Chair for Network Architectures and Services
Department of Informatics
TU München – Prof. Carle

Network Security

Chapter 7
Middleboxes

Overview

- Introduction
- Firewalls
- Application Proxies
- Networks Address Translators (NAT)
- Virtual Private Networks
- Case study: Linux Netfilter

Overview

- Introduction
  - Firewalls
  - Application Proxies
  - Networks Address Translators
  - Virtual Private Networks
  - Case study: Linux Netfilter

Introduction

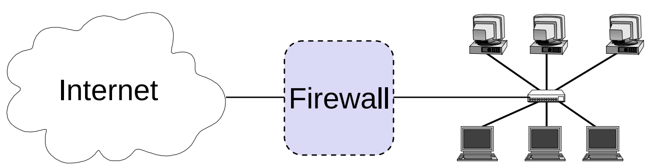
- Definition:
  - *"A middlebox is defined as any intermediary device performing functions other than the normal, standard functions of an IP router on the datagram path between a source host and destination host."* [RFC3234]
- The Internet was originally designed with the end-to-end connectivity principle
- However, in the meanwhile there are many devices on the datagram path that manipulate the IP packets
- [RFC3234] provides an overview of some commonly used middleboxes
  - e.g. firewalls, NATs, proxies, transcoders, load balancers, anonymisers
- In this chapter, we will restrict the discussion to some types of middleboxes that perform security-related manipulation of packets:
  - Firewalls, proxies, NATs, VPNs gateways

Overview

- Introduction
- **Firewalls**
- Application Proxies
- Networks Address Translators
- Virtual Private Networks
- Case study: Linux Netfilter

Introduction to Network Firewalls (1)

- 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 network level

Introduction to Network Firewalls (2)

- 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 block unwanted traffic if the traffic can be characterized,
    - e.g. with an IP 5-tuple: IP source address, IP destination address, source port number, destination port number, transport protocol
  - A firewall can limit exposure to security problems in one part of a network

Introduction to Network Firewalls (3)

- 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
  - A firewall can't protect against completely new threats
  - A firewall can't fully protect against viruses,
    - e.g. if viruses are spread through emails, and the email service is allowed through the firewall, which is typically the case
  - A firewall does not perform cryptographic operations, e.g. message authentication

    (however, firewalls are often co-located with VPN end-points)
  - A firewall can't set itself up correctly ($\Rightarrow$ cost of operation)

Protocol Fields Important for Firewalls (1)

- Access Protocol:
  - Network Layer Protocol: IP, Appletalk, IPX (Novell), DecNet, etc.
    (Note: among these protocols nowadays, IP is nearly the only protocol that is being deployed)
  - Access Protocol Addresses: Ethernet MAC Address, E.164 Address, etc.
    - These addresses either refers to the final source / destination or the addresses of the intermediate nodes of this link
- IP:
  - Source address
  - Destination address
  - Flags, especially the indication of an IP fragment
  - Transport protocol type: TCP, UDP, ICMP, ...
  - Options:
    - E.g. source routing:
      - the sender explicitly specifies the route an IP packet will take
      - as this is often used for attacks most firewalls discard these packets
    - In general, IP options are rarely used

Protocol Fields Important for Firewalls (2)

- TCP:
  - Source Port, Destination Port:
    - Evaluation of source and destination ports allow to determine (with a limited degree of confidence) the sending / receiving application, as most Internet services use well-known port numbers
  - Control:
    - ACK: this bit is set in every segment but the very first one transmitted in a TCP connection, it therefore helps to identify connection requests
    - SYN: this bit is only set in the first two segments of a connection, so it can be used to identify connection confirmations
    - RST: if set this bit indicates an ungraceful close of a connection, it can be used to shut peers up without returning helpful error messages
- Application Protocol:
  - In some cases a firewall might even need to peek into application protocol header fields
  - However, examination of application layer payloads is usually left to application proxies that are aware of the type of the application

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 is not permitted across the firewall
    - Incoming telnet connections are only allowed to one specific host

What Internet Services & Protocols should be Considered?

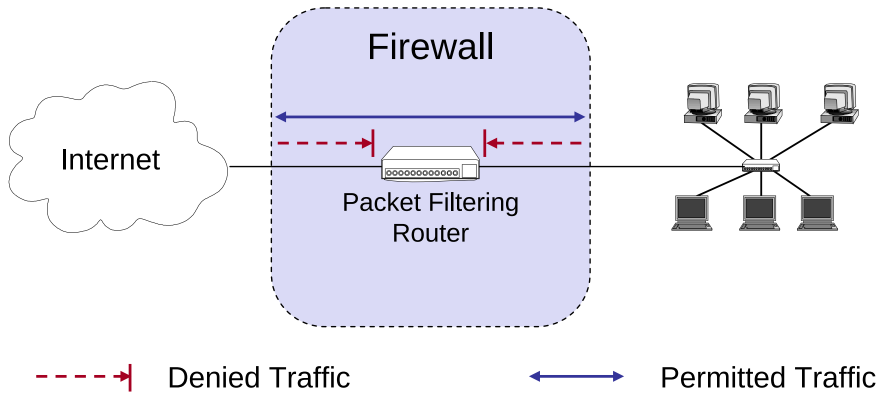
- Electronic mail: simple mail transfer protocol (SMTP)
- File exchange: file transfer protocol (FTP), network file system (NFS)
- Remote terminal access and command execution: telnet, rlogin, ssh
- Usenet news: network news transfer protocol (NNTP)
- World wide web: hypertext transfer protocol (HTTP)
- Information about people: finger
- Real-time conferencing services: CUseeMe, Netmeeting, Netscape conference, MBone tools, ...
- Name services: domain name service (DNS)
- Network management: simple network management protocol (SNMP)
- Time service: network time protocol (NTP)
- Window systems: X-Windows
- Printing systems: line printing protocols (LPR/LPD)

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
- *De-militarized zone (DMZ) :*
  - A subnetwork added between an external and an internal network, in order to provide an additional layer of security; also called **perimeter network**
- *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

Firewall Architectures (1)

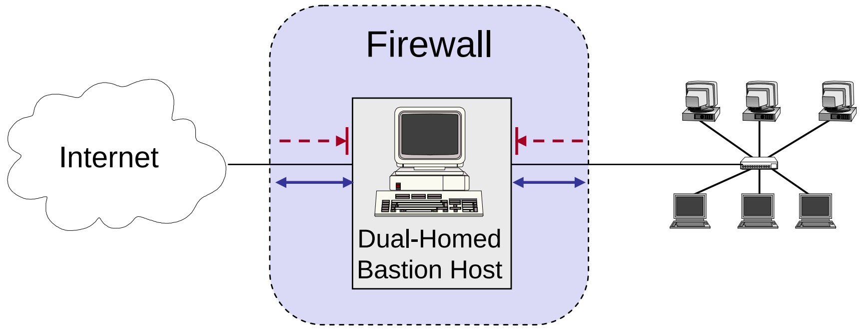
The Simple Packet Filter Architecture



- 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

Firewall Architectures (2)

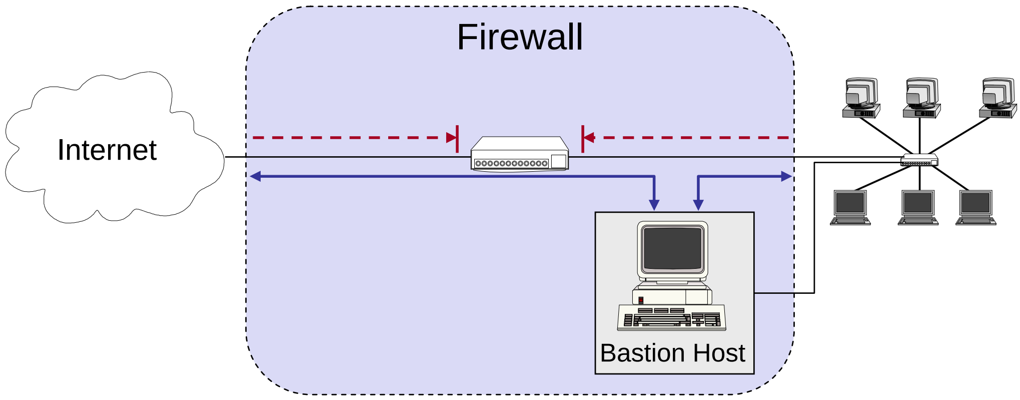
The Dual-Homed Host Architecture



- The dual-homed host provides:
  - Proxy services to internal and / or external clients
  - Potentially additional packet filtering capabilities
- Properties of the dual-homed host:
  - It has at least two network interfaces
- Drawback: As all permitted traffic passes through the bastion host, this may introduce a performance bottleneck

Firewall Architectures (3)

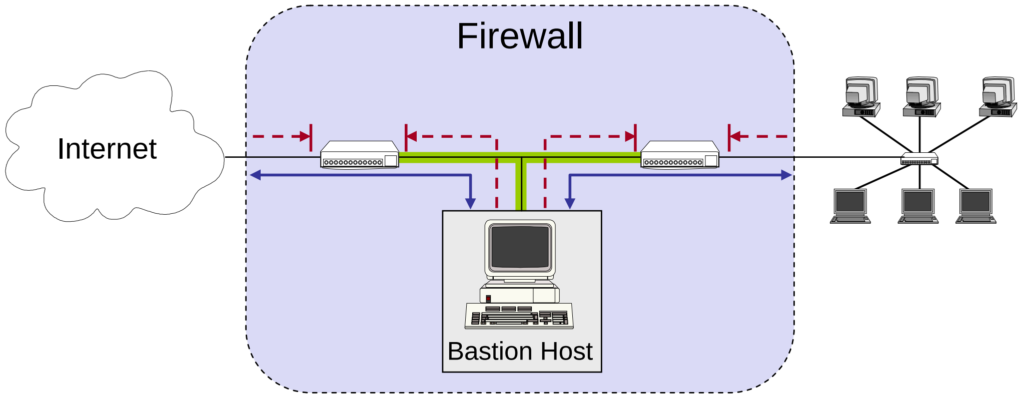
The Screened Host Architecture



- The packet filter:
  - Allows permitted IP traffic between screened host and the Internet
  - Blocks direct traffic between other internal hosts and the Internet
- The screened host provides proxy services:
  - The screened host acts as a bastion host, being partially protected by the packet filter

Firewall Architectures (4)

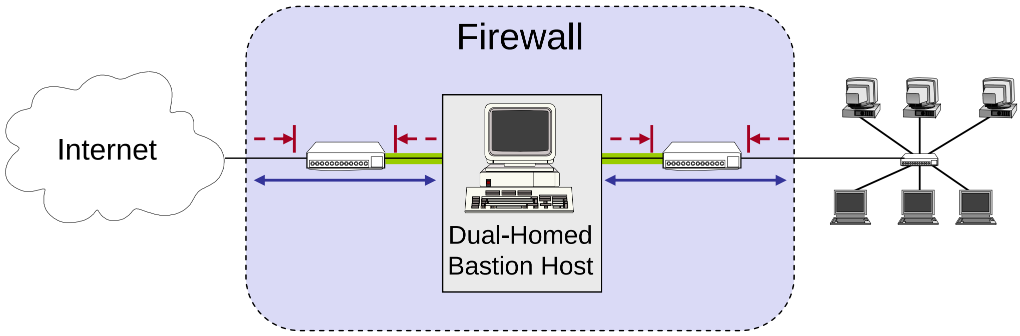
The Screened Subnet Architecture



- A DMZ is created between two packet filters
- The inner packet filter serves for additional protection in case the bastion host is compromised:
  - This avoids a compromised bastion host to sniff internal traffic
- The DMZ (i.e., perimeter network) is also a good place to host a publicly accessible information server, e.g. a www-server

Firewall Architectures (5)

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 (1)

- 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 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
- Basic packet filtering enables to control data transfer based on:
  - Source IP Address
  - Destination IP Address
  - Transport protocol
  - Source and destination application port
  - Specific protocol flags (e.g. TCP's ACK- and SYN-flag)
  - The network interface a packet has been received on

Packet Filtering (2)

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

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
  - For source and destination ports we sometimes write ranges, e.g. ">1023"
  - 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 (1)

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 ruleset 1 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 (2)

- Consider, for example, a packet which "wants" to enter the protected subnet and has a *spoofed* 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 (D), neither is any answer (B)
  - 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 (3)

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 flaw of ruleset 1 can be fixed in this updated ruleset 2 by including the source ports into the 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 (4)

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 in ruleset 3 by also specifying TCP's ACK-flag in rules B and D:
  - As the ACK-flag 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 has the ACK-flag not set
  - The same holds for the inbound direction, as rule D requires the ACK-flag to be set
- As a basic rule, any filtering rule that permits incoming TCP packets for outgoing connections should require the ACK-flag to be set

An Example Packet Filtering Ruleset (5)

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 ($\rightarrow$ 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

Bastion Hosts (1)

- A bastion host is defined as a host that is more exposed to the hosts of an external network than the other hosts of the network it protects
- A bastion host may serve for different purposes:
  - Packet filtering
  - Providing proxy services
  - A combination of both
- The principles for building a bastion hosts are extensions of those for securing any mission critical host:
  - Keep it simple
  - Prepare for the bastion host to be compromised:
    - Internal hosts should not trust it more than necessary
    - If possible, it should be connected in such a way to the network that it can not sniff internal traffic
    - Provide extensive logging for incident detection / analysis, if possible such that it can not be easily tampered with even when the host is compromised

Bastion Hosts (2)

- Further guidelines:
  - Make the bastion host unattractive:
    - Slower machines are less appealing targets and are less useful if compromised
    - However, if the bastion host offers some resource consuming service, e.g. WWW-service, it may be wiser not to make it too slow
    - The fewer tools are available on the bastion host, the less useful the machine is to an attacker
  - Get a reliable hardware configuration
  - The bastion host should be placed at a physically secure location
  - Disable any user accounts on the bastion host (e.g. only administrators can login to the Bastion host)
  - Secure the system logs (e.g. by writing them directly to a printer, or using the printer port to a dedicated PC which is not networked)
  - Do regular backups of the system logs and the configuration (using a dedicated backup device)
  - Monitor the machine closely (reboots, usage / load patterns, etc.)
  - If possible, restore the machine regularly from a prepared installation

Overview

Proxy Services

- *Proxy:*
  - 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* (e.g. a Web proxy)*,*
  - If it just passes the PDUs between the client and the server it is called a *circuit level proxy* (e.g. SOCKS proxy)
- Candidate services for proxying:
  - FTP, Telnet, DNS, SMTP, HTTP
- The use of a proxy service usually leads to the following situation:
  - The user of a proxy service has the illusion of exchanging data with the actual server host
  - The actual server has the illusion of exchanging data with the proxy host

Overview

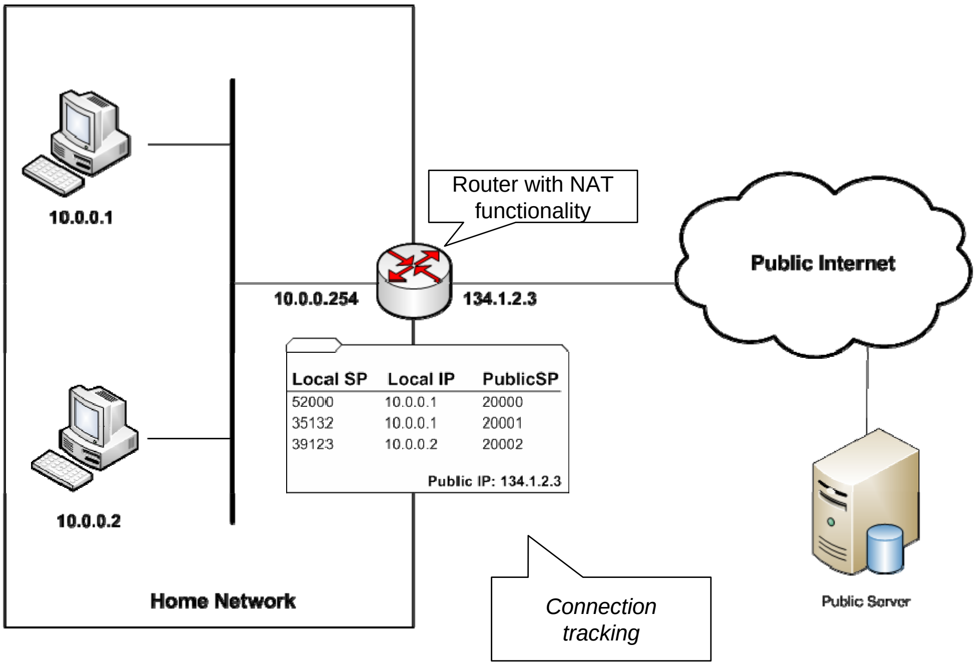
- Introduction
- Firewalls
- Application Proxies
- **Networks Address Translators**
- Virtual Private Networks
- Case study: Linux Netfilter

Network Address Translation (NAT) (1)

- *Network Address Translation (NAT):*
  - A procedure by which a router changes data in packets to modify the network addresses
- NATs were originally introduced due to the lack of IP addresses
- However, NATs are now also used to hide the internal topology of the network, and to limit connectivity to internal hosts
- NATs provide limited security
  - If a client behind a NAT is not directly reachable from the public Internet, the probability is lower that the client will be infected by a virus/worm that is spreading at the network layer
- However, for many applications, it is required that the client is reachable from the Internet,
  - e.g. Peer-to-Peer applications, Voice over IP (VoIP)

Network Address Translation (NAT) (2)

- Example



- Local SP: Local Source Port; Public SP: Public Source Port

Network Address Translation (NAT) (3)

- NAT present a big burden in today‘s Internet applications
- One of the main problems is that the behavior of NATs is not properly standardized
  - It depends on the implementation of the NAT how the mapping between internal and external IP addresses and port numbers is performed

Overview

- Introduction
- Firewalls
- Application Proxies
- Networks Address Translators

- **Virtual Private Networks**

- Case study: Linux Netfilter

Virtual Private Networks - Defintions

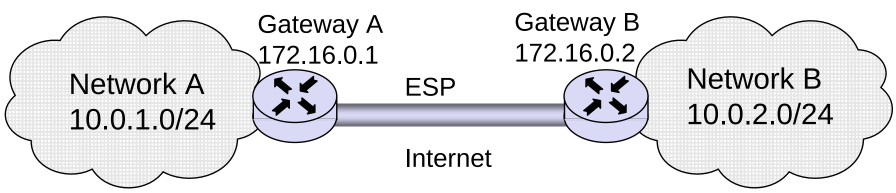
- Various definitions of the term *virtual private network (VPN):*
  - A virtual private network (VPN) is a network that **uses a public telecommunication infrastructure**, such as the Internet, to provide remote offices or individual users with secure access to their organization's network. An alternative to such a virtual private network is an expensive system of owned or leased lines that can be used only by a single organization. The goal of a VPN is to provide the organization with the same capabilities, but at a much lower cost. [SeSec08]
  - A communications environment in which **access is controlled** to permit peer connections only within a defined community of interest, and is constructed through **some form of partitioning** of a common underlying communications medium, where this underlying communications medium provides services to the network on a non-exclusive basis
  - A restricted-use, logical computer network that is constructed from the system resources of a relatively public, physical network (such as the Internet), often by using encryption, and often by tunneling links of the virtual network across the real network [RFC2828]

Techniques for building Virtual Private Networks

- Make use of dedicated links
  - ATM or Frame Relay virtual connections
  - Multi-Protocol Over ATM (MPOA)
  - Multi-Protocol Label Switching (MPLS)
- *Controlled route leaking / route filtering:*
  - Basic idea: control route propagation to the point that only certain networks receive routes for other networks
  - This intends to realize *"security by obscurity"*
- *Tunneling:*
  - Generic routing encapsulation (GRE)
  - PPP / PPTP / L2TP
    - Note: PPTP was developed by Microsoft and used MS-CHAP(Microsoft Challenge-handshake Authentication Protocol) and MPPE (Microsoft Point-to-Point Encryption Protocol)
    - It is currently considered as inherently insecure, since messages can be easily spoofed
  - IPSec (See examples provided in Chapter 8)
  - SSL (e.g. OpenVPN is based on SSL/TLS)
  - SSH: OpenSSH offers also (since Version 4.3) a possibility for building VPNs

Example: VPN with IPSec

- ***ESP Tunnel*** for VPN



- Configuration at Gateway A:
  - spdadd 10.0.1.0/24 10.0.2.0/24 any -P out ipsec
    esp/tunnel/**172.16.0.1-172.16.0.2**/require ;
  - spdadd 10.0.2.0/24 10.0.1.0/24 any -P in ipsec
    esp/tunnel/**172.16.0.2-172.16.0.1**/require ;

  Note
  - protocol and port (any)
  - the policy to use (–P) specifying direction (in/out), action (ipsec/discard/none), the protocol (ah/esp), the mode (tunnel/transport), and the level (uses/require)
- Configuration at Gateway B:
  - spdadd 10.0.2.0/24 10.0.1.0/24 any -P out ipsec
    esp/tunnel/**172.16.0.2-172.16.0.1**/require ;
  - spdadd 10.0.1.0/24 10.0.2.0/24 any -P in ipsec
    esp/tunnel/**172.16.0.1-172.16.0.2**/require ;

Overview

- Introduction
- Firewalls
- Proxies
- Networks Address Translators
- Virtual Private Networks
- **Case study: Linux Netfilter**

Linux Firewalls

- Firewalls in Linux are implemented using the *Netfilter* architecture (www.netfilter.org)
- The Linux command „iptables“ is used to add firewalls rules, e.g.

```
# Allow port 22 (ssh) new TCP connection from
# source IP address range 192.168.1.100/32
iptables -A INPUT -p tcp -s 192.168.1.100/32 --dport 22 -m state --state NEW -j ACCEPT
```

-A INPUT: append new rule to rule chain INPUT
-m (match)
-m state --state: NEW/ESTABLISHED/RELATED
-j (jump): ACCEPT/DROP

Netfilter Chains (1)

- The packets are processed in so-called „***chains***“
- A “chain” is a checklist of rules
- Each incoming or outgoing packet is sequentially checked against these rules
- Each rule says “if the packet header looks like this, then here's what to do with the packet”, e.g. drop or accept
- Firewall functionality is implemented using 3 chains
  - The „***input***“ chain: for processing packets addressed to this host
  - The „***output***“ chain: for processing packets coming from local processes and leaving this host
  - The „***forward***“ chain: for processing packets that are traversing this host.

    If a Linux machine is used as a router, this chain will be frequently used.

    If a Linux machine is used as an end-host and not as a Linux router, this chain will not be used

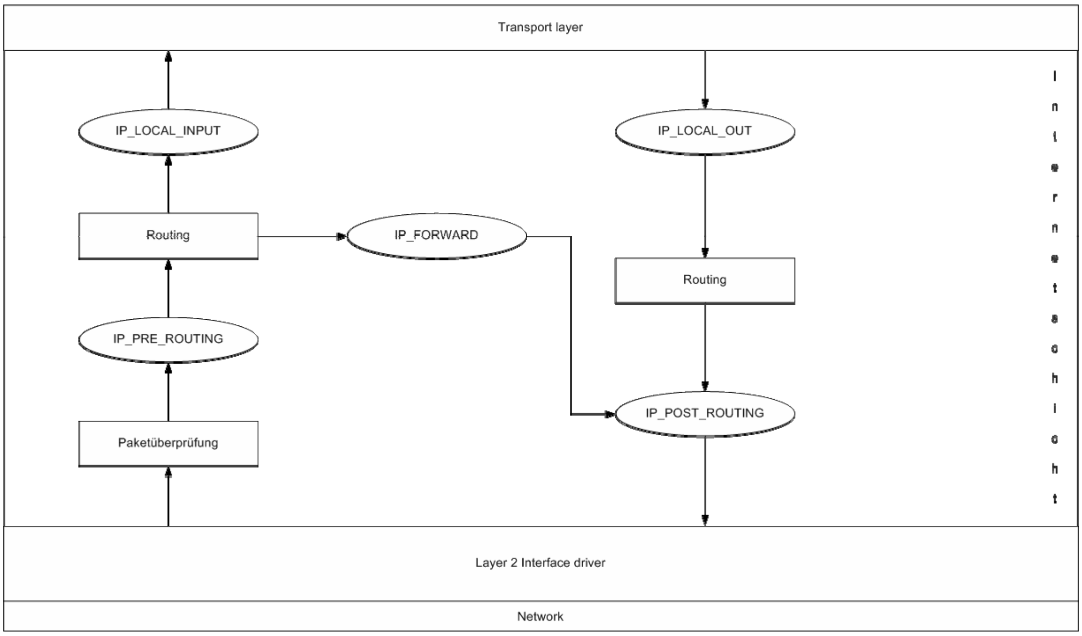
Netfilter Chains (2)

- Additionally, two chains are used for NAT functionality
  - „***Pre-Routing***“: here packets are processed before a routing decision is taken

    ➔ If the destination IP address needs to be modified, an appropriate rule is required here
  - “***Post-routing***”: here packets are processed after a routing decision has been taken

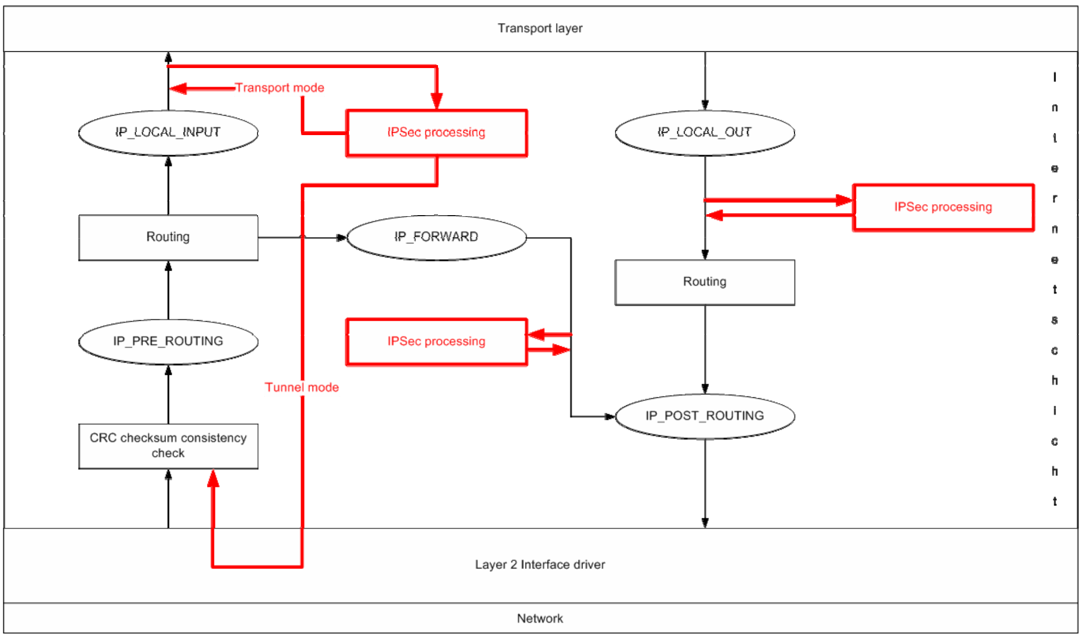
    ➔ The destination IP address can not be modified here.

Netfilter Chains



Native IPSec Support

- In Linux 2.6, native support for IPSec processing is integrated with the netfilter architecture



Additional References

[netfi08] "Netfilter Online Documentation",
http://www.netfilter.org/documentation/HOWTO//packet-filtering-HOWTO.html

[Sem96a] C. Semeria. *Internet Firewalls and Security.* 3Com Technical Paper, 1996.

[Wack95a] J. P. Wack, L.J. Carnahan. *Keeping Your Site Comfortably Secure: An Introduction to Internet Firewalls.* NIST Special Publication 800-10, 1995.

[Zwi00a] E. Zwicky, S. Cooper, B. Chapman. *Building Internet Firewalls.* Second Edition, O'Reilly, 2000.

[SeSec08] "SearchSecurity.com Definitions; virtual private network"
http://searchsecurity.techtarget.com/sDefinition/0,,sid14_gci213324,00.html