–dport 80 –j ACCEPT
  - Accept all tcp connections to port 80 coming in at my second network interface to my ip
- iptables –A FORWARD –m limit -–limit 3/minutes –j LOG
  - Log all refused connections but max. 3 per minute