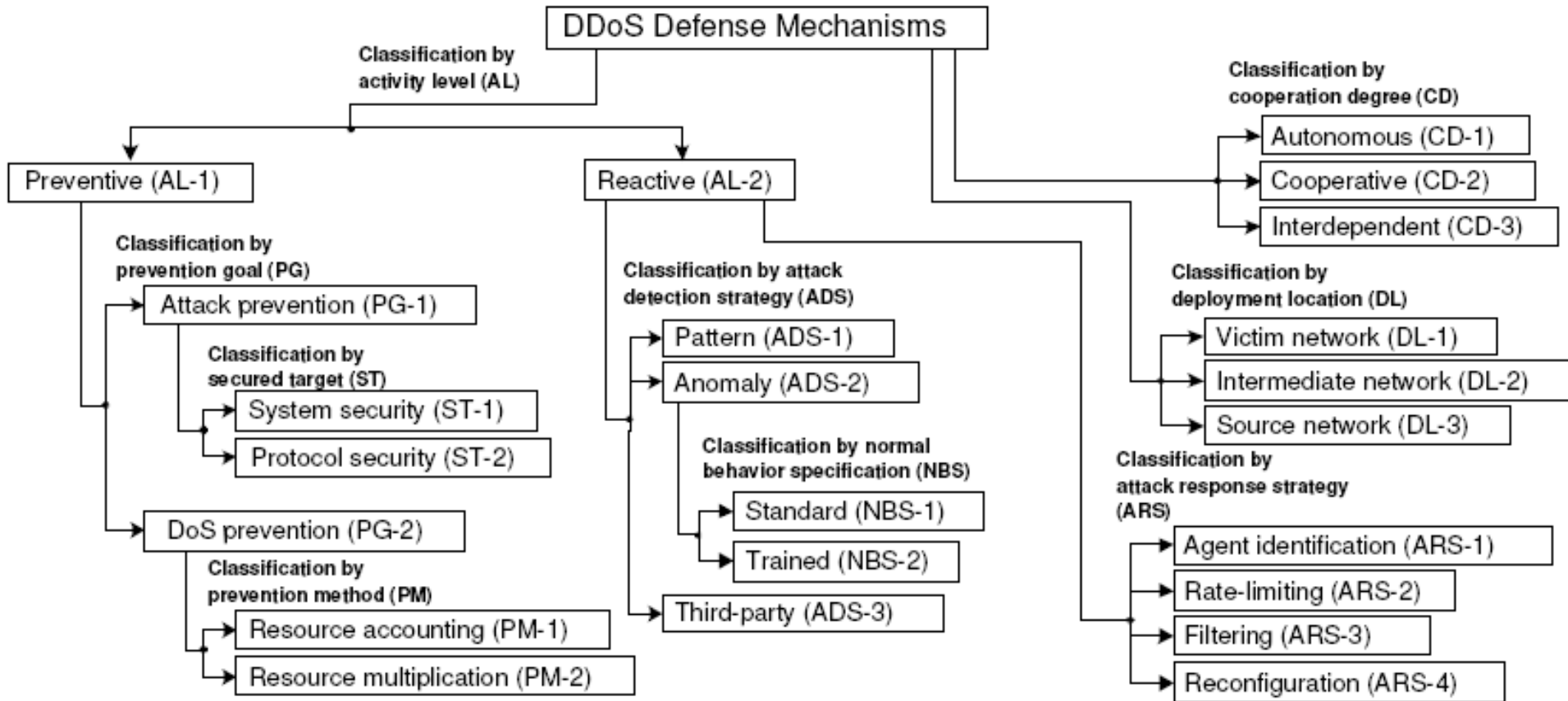


Chapter 16

Attack Mitigation and Countermeasures

- ❑ Defense techniques
 - ❑ TCP SYN flood, ICMP/UDP flood
- ❑ IP address spoofing and traceback
- ❑ Firewalls

Defense Taxonomy



Source: [Mircovic2004]

Challenges



- ❑ DDoS attacks
 - ❑ Need for a distributed response at many points on the Internet
 - Coordinated response is necessary for successful countermeasures
 - ❑ There is no single response strategy against DDOS attacks
 - Combination of different responses needed

- ❑ Economic and social factors
 - ❑ Deployment of response systems at parties that do not suffer direct damage from a DDoS attack

- ❑ Lack of detailed information
 - ❑ Thorough understanding of attacks is required

- ❑ Lack of defense system benchmarks

- ❑ Difficulty of large-scale testing

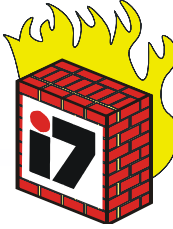
SYN Flood



- ❑ SYN flood attack
 - ❑ TCP SYN flooding constitutes more than 50% of all attacks.
 - ❑ Resource depletion by TCP SYN packets with forged source addresses.
 - ❑ Victim responds by SYN ACK messages that will not be answered.
 - ❑ Allocated resources of the half-open TCP connections at the victim will only be released after time-out.

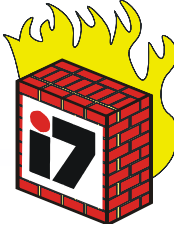
- ❑ Load balancing and replication of resources
 - ❑ The attack will pass unnoticed.
 - ❑ With a sufficient number of attackers the server can still be saturated

ICMP/UDP Flood



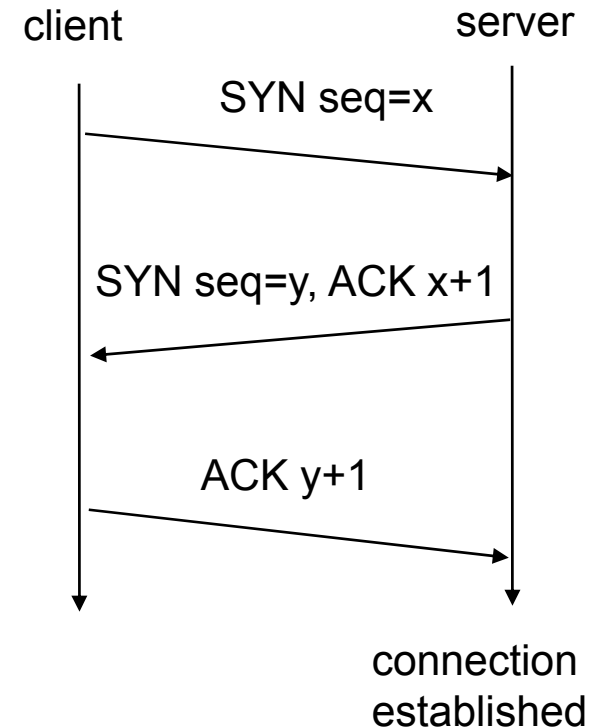
- ❑ ICMP flood
 - ❑ Exploitation of the standard behavior of TCP/IP stacks
 - ❑ Attack type 1
 - ICMP messages need to be processed at the destination
 - Sending huge amounts of ICMP messages blocks computational resources, especially at networking components such as switches and routers
 - Big ICMP messages may crash some TCP/IP stacks
 - ❑ Attack type 2
 - ICMP echo requests need to be answered with an ICMP echo response
 - If sent to a broadcast address, ICMP storms can be initiated
- ❑ UDP flood
 - ❑ Usually targeting small bandwidth connections as UDP is not congestion aware

SYN Flood Protection



Reminder: Regular TCP 3-Way Handshake

- ❑ The client sends a 'TCP SYN' message
 - seq number = x (chosen by the client)
 - ACK flag = 0
 - SYN flag = 1
- ❑ The server allocates a memory for the Transmission Control Block (TCB)
- ❑ The server sends a 'TCP SYN ACK'
 - seq number = y (chosen by the server)
 - ack number = $x + 1$
 - ACK flag = 1
 - SYN flag = 1
- ❑ The client sends a 'CONNECT ACK'
 - seq number = $x + 1$
 - ack number = $y + 1$
 - ACK flag = 1
 - SYN flag = 0
- ❑ The handshake ensures that both sides are ready to transmit data.



SYN Flood Protection



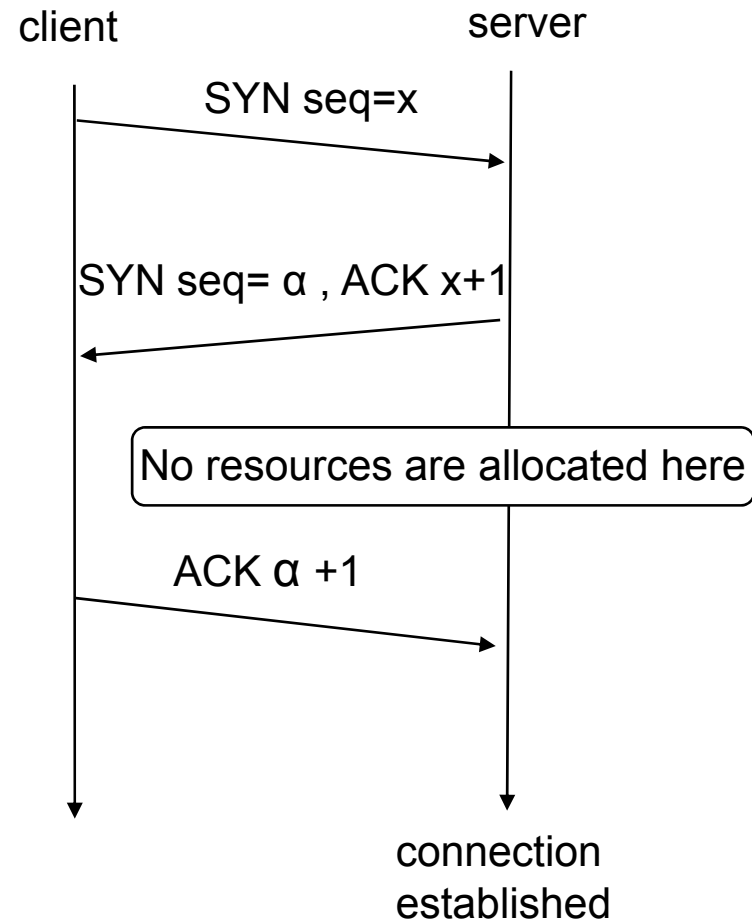
- ❑ TCP stack tweaking
 - ❑ Increase backlog size
 - limited by the memory of the server (each entry ~600 B)
 - ❑ Decrease waiting time for a SYN ACK
 - Even if timeout is short, sufficient number of attack packets still can saturate the server.

- ❑ TCP proxies
 - ❑ TCP connections are intercepted by a TCP proxy.
 - ❑ When the 3-way handshake is complete, the connection is forwarded to the server.
 - ⇒ TCP connections are slower.
 - ⇒ Use only when an attack is assumed.
 - ❑ The server remains safe.
 - ⇒ Only a “fuse”. Does not solve the real problem

SYN Flood Protection: TCP SYN cookies



- ❑ SYN cookies as a reaction to an attack
- ❑ SYN cookies are a particular choice of the initial *seq number*.
- ❑ The server generates the initial sequence number α such as:
 - ❑ $\alpha = h(S_{\text{SYN}}, D_{\text{SYN}}, K)$
 - ❑ S_{SYN} : src addr of the SYN packet
 - ❑ D_{SYN} : addr of the server
 - ❑ K : a secret key
 - ❑ h is a cryptographic hash function.
- ❑ At arrival of the ACK message, the server calculates α again.
- ❑ Then, it verifies if the *ack number* is correct.
- ❑ If yes, it assumes that the client has sent a SYN message recently (considered as normal behavior), and allocates TCB memory.



SYN Flood Protection: TCP SYN cookies



❑ Pros

- ❑ The server does not need to allocate resources after the first SYN packet.
- ❑ The client does not need to be aware that the server is using SYN cookies.
⇒ SYN cookies don't require changes in the specification of the TCP protocol.

❑ Cons

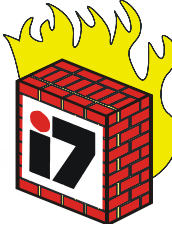
- ❑ Calculating α consumes CPU resources.
⇒ Moving the vulnerability from memory overload to CPU overload.
- ❑ TCP options can not be negotiated properly: Initial Window Size, Maximum Segment Size
⇒ Use only when an attack is assumed.
- ❑ Is vulnerable to crypt analysis: even if h is a secure function the sequence numbers generated by the server can be predicted after receiving/ hijacking a sufficient number of cookies.
⇒ The secret code need to be changed regularly, e.g. by including a timestamp.
- ❑ Note: SYN cookies are available as an optional feature in the Linux Kernel (using the MD5 hash function), and are also available in other operating systems.

Response Strategies: Packet Filtering



- ❑ Packet Filtering - Firewall Rule Manipulation
 - ❑ Attack packets are filtered out and dropped.
 - ❑ Challenge 1
 - How to distinguish between the „good“ packets and the „bad“ packets?
 - ❑ Filterable attacks
 - If the flood packets are not critical for the service offered by the victim, and therefore can be filtered.
 - Example: UDP flood or ICMP request flood on a web server.
 - ❑ Non-filterable attacks
 - The flood packets request legitimate services from the victim.
 - Examples include
 - HTTP request flood targeting a Web server
 - CGI request flood targeting a Web server
 - DNS request flood targeting a name server
 - Filtering all the packets would result in DoS to both attackers and legitimate users.
 - ❑ Packet filtering is not effective if flooding attacks use legitimate services.

Response Strategies: Packet Filtering, Kill Connection



- ❑ Packet Filtering - Firewall Rule Manipulation
 - ❑ Challenge 2
 - Attacker's packet have spoofed source addresses
 - ⇒ The attacker may use an address of a legitimate user.
 - ⇒ Denial-of-Service to legitimate users.

- ❑ Kill Connection
 - ❑ TCP connections can be killed using RST packets that are sent to both connection end points.
 - ❑ The RST packets require correct sequence and acknowledgement numbers, otherwise they are ignored.
 - ❑ Note: an attacker may use a non-conformant TCP/IP stack, which could ignore RST packets.

Response Strategies: Rate-Limiting



□ Rate-Limiting

□ Motivation

- Need of a countermeasure against on-filterable attacks
- Attack packets may violate end-to-end congestion algorithms
- Many attack tools do use a non standard-compliant TCP/IP stack
- In case of false positives the collateral damage is smaller

□ Countermeasure

■ Enforce rate-limits

- Allow a router to control the transmission rate of specific flows and aggregates
- Can be used to control network congestions
- If packets arrive at a higher rate they will be queued or dropped
- If attack flows can be identified they can be rate-limited

□ Rate-limiting mechanisms are deployed when the attack detection has a high number of false positives or cannot precisely characterize the attack stream.

□ Problem: Legitimate users will experience degraded service.

Response Strategies: Rate-Limiting



- ❑ Rate-Limiting:
Aggregate based congestion control (ACC) [Mahajan2001]
 - ❑ Motivation
 - Observing single flows to detect congestion is not sufficient, as congestion may be the result of many low-bandwidth flows.
 - Solution: Aggregate-based Congestion Control (ACC)
 - ❑ Overview
 - Detect the occurrence of a DDoS attack by observing congestion at a router's buffer.
 - Local ACC: identify attack aggregates using a **congestion signature**
 - Act locally to enforce rate-limit on aggregates.
 - Pushback: request upstream routers to help in rate-limiting aggregates.
 - Reviewing: revisit rate-limits periodically.

Response Strategies: Rate-Limiting

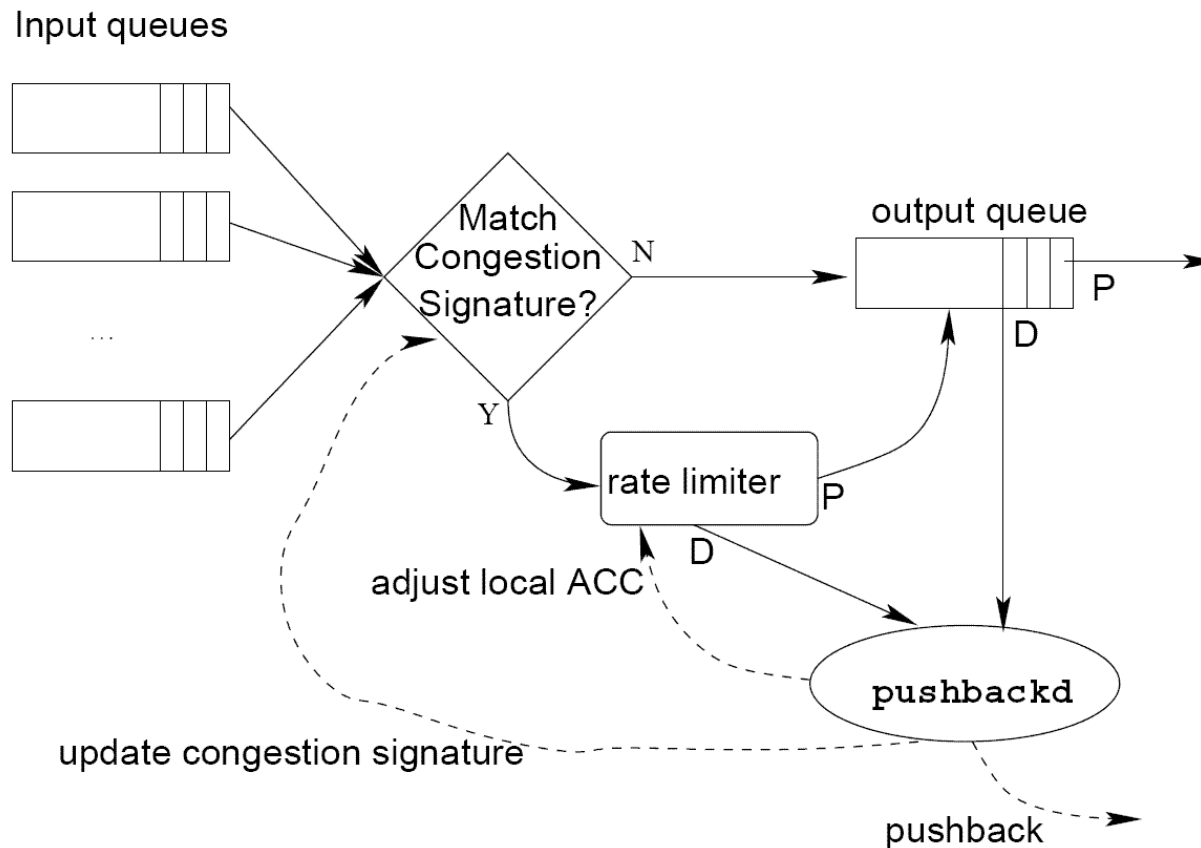


- ❑ Aggregate based congestion control (ACC)
 - ❑ An **aggregate** is a collection of packets from one or more flows that have some properties in common.
 - Examples:
 - Packets to destination D
 - TCP SYN packets
 - IP packets with a bad checksum
 - ICMP ECHO packets
 - HTTP traffic to a specific host
 - ❑ A **congestion signature** is the common property of the congestion flows.
 - ❑ The **congestion signature** is determined using a **drop set** which is a sample of the dropped packets.
 - ❑ The size of the drop set should be large enough to allow meaningful results but also small enough to be able to react quickly.

Response Strategies: Rate-Limiting



- ❑ Aggregate based congestion control (ACC)
 - ❑ Partial view of a router:

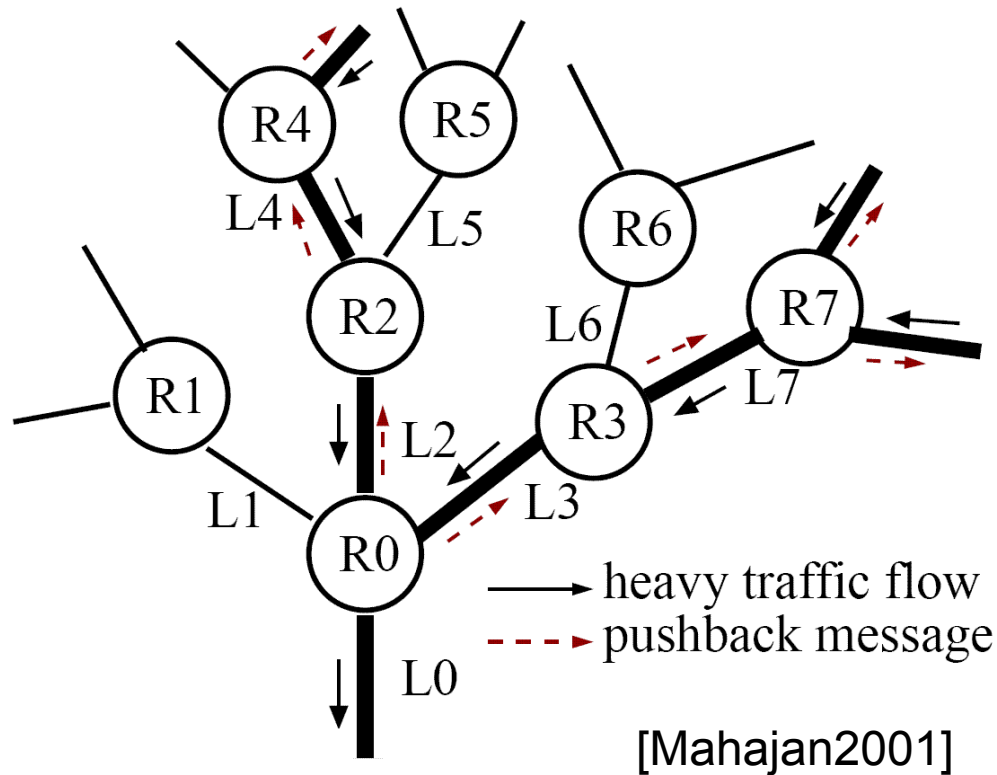


[Ioannidis2002]

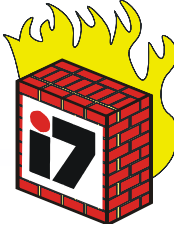
Response Strategies: Rate-Limiting



- Propagating rate-limit requests

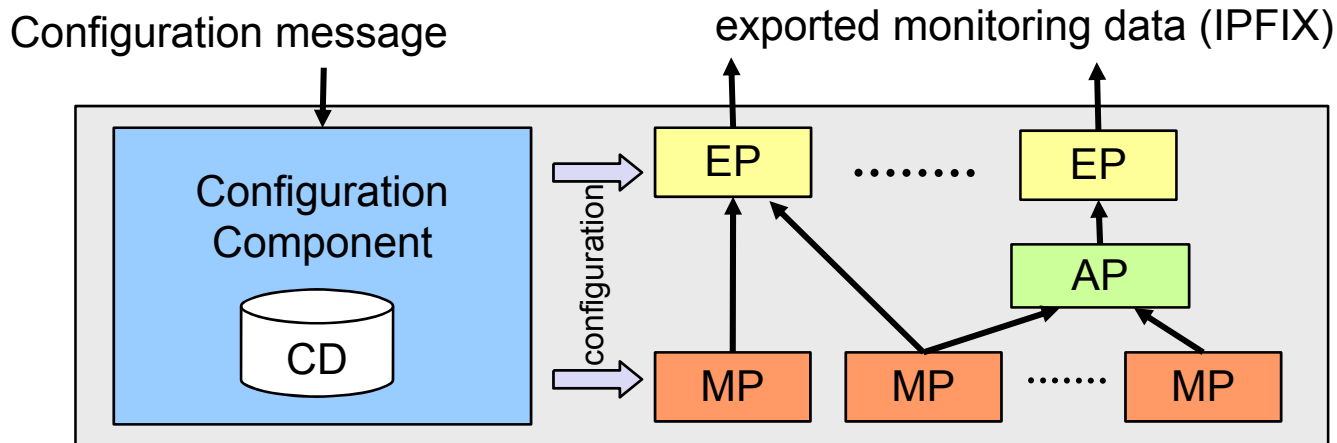


Response Strategies: Tracking



□ Tracking

- IP Traceback
- Re-configuration of the monitoring environment
 - Re-configuration of network monitoring probes
 - Standards used for monitoring: IPFIX, PSAMP
 - Configuration protocols: e.g. NETCONF, NSIS (work in progress)



CD: Configuration Datastore EP: Exporting Process
AP: Aggregation Process
MP: Metering Process (or Sampling Process)

Response Strategies: Redirection



❑ Redirection

- ❑ Based on the modification of routing tables
- ❑ **Black hole routing**
 - packets are sent to a null IP address.
 - Problem: all traffic to a victim including legitimate traffic is dropped.
- ❑ **Sink hole routing**
 - Traffic is sent to an IP address where it is logged for examination to determine the kind of attack.
- ❑ **Shunting**
 - Traffic is redirected to an analysis location within the operator network.
 - Distinguish between suspicious traffic and legitimate traffic.
 - Suspicious traffic is dropped or rate-limited.
 - Legitimate looking traffic is re-inserted into the network using MPLS or GRE tunnels to avoid routing loops.

Honeypots

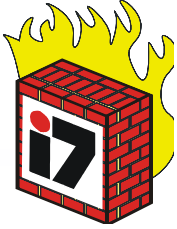


- ❑ A **Honeypot** is a resource which pretends to be a real target, but is an isolated resource where the attacker can not do any real damage.
- ❑ Motivation
 - ❑ **Get to know the “enemy”!!**
 - ❑ Steer the attention of attackers towards irrelevant targets
- ❑ **Low-Interaction** Honeypots:
 - ❑ Emulated services (e.g. FTP) and emulated operations systems
 - ❑ Easier to deploy and maintain
 - ❑ Can log only limited information
 - ❑ Limited capture of activities
- ❑ **High-Interaction** Honeypots
 - ❑ Involves real operation systems and real applications
 - ❑ Can capture extensive amount of information
 - ❑ Problem: Attackers can use the real operating system to attack non-honeypot systems.

Honeypots



- ❑ Honeypots can **capture unknown attacks**.
- ❑ Honeypots can **slow down the spreading** of worms.
 - ❑ Worms scan for vulnerabilities, and take over vulnerable systems.
 - ❑ A honeypot can slow the scanning capabilities of the worm.
 - scan unused IP spaces
- ❑ Production systems frequently cannot be taken offline for analysis.
 - ❑ Taking production systems offline may cause significant damage.
 - ❑ Such systems may be very complex, making it difficult to determine what an attacker actually did.
- ❑ Honeypots can quickly and easily be taken offline for a full forensic analysis.
- ❑ They **provide in-depth knowledge** about the behavior of attackers.



❑ **The Spoofing Problem**

- ❑ Packet routing in IP networks is based on destination address information only, correctness of source address is not verified
- ❑ Most (D)DoS attacks consist of packets with spoofed or faked source addresses in order to disguise the identity of the attacking systems
- ❑ Identification of the attacking systems is needed for installing efficient defense mechanisms
- ❑ Some detection mechanisms also require valid information about the attack sources
- ❑ Further issues: legal prosecution of attackers and prevention of new attacks

Anti-Spoofing Mechanisms



❑ Filtering of forged packets

- ❑ Ingress / egress filtering
- ❑ Unicast reverse path forwarding (uRPF)
- ❑ Source address validity enforcement protocol (SAVE)

❑ IPSec

- ❑ Address authentication with cryptographic hash functions and a secret key
- ❑ Problems:
 - IPSec requires key exchange
 - authentication is CPU power consuming
 - ➔ false authentication may cause DoS

Ingress/ Egress Filtering



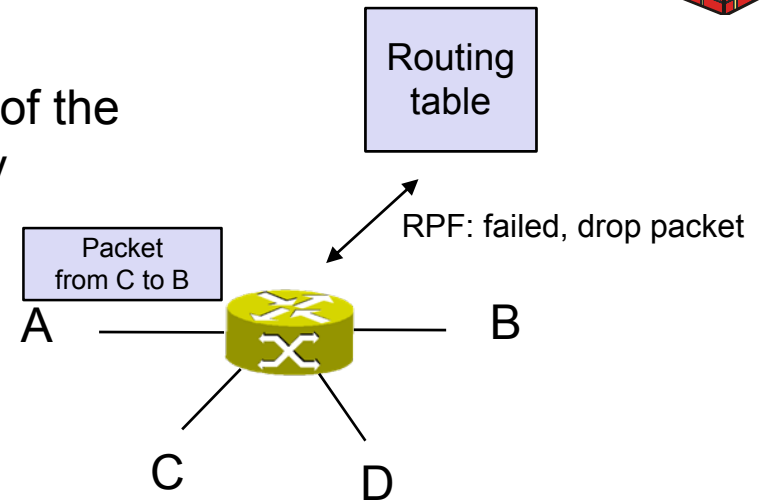
- ❑ To reduce the address space that can be used by the attacker by filtering the packets at the edge of the network
- ❑ Ingress filtering:
 - ❑ Incoming packets with a source address belonging to the network are blocked
 - ❑ Incoming packets from the public Internet with a private source address are blocked
- ❑ Egress filtering:
 - ❑ Outgoing packets that carry a source IP address that does not belong to the network are blocked
- ❑ Problems with ingress/egress filtering:
 - ❑ Requires a lot of **management**
 - ❑ **Profit** is mostly at the victim side
 - ➔ ISPs do not want to spend effort for egress filtering
 - ❑ **Decreases performance**
 - ❑ Some **protocols require a foreign address**, e.g. Mobile IP
 - ❑ A large number of users are directly connected to the Internet
 - ➔ no egress filtering is possible

Unicast Reverse Path Forwarding (uRPF)



❑ Idea:

- ❑ Use the Forward Information Base (FIB) of the router to build filtering rules automatically



❑ Various modes:

❑ Strict mode:

- check that the receiving interface is the shorter to the source
- does not support *asymmetric routes* or *multi-homing*

❑ Loose mode:

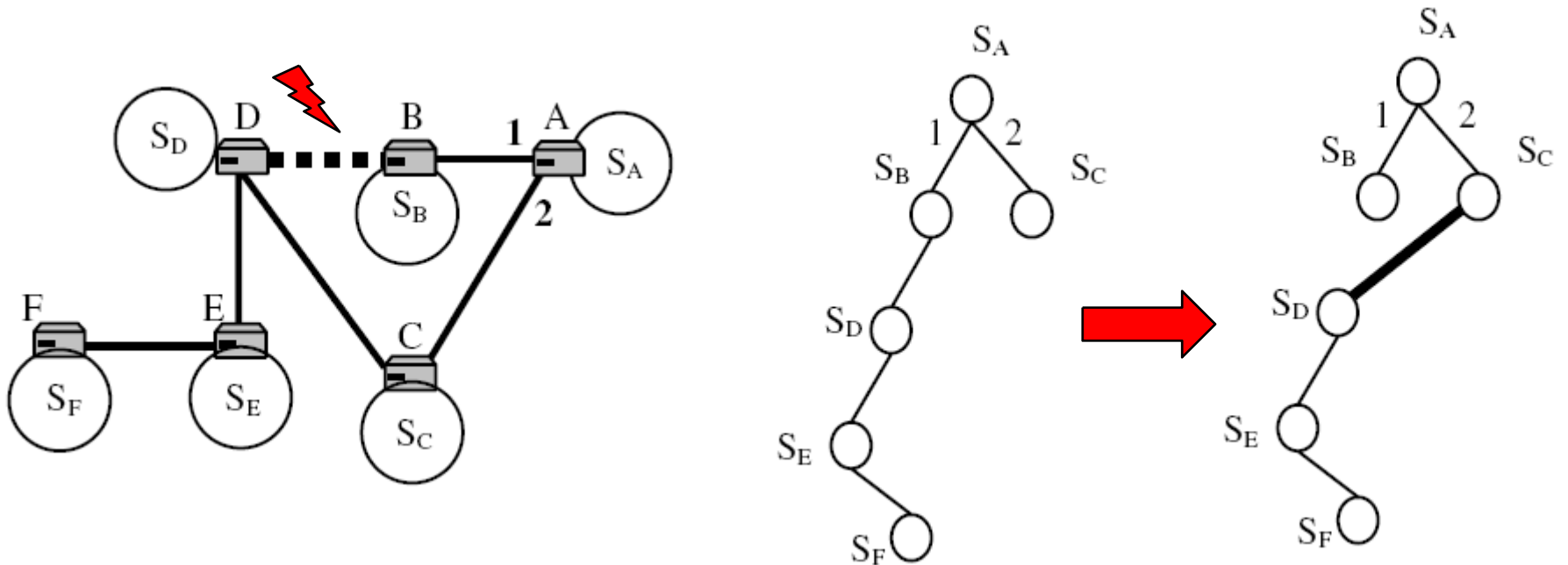
- check that the receiving interface knows a path to the source
- less secure (source just needs to be routable)
- useless if there is a default route in the routing table

- ❑ Available in several commercial routers, implementation varies

Anti-Spoofing Mechanisms



- ❑ **Source address validity enforcement protocol (SAVE) [Li2002]:**
 - ❑ Routers send information about network addresses of directly connected networks to all destinations in forwarding table
 - ❑ Routers maintain *incoming tree* indicating valid source addresses for each interface
 - ❑ Example: route change



Traceback



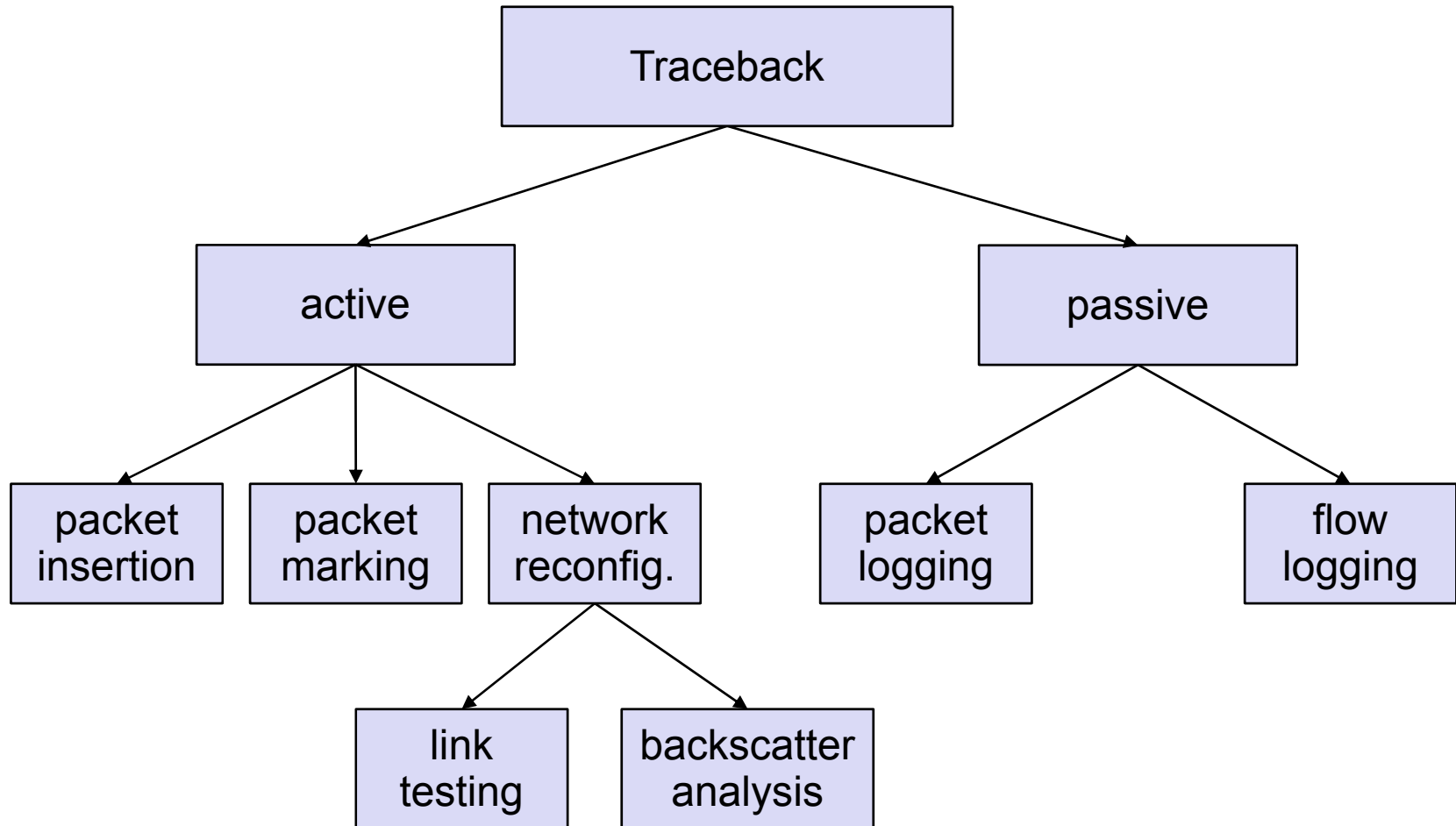
- ❑ Goal:
 - ❑ Identify the source address (or at least the ingress point) and the attack path of a packet without relying on the source address information

- ❑ Challenges:
 - ❑ Short path reconstruction time
 - ❑ Processing and storage requirements
 - ❑ Scalability
 - ❑ Compatibility with existing protocols

Traceback



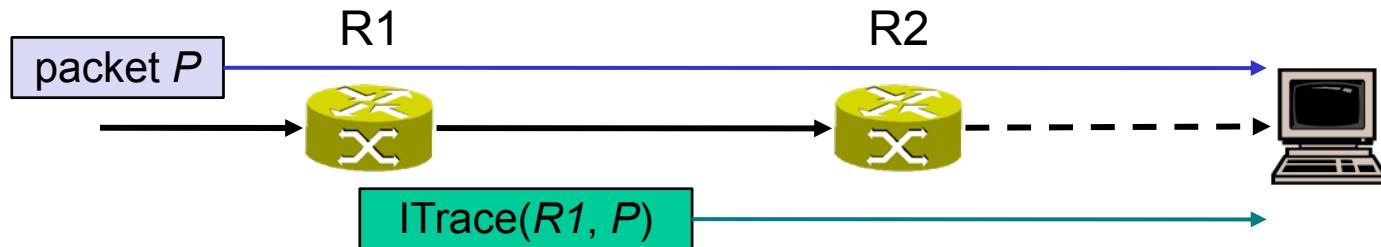
□ Taxonomy of traceback mechanisms



Packet Insertion



- ❑ ICMP traceback (ITrace) [Bellovin2000]:
 - ❑ For 1 out of 20.000 packets, routers send an ITrace message with router ID and information about original packet to the same destination



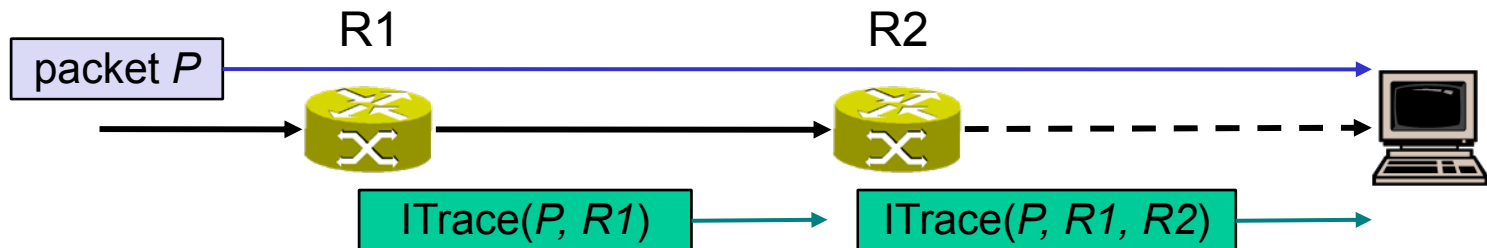
- ❑ If a flow contains enough packets, the destination is likely to receive ITrace messages from every router on the path.
- ❑ Limitations:
 - ❑ Router infrastructure has to be modified
 - ❑ Needs large number of packets/flow \rightarrow long t.b. time for distributed low-rate attacks
 - ❑ Destination has to store original packets for later comparison with ITrace message
 - ❑ ITrace messages need to be authenticated, e.g. using PKI
 - ❑ Inserted ICMP packets may influence network behavior
 - ❑ ICMP traffic is often rate-limited by routers and preferentially dropped during congestion

Packet Insertion



Advanced ICMP traceback:

- ❑ Intention-based ICMP traceback [Mankin2001]
 - ❑ Increase capture probability and number of ITrace messages for packets belonging to potential low-rate attack flows → shorter traceback time
- ❑ Reverse ICMP traceback [Barros2001]
 - ❑ Send additional ITrace messages to the source address
 - in case of a reflector attack, these messages go to the victim
 - enables reconstruction of the path between attacker reflectors
- ❑ ICMP traceback with cumulative path (ITrace-CP) [Lee2003]
 - ❑ ITrace-IP messages sent to next hop router
 - ❑ Next hop routers append their ID if they have also seen the original packet



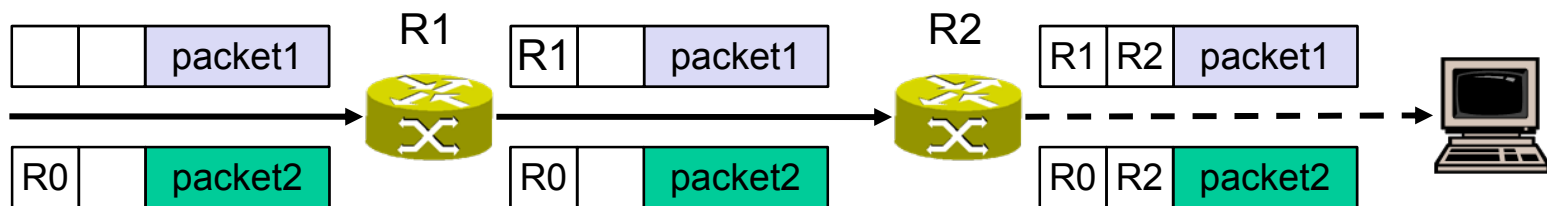
- ❑ Advantage: simpler path reconstruction
- ❑ Disadvantage: routers have to temporally store information about every passing IP packet

Packet Marking



- ❑ Idea:
 - ❑ Use normal (attack) packets to send path information to the destination
 - ❑ Approaches can be distinguished depending on:
 - what path information is provided
 - how the path information is stored and encoded in the packet

- ❑ Router stamping [Doepfner2000]:
 - ❑ Packets carry IP addresses of routers they run through
 - ❑ IP header is extended to carry a fixed number of router IP addresses
 - ❑ Routers randomly decide to store their IP address in one of the reserved slots



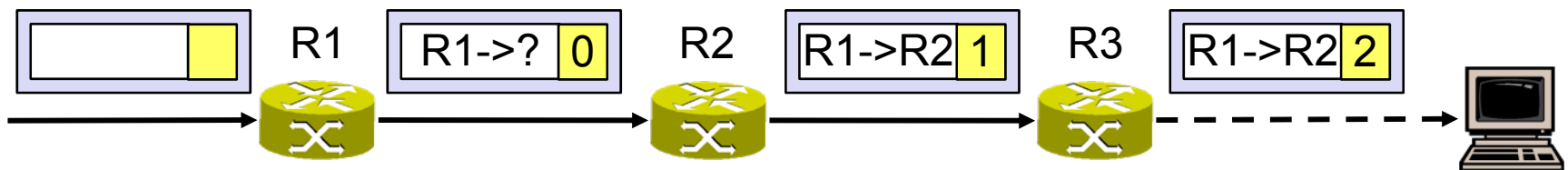
- ❑ Destination is able to deduce the complete path if enough packets are received

Packet Marking



Edge coding [Savage2000]:

- ❑ Packets carry edge information (= start and end address of a link between two subsequent routers plus the distance of that link to the destination)
- ❑ Edge information is compressed, fragmented, and piecewise encoded into the identification field of the IP header
- ❑ Routers randomly decide to store edge information into a passing IP packet
- ❑ Destination is able to deduce the path if enough packets are received



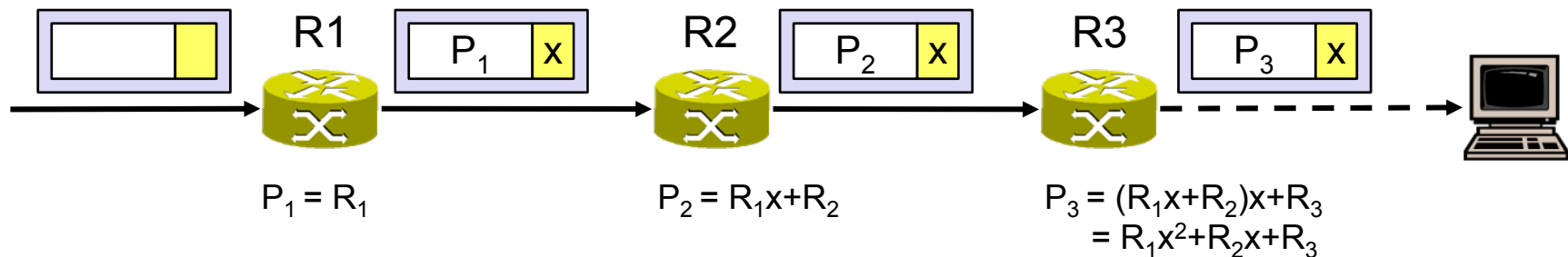
- ❑ Problem:
 - ❑ fragmented edge information is hard to reassemble in case of distributed attack
→ many false positives and high computational complexity

Packet Marking



Improved edge coding:

- ❑ Advanced and authenticated marking [Song2001]:
 - ❑ usage of hash functions instead of fragmentation
→ reduces number of false positives and computational complexity
 - ❑ edge information can be authenticated
 - ❑ disadvantage: destination needs an upstream router map
- ❑ Algebraic coding [Dean2001]:
 - ❑ router IDs = coefficients of a polynomial
 - ❑ packets carry value of polynomial evaluated at different points
 - ❑ destination is able to recover polynomial of degree n if $(n+1)$ unique points have been received



- ❑ Probabilistic path encoding [Adler2002]:
 - ❑ probabilistically encodes path in binary tree using a single bit
 - ❑ does not work with multiple sources

Packet Marking



Limitations:

- ❑ **Router infrastructure** has to be modified
- ❑ Requires **large number of packets** per flow
- ❑ In some schemes, destination has to perform **complex path reconstruction** algorithm
- ❑ **Routers** have to perform **complex computation** to get the right marking
- ❑ Most schemes cannot cope well with **distributed attacks**
- ❑ Most schemes are **vulnerable to fake markings** made by the attackers
- ❑ Drawbacks of marking the IP header:
 - ❑ **additional packet overhead** in case of IP option fields
 - ❑ **incompatible** with IP fragmentation if identification field is used

Link Monitoring



- ❑ Goal:
 - ❑ Test if a given link carries attack traffic directed to the victim
 - ❑ Sequential hop-by-hop link testing starting at the victim until sources or ingress points are found
- ❑ Various approaches:
 - ❑ Using router logging functionality (input debugging, Netflow, ACL,...)
 - ➔ look for packets matching the attack signature, determine the ingress interface, and continue with adjacent router
 - ❑ Temporary interruption of traffic directed to the victim
 - ➔ if link contributes to the attack, attack traffic at the victim stops or declines
 - ❑ Temporary flooding towards the router that is nearest to victim [Burch2000]
 - ➔ if link contributes to the attack, link congestion will cause more attack packets to be dropped before reaching the victim
- ❑ Disadvantages:
 - ❑ **Mechanisms are triggered and controlled manually** by network management
 - ➔ interdomain cooperation is necessary to traceback packets over multiple domains
 - ❑ **No traceback of single packets** since only flow signatures are considered
 - ❑ Interruption and flooding represent a controlled DoS attack themselves

Packet Logging



- ❑ Idea:
 - ❑ Network router log the passage of packets
 - ❑ Two possibilities how to treat the logging information:
 - Routers send periodical reports to a measurement system
 - Routers store the information and may be queried if a given packet has been seen
 - ❑ Problem: reporting/storing all packets requires huge amount of bandwidth/memory
- ❑ Trajectory Sampling [Duffield2001]:
 - ❑ Do not log all packets, instead apply hash-based packet filtering/sampling at routers
 - ❑ All routers select the same packets because they use the same hash function
 - ❑ Information about logged packets is periodically reported to a measurement system
- ❑ SPIE (Source Path Isolation Engine) [Snoeren2001]:
 - ❑ Log and store all packets for later queries
 - ❑ Reduce the required amount of storage by applying hash functions on invariant parts of the packet
 - ❑ Bloom-filters: apply k hash functions on a packet and use n -bit results to mark bits in a $2n$ -sized bit array

Version	Header Length	Type of Service	Total Length	
Identification			DM FF	Fragment Offset
TTL	Protocol		Checksum	
Source Address				
Destination Address				
Options				
Payload				

Packet Logging



- ❑ Limitations:
 - ❑ Low but non-zero false positive rate due to hash collisions
 - ❑ Some packet transforms (fragmentation, IP tunnelling, NAT, IPSec,...) require special treatment
 - ❑ **Router infrastructure** has to be modified
 - ❑ Acceleration **hardware needed** for packet processing in high-speed networks

- ❑ Trajectory Sampling:
 - ❑ **Probabilistic approach** since only selected packets are logged
 - ❑ **Hash function** used for filtering/sampling may bias the result

- ❑ SPIE:
 - ❑ Packet information is only stored for a **short period of time**
 - ❑ **Required memory** still about 0.5% of link capacity

Flow Logging



- ❑ Idea [Lee2004]:
 - ❑ Log packet flows or source-destination pairs instead of single packets in order to reduce required storage
 - ❑ Average flow consists of approximately 7.75 packets
 - ➔ memory can be reduced or storage time can be increased by the same factor

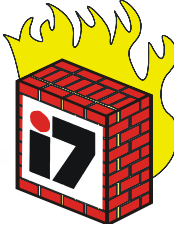
- ❑ Limitations and Drawbacks:
 - ❑ No guarantee that the **same packets** have been observed since only the IP-5-tuple is considered ➔ false positives
 - ❑ No accurate results in case of **multi-path routing**
 - ❑ During DDoS attack with **randomly spoofed source addresses**, number of flows/source-destination pairs is almost equal to the number of packets.
 - ➔ almost no storage gain compared to packet logging

Comparison of Traceback Mechanisms

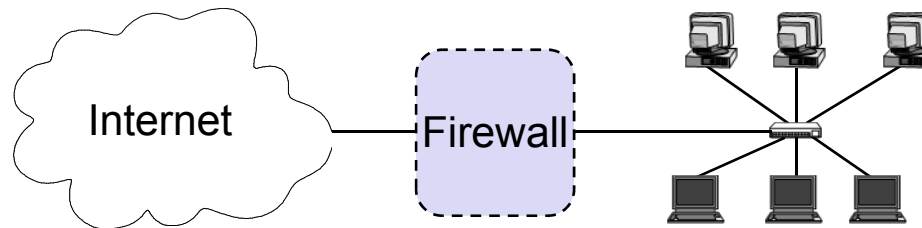


	Packet insertion	Packet marking	Network reconfiguration	Packet logging	Flow logging
Modified routing infrastructure	yes	yes	no	yes	yes
Router overhead	low	low	none	high	medium
Complexity at destination	high	high	low	low	low
Necessary number of attack packets	high	high	low	low	low
Distributed/centralized	distributed	distributed	centralized	both possible	both possible
Packet-based/flow-based	Packet-based	Packet-based	Flow-based	Packet-based	Flow-based

Introduction to Network Firewalls



- ❑ In building construction, a firewall is designed to keep a fire from spreading from one part of the building to another
- ❑ A network firewall, however, can be better compared to a moat of a medieval castle:
 - ❑ It restricts people to entering at one carefully controlled point
 - ❑ It prevents attackers from getting close to other defenses
 - ❑ It restricts people to leaving at one carefully controlled point
- ❑ Usually, a network firewall is installed at a point where the protected subnetwork is connected to a less trusted network:
 - ❑ Example: Connection of a corporate local area network to the Internet



- ❑ So, basically firewalls realize access control on the subnetwork level

Introduction to Network Firewalls



- ❑ What firewalls can do:
 - ❑ A firewall is a focus for security decisions
 - ❑ A firewall can enforce a security policy, i.e. concerning access control
 - ❑ A firewall can log Internet activity efficiently
 - ❑ A firewall can limit exposure to security problems in one part of a network
- ❑ What firewalls can not do:
 - ❑ A firewall can't protect against malicious insiders
 - ❑ A firewall can't protect against connections that don't go through it
 - If, for example, there is a modem pool behind a firewall that provides PPP service to access a subnetwork, the firewall can not provide any protection against malicious traffic from dial-in users
 - ❑ A firewall can't protect against completely new threats
 - ❑ A firewall can't fully protect against viruses
 - ❑ A firewall can't set itself up correctly (\Rightarrow cost of operation)

Two Fundamental Approaches Regarding Firewall Policy



❑ **Default deny strategy**

- ❑ *“Everything that is not explicitly permitted is denied”*
- ❑ Examine the services the users of the protected network need
- ❑ Consider the security implications of these services and how the services can be safely provided
- ❑ Allow only those services that can be safely provided and for which there is a legitimate need
- ❑ Deny any other service

❑ **Default permit strategy**

- ❑ *“Everything that is not explicitly forbidden is permitted”*
- ❑ Permit every service that is not considered dangerous
- ❑ Example:
 - Network file system (NFS) and X-Windows are not permitted across the firewall
 - Incoming telnet connections are only allowed to one specific host

Protocol Fields Important for Firewalls



- ❑ Access Protocol
 - ❑ Network layer protocol: IP, Appletalk, IPX (Novell), DecNet, etc.
 - ❑ Access protocol addresses: Ethernet MAC Address, E.164 Address, etc.

- ❑ Network Protocol (IP)
 - ❑ Source and destination addresses
 - ❑ Flags: especially the indication of an IP fragment
 - ❑ Transport layer protocol: TCP, UDP, ICMP, ...
 - ❑ Options: source routing, ...

- ❑ Transport Protocol (TCP/UDP)
 - ❑ Source and destination port: allow to determine (with a limited degree of confidence) the sending / receiving application, as most Internet services use well-known port numbers
 - ❑ Control information:
 - ACK: this bit is set in every segment but the very first one transmitted in a TCP connection
 - SYN: this bit is only set in the first two segments of a connection
 - RST: if set this bit indicates an ungraceful close of a connection

- ❑ Application Protocol
 - ❑ In some cases a firewall might need to peek into application protocol header fields
 - ❑ However, as this is application-dependent this class will not go into detail...

Firewall Terminology & Building Blocks for Firewalls

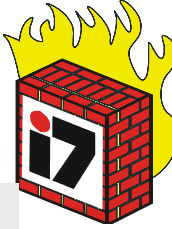


- ❑ *Firewall:*
 - ❑ A component or a set of components that restricts access between a protected network and the Internet or between other sets of networks
- ❑ *Packet Filtering:*
 - ❑ The action a device takes to selectively control the flow of data to and from a network
 - ❑ Packet filtering is an important technique to implement access control on the subnetwork-level for packet oriented networks, e.g. the Internet
 - ❑ A synonym for packet filtering is *screening*
- ❑ *Bastion Host:*
 - ❑ A computer that must be highly secured because it is more vulnerable to attacks than other hosts on a subnetwork
 - ❑ A bastion host in a firewall is usually the main point of contact for user processes of hosts of internal networks with processes of external hosts
- ❑ *Dual homed host:*
 - ❑ A general purpose computer with at least two network interfaces

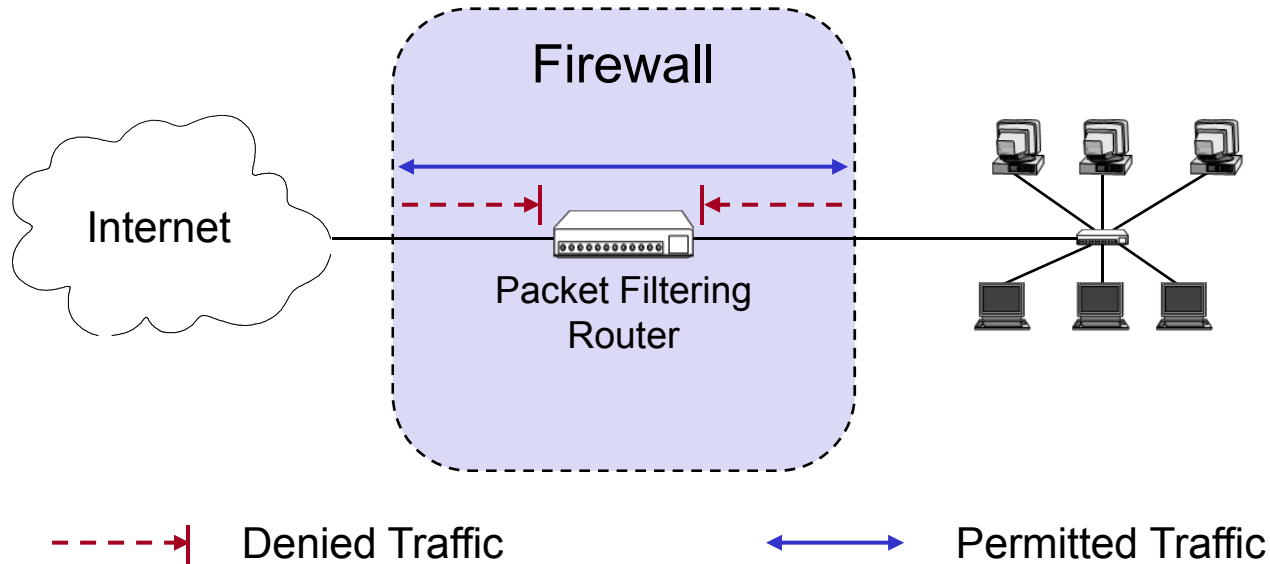
Firewall Terminology & Building Blocks for Firewalls



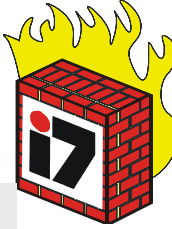
- ❑ *Proxy (Application Layer Gateway):*
 - ❑ A program that deals with external servers on behalf of internal clients
 - ❑ Proxies relay approved client requests to real servers and also relay the servers answers back to the clients
 - ❑ If a proxy interprets and understands the commands of an application protocol it is called an *application level proxy*, if it just passes the PDUs between the client and the server it is called a *circuit level proxy*
- ❑ *Network Address Translation (NAT):*
 - ❑ A procedure by which a router changes data in packets to modify the network addresses
 - ❑ This allows to conceal the internal network addresses (even though NAT is not actually a security technique)
- ❑ *Perimeter Network:*
 - ❑ A subnetwork added between an external and an internal network, in order to provide an additional layer of security
 - ❑ A synonym for perimeter network is *de-militarized zone (DMZ)*



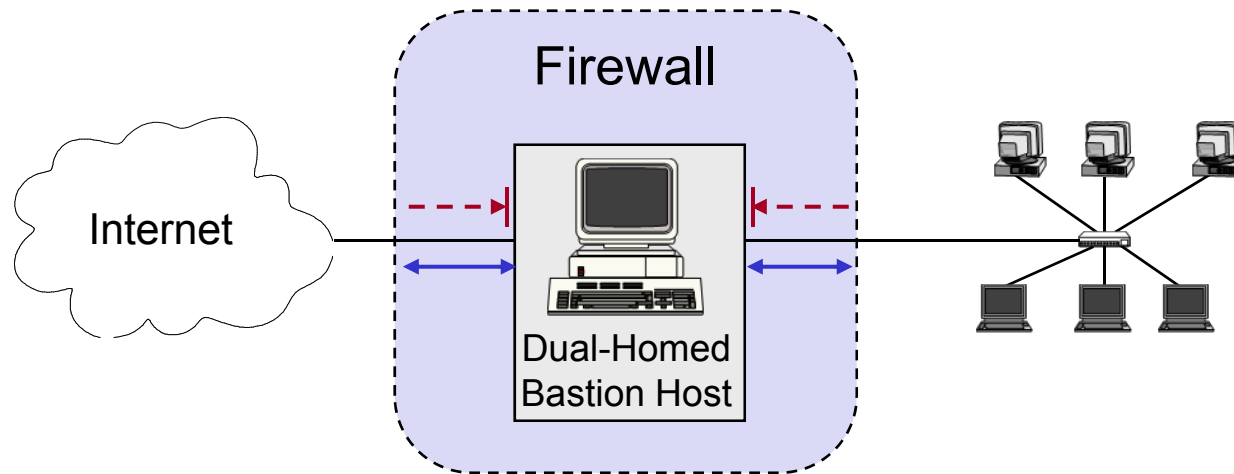
The Simple Packet Filter Architecture



- ❑ The most simple architecture just consists of a packet filtering router
- ❑ It can be either realized with:
 - ❑ A standard workstation (e.g. Linux PC) with at least two network interfaces plus routing and filtering software
 - ❑ A dedicated router device, which usually also offers filtering capabilities



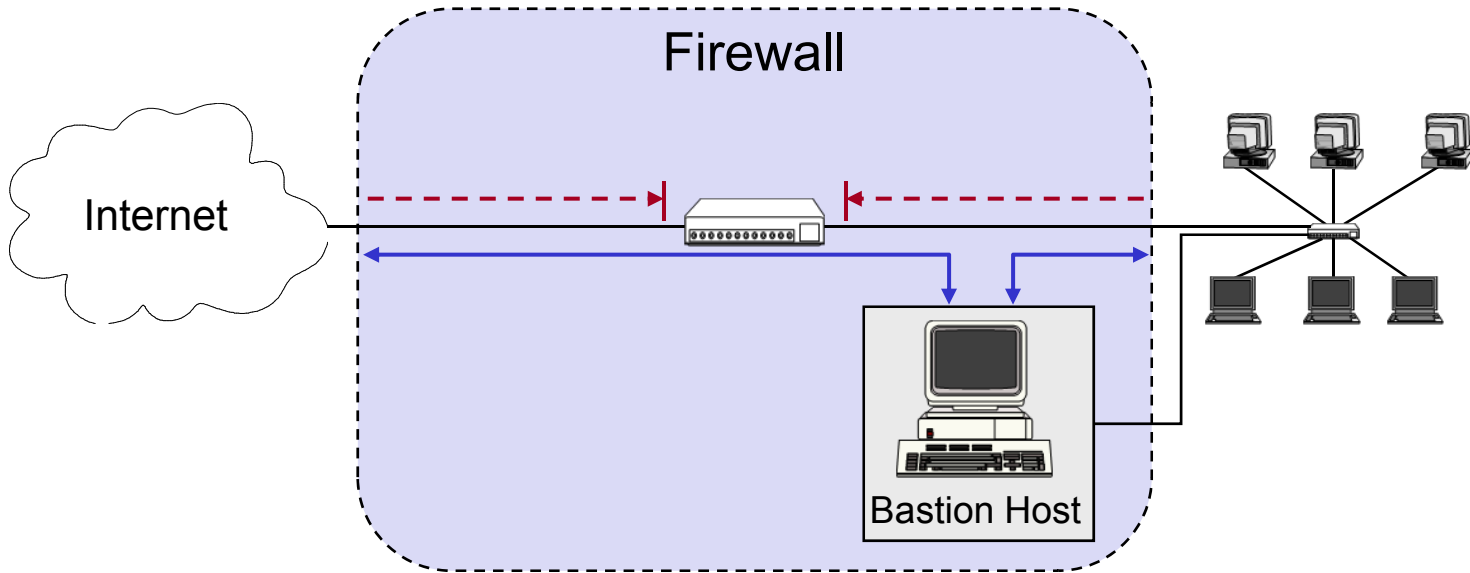
The Dual-Homed Host Architecture



- ❑ The dual-homed host provides:
 - ❑ Proxy services to internal and / or external clients
 - ❑ Eventually packet filtering capabilities if it is also acting as a router
- ❑ Properties of the dual-homed host:
 - ❑ It has at least two network interfaces
- ❑ Drawback: As all permitted traffic passes through the bastion host, this might introduce a performance bottleneck



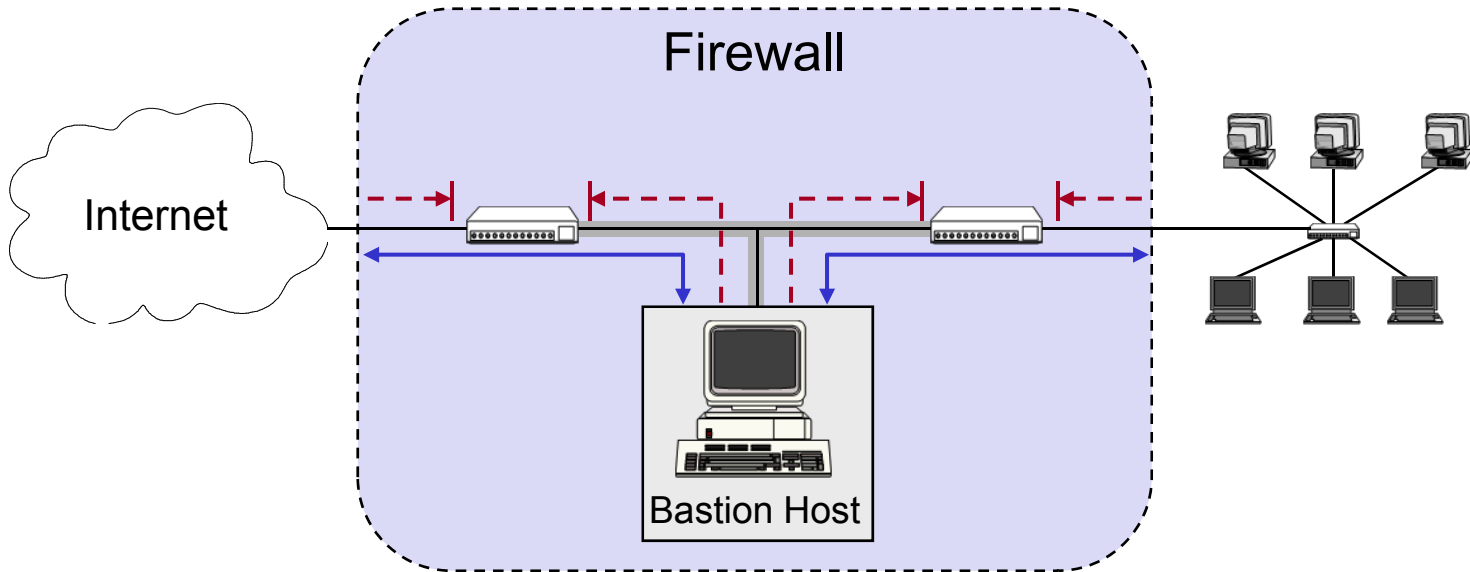
The Screened Host Architecture



- ❑ The packet filter:
 - ❑ Allows permitted IP traffic to flow between the screened host and the Internet
 - ❑ Blocks all direct traffic between other internal hosts and the Internet
- ❑ The screened host provides proxy services:
 - ❑ Despite partial protection by the packet filter the screened host acts as a bastion host



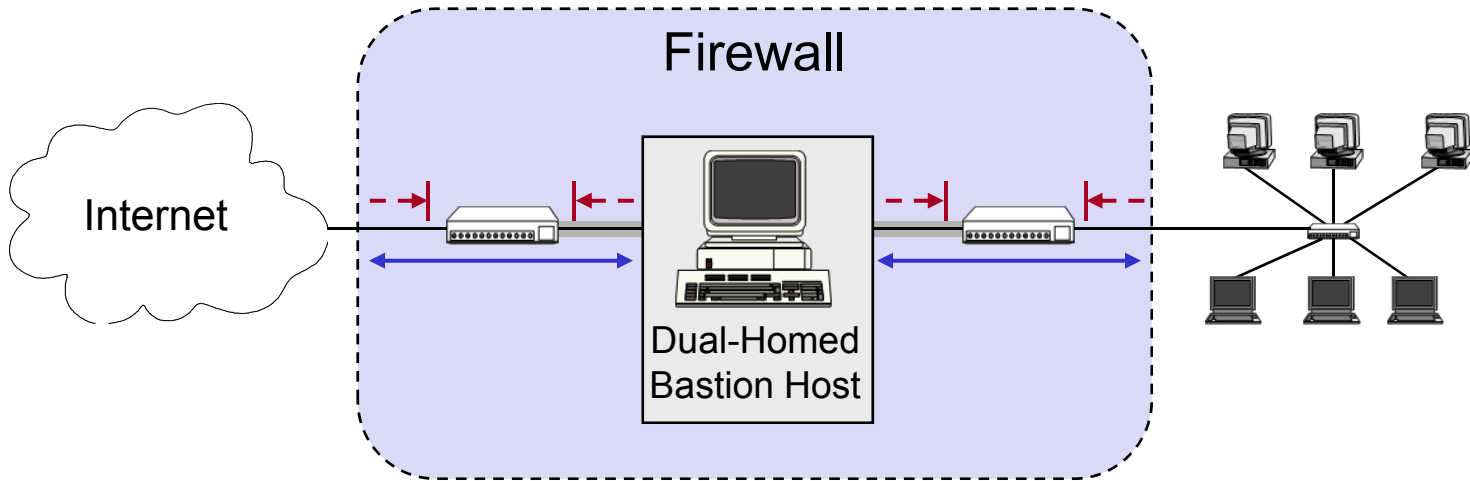
The Screened Subnet Architecture



- ❑ A perimeter network is created between two packet filters
- ❑ The inner packet filter serves for additional protection in case the bastion host is ever compromised:
 - ❑ For example, this avoids a compromised bastion host to sniff on internal traffic
- ❑ The perimeter network is also a good place to host a publicly accessible information server, e.g. a www-server

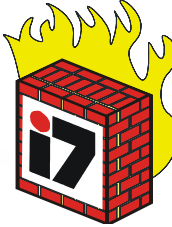


The Split Screened Subnet Architecture



- ❑ A dual-homed bastion host splits the perimeter network in two distinct networks
- ❑ This provides defense in depth, as:
 - ❑ The dual-homed bastion host provides finer control on the connections as his proxy services are able to interpret application protocols
 - ❑ The bastion host is protected from external hosts by an outer packet filter
 - ❑ The internal hosts are protected from the bastion host by an inner packet filter

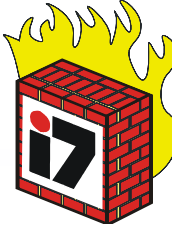
Packet Filtering



- ❑ What can be done with packet filtering?
 - ❑ Theoretically speaking everything, as all information exchanged in a communication relation is transported via packets
 - ❑ In practice, however, the following observations serve as a guide:
 - Operations that require quite detailed knowledge of higher layer protocols or prolonged tracking of past events are easier to realize in proxy systems
 - Operations that are simple but need to be done fast and on individual packets are easier to do in packet filtering systems

- ❑ Actions of a packet filter
 - ❑ Pass the packet
 - ❑ Drop the packet
 - ❑ Possibly, log the passed or dropped packet (entirely or parts of it)
 - ❑ Possibly, pass an error message to the sender (may help an attacker!)

Packet Filtering



- ❑ More elaborate packet filtering:
 - ❑ *Stateful or dynamic packet filtering:*
 - Example 1: *“Let incoming UDP packets through only if they are responses to outgoing UDP packets that have been observed”*
 - Example 2: *“Accept TCP packets with the SYN bit set only as part of TCP connection initiation”*
 - ❑ *Protocol checking:*
 - Example 1: *“Let in packets bound for the DNS port, but only if they are formatted like DNS packets”*
 - Example 2: *“Do not allow HTTP transfers to these sites”*

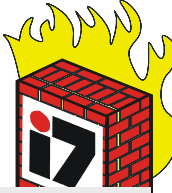
- ❑ However, more elaborate packet filtering consumes more resources!

Packet Filtering



- ❑ Specifying packet filtering rules
 - ❑ As a packet filter protects one part of a network from another one, there is an implicit notion of the direction of traffic flow:
 - *Inbound*: The traffic is coming from an interface which is outside the protected network and its destination can be reached on an interface which is connected to the protected network
 - *Outbound*: the opposite of inbound
 - For every packet filtering rule this direction is specified as either “*inbound*”, “*outbound*”, or “*either*”
 - ❑ Source and destination address specifications can make use of wildcards, e.g. 125.26.*.* denotes all addresses starting with 125.26.
 - In our examples, we denote often simply denote addresses as “*internal*” or “*external*” when we want to leave exact network topology out of account
 - ❑ We assume filtering rules to be applied in the order of specification, that means the first rule that matches a packet is applied

An Example Packet Filtering Ruleset



Rule	Direction	Src. Addr.	Dest. Addr.	Protocol	Src. Port	Dest. Port	ACK	Action
A	Inbound	External	Internal	TCP		25		Permit
B	Outbound	Internal	External	TCP		>1023		Permit
C	Outbound	Internal	External	TCP		25		Permit
D	Inbound	External	Internal	TCP		>1023		Permit
E	Either	Any	Any	Any		Any		Deny

- ❑ This first ruleset aims to specify, that incoming and outgoing email should be the only allowed traffic into and out of a protected network
- ❑ Email is relayed between two servers by transferring it to an SMTP-daemon on the target server (server port 25, client port > 1023)
- ❑ Rule A allows incoming email to enter the network and rule B allows the acknowledgements to exit the network
- ❑ Rules C and D are analogous for outgoing email
- ❑ Rule E denies all other traffic

An Example Packet Filtering Ruleset



- ❑ Consider, for example, a packet which “wants” to enter the protected subnet and has a forged IP source address from the internal network:
 - ❑ As all allowed inbound packets must have external source and internal destination addresses (A, D) this packet is successfully blocked
 - ❑ The same holds for outbound packets with external source addresses (B, C)
- ❑ Consider now telnet traffic:
 - ❑ As a telnet server resides usually at port 23, and all allowed inbound traffic must be either to port 25 or to a port number > 1023, incoming packets to initiate an incoming telnet connection are successfully blocked
 - ❑ The same holds for outgoing telnet connections
- ❑ However, the ruleset is flawed as, for example, it does not block the X11-protocol for remote operation of X-Windows applications:
 - ❑ An X11-server usually listens at port 6000, clients use port numbers > 1023
 - ❑ Thus, an incoming X11-request is not blocked (B), neither is any answer (D)
 - ❑ This is highly undesirable, as the X11-protocol offers many vulnerabilities to an attacker, like reading and manipulating the display and keystrokes

An Example Packet Filtering Ruleset



Rule	Direction	Src. Addr.	Dest. Addr.	Protocol	Src. Port	Dest. Port	ACK	Action
A	Inbound	External	Internal	TCP	>1023	25		Permit
B	Outbound	Internal	External	TCP	25	>1023		Permit
C	Outbound	Internal	External	TCP	>1023	25		Permit
D	Inbound	External	Internal	TCP	25	>1023		Permit
E	Either	Any	Any	Any	Any	Any		Deny

- ❑ The above flaw can be fixed by including the source ports into the ruleset specification:
 - ❑ As now outbound traffic to ports >1023 is allowed only if the source port is 25 (B), traffic from internal X-clients or -servers (port >1023) will be blocked
 - ❑ The same holds for inbound traffic to ports >1023 (D)
- ❑ However, it can not be assumed for sure, that an attacker will not use port 25 for his attacking X-client:
 - ❑ In this case the above filter will let the traffic pass

An Example Packet Filtering Ruleset



Rule	Direction	Src. Addr.	Dest. Addr.	Protocol	Src. Port	Dest. Port	ACK	Action
A	Inbound	External	Internal	TCP	>1023	25	Any	Permit
B	Outbound	Internal	External	TCP	25	>1023	Yes	Permit
C	Outbound	Internal	External	TCP	>1023	25	Any	Permit
D	Inbound	External	Internal	TCP	25	>1023	Yes	Permit
E	Either	Any	Any	Any	Any	Any	Any	Deny

- ❑ This problem can be addressed by also specifying TCP's ACK-bit:
 - ❑ As the ACK-bit is required to be set in rule B, it is not possible to open a new TCP connection in the outbound direction to ports >1023, as TCP's connect-request is signaled with the ACK-bit not set
 - ❑ The same holds for the inbound direction, as rule D requires the ACK bit
- ❑ As a basic rule, any filtering rule that permits incoming TCP packets for outgoing connections should require the ACK-bit (however, in practice usually the SYN-bit is checked)

An Example Packet Filtering Ruleset



Rule	Direction	Src. Addr.	Dest. Addr.	Protocol	Src. Port	Dest. Port	ACK	Action
A	Inbound	External	Bastion	TCP	>1023	25	Any	Permit
B	Outbound	Bastion	External	TCP	25	>1023	Yes	Permit
C	Outbound	Bastion	External	TCP	>1023	25	Any	Permit
D	Inbound	External	Bastion	TCP	25	>1023	Yes	Permit
E	Either	Any	Any	Any	Any	Any	Any	Deny

- ❑ If the firewall comprises a bastion host, the packet filtering rules should further restrict traffic flow (→ screened host architecture):
 - ❑ As in the modified rules above only traffic between the Internet and the bastion host is allowed, external attackers can not attack SMTP on arbitrary internal hosts any longer
- ❑ In a screened subnet firewall, two packet filtering routers are set up:
 - ❑ one for traffic allowed between the Internet and the bastion host, and
 - ❑ one for traffic allowed between the bastion host and the internal network

Summary (what do I need to know)



- ❑ Defense techniques
 - ❑ Principles of TCP SYN flood and ICMP/UDP flood
 - ❑ Countermeasures (principles)
 - SYN cookies
 - Rate limiting
 - Redirection
- ❑ IP address spoofing and traceback
 - ❑ What is address spoofing?
 - ❑ How to detect spoofed packets?
- ❑ Firewalls
 - ❑ Architecture
 - ❑ How to use simple filter lists

Additional References



- [Adler2002] M. Adler, "Tradeoffs in Probabilistic Packet Marking for IP Traceback," Proceedings of 44th Annual ACM Symposium on Theory of Computing, Montreal, Quebec, Canada, 2002, pp. 407-418.
- [Dean2001] D. Dean, M. Franklin, and A. Stubblefield, "An Algebraic Approach to IP Traceback," Proceedings of 2001 Network and Distributed Systems Security Symposium, February 2001.
- [Doepfner2000] T. D. Doepfner, P. N. Klein, and A. Koyfman, "Using Router Stamping to Identify the Source of IP Packets," Proceedings of 7th ACM conference on Computer and Communications Security, Athens, Greece, 2000, pp. 184-189.
- [Ioannidis2002] J. Ioannidis and S. M. Bellovin. Implementing Pushback: Router-Based Defense Against DDoS Attacks". Proceedings of Network and Distributed System Security Symposium, Catamaran Resort Hotel San Diego, California, February 2002
- [Krishnamoorthy2004] S. Krishnamoorthy and P. Dasgupta, "Tackling Congestion to Address Distributed Denial of Service: A Push-Forward Mechanism," Proceedings of IEEE Globecom 2004, Dallas, TX, USA, December 2004.
- [Lee2004b] T.-H. Lee, W.-K. Wu, and T.-Y. W. Huang, "Scalable Packet Digesting Schemes for IP Traceback," Proceedings of IEEE International Conference on Communications, Paris, France, June 2004b.
- [Li2002] J. Li, J. Mirkovic, M. Wang, P. Reiher, and L. Zhang, "SAVE: Source Address Validity Enforcement Protocol," Proceedings of IEEE Infocom 2002, New York, USA, June 2002. [Mahajan2001] R. Manajan, et al., "Controlling High Bandwidth Aggregates in the Network". Proceedings of SIGCOMM 2001.
- [Mirkovic2004] J. Mirkovic and P. Reiher, "A Taxonomy of DDoS Attack and DDoS Defense Mechanisms," *ACM SIGCOMM Computer Communication Review*, vol. 34, April 2004, pp. 39-53.
- [Snoeren2001] A. C. Snoeren, C. Partridge, L. A. Sanchez, C. E. Jones, F. Tchakountio, S. T. Kent, and W. T. Strayer, "Hash-Based IP Traceback," Proceedings of ACM SIGCOMM 2001 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication, August 2001.
- [Song2001] D. X. Song and A. Perrig, "Advanced and Authenticated Marking Schemes for IP Traceback," Proceedings of IEEE Infocom 2001, 2001.
- [Wang2002] H. Wang, D. Zhang, and K. G. Shin, "Detecting SYN Flooding Attacks," Proceedings of IEEE INFOCOM 2002, 2002.
- [Zwi00a] E. Zwicky, S. Cooper, B. Chapman. *Building Internet Firewalls*. Second Edition, O'Reilly, 2000.