Firewall and IDS/IPS

Diana Gratiela Berbecaru
diana.berbecaru@polito.it

Politecnico di Torino
Dip. Automatica e Informatica

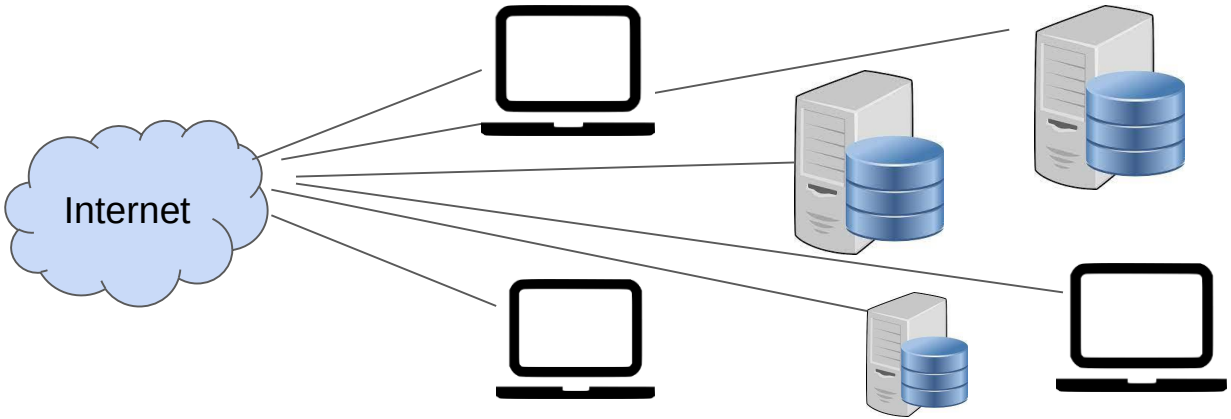
AY. 2023 - 2024

Acknowledgment

- **Slides content has been prepared by Prof. Antonio Lioy for the course Information Systems Security (2005 - 2022)**
  - some modifications applied
- **… so this set of slides is entirely compatible with the course of the previous year(s)**
- **Some figures have been imported from Chapter 10.2 «Proxy firewalls and firewalls architectures» in the book of Paul C. van Oorschot «Computer Security and the Internet: Tools and Jewels from Malware to Bitcoin» (https://people.scs.carleton.ca/~paulv/toolsjewels.html)**

Why (network) firewall(s)?

- **Assume a company network is composed of many servers and employee computers**
- **Assume the users inside the network are trusted, while the ones outside are not**
- **How to protect the systems (inside the network) from external attacks?**

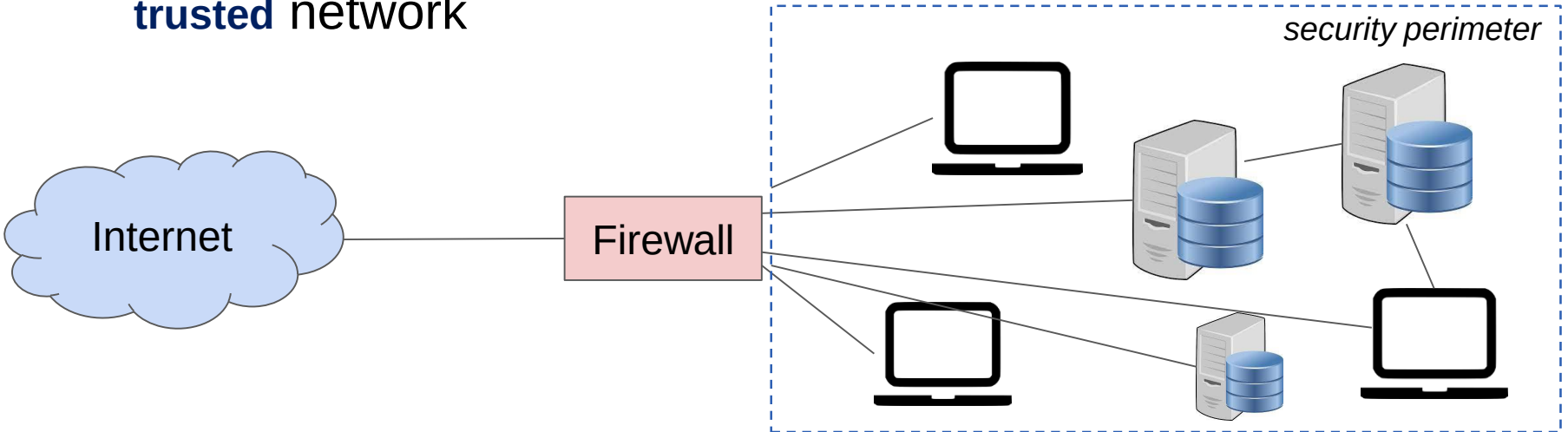


Why (network) firewall(s)? (II)

- **multiple network services = many risks**
  - each network connection creates opportunities for attacks
  - turning off all network services is not possible (some servers need to reached from outside, e.g. web server, DNS server, mail server)
- **multiple networked machines = many risks**
  - what if you have thousands of systems?
  - what if they run different hardware, operating systems, and have different configurations
  - what is there are systems that you are not even aware of?
- **Idea: instead of securing individual machines, we define a security perimeter and a point of access**
  - similar to controls at the airports, entrance in stadium, …

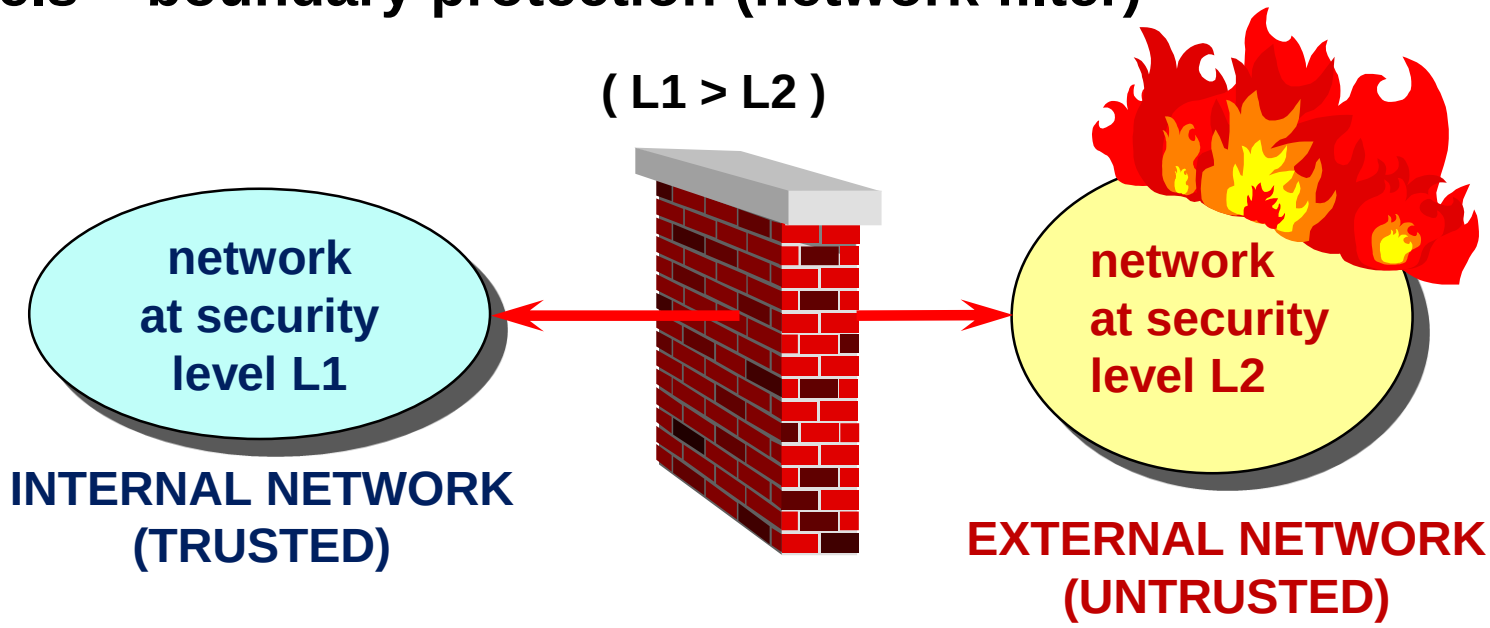
Network firewalls as single point of access

- **Single point of access in and out of (corporate) network**
  - any traffic that could affect systems must pass through the firewall
  - provides access control functionality: allow, deny, and optionally modify data passing between the **untrusted** network and the **trusted** network



What is a (network) firewall?

- **firewall = wall to protect against fire propagation**
- **controlled connection between networks at different security levels = boundary protection (network filter)**

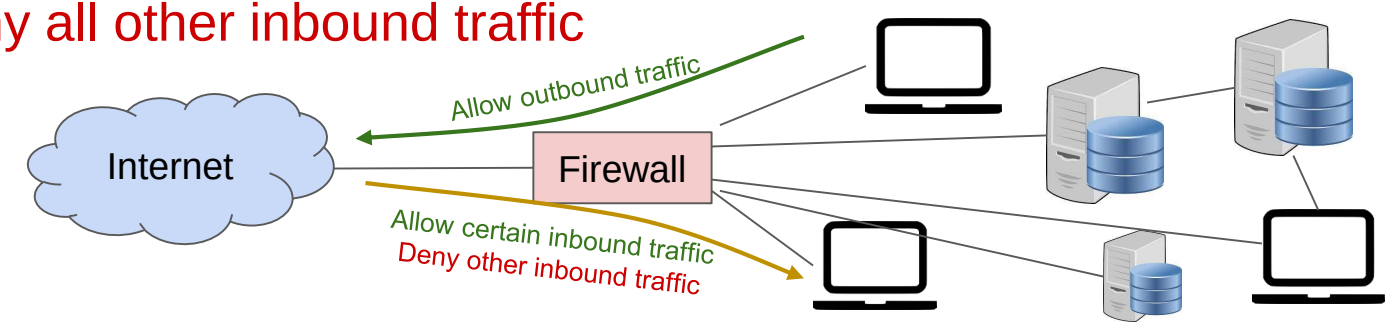


Inbound vs. Outbound

- **from the internal network viewpoint, packets arriving are inbound, and those leaving are outbound**
- **network access is controlled by policies**
  - **outbound policy**: defines what traffic is allowed to exit the network
  - **inbound policy**: defines what traffic is allowed to enter the network
  - remember: policies are based on the threat model – assume the users inside the network are trusted, those outside are not

Example of policies for a small office network

- **outbound policy: Allow outbound traffic**
  - for outgoing connections: users inside the network can connect to any service
- **inbound policy: Only some traffic is able to enter the network**
  - for incoming connections:
    - Allow inbound traffic to certain services (e.g. Web server, DNS, mail)
    - Allow inbound traffic in response an outbound connection
    - Deny all other inbound traffic



Ingress vs. Egress firewall

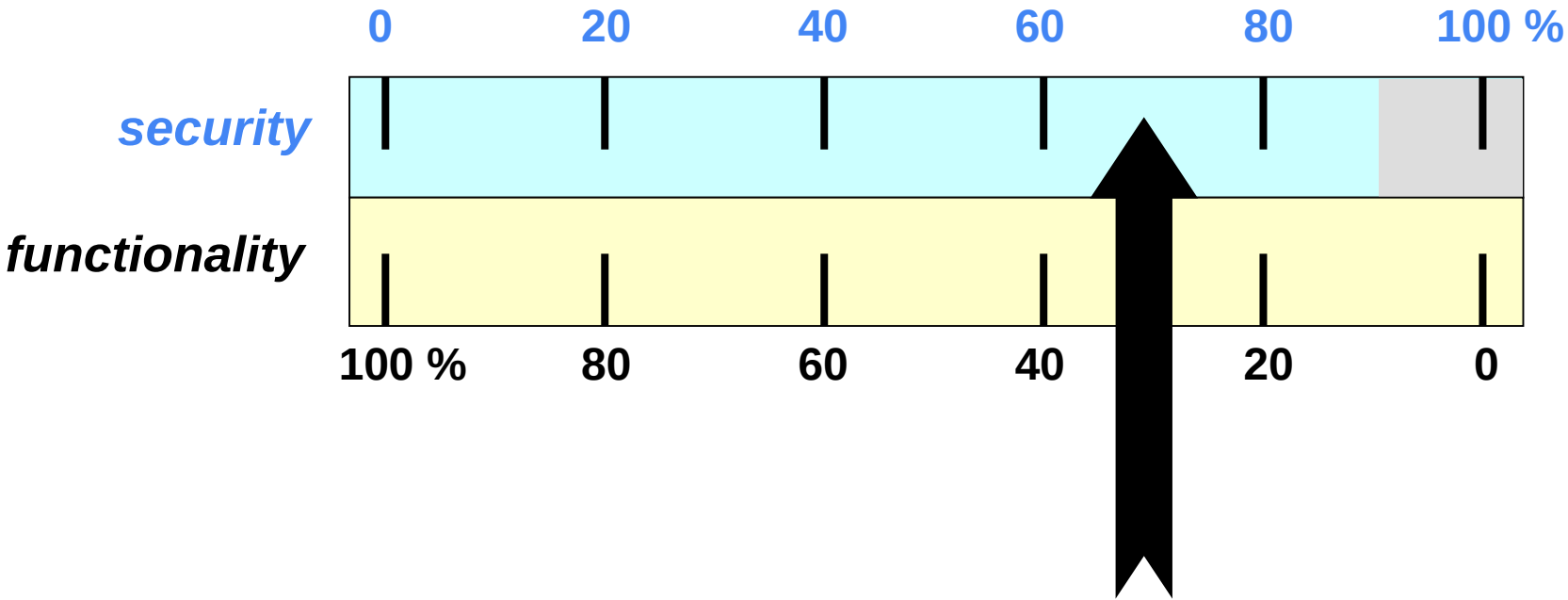
- **ingress firewall**
  - incoming connections
  - typically to select the (public) services offered
  - sometimes as part of an application exchange initiated by my users
- **egress firewall**
  - outgoing connections
  - typically to check the activity of my personnel (!)
- **easy classification for channel-based services (e.g. TCP applications), but difficult for message-based services (e.g. ICMP, UDP applications)**

Firewall design

You don’t “buy” a firewall, you design it!
(you can buy its components)

- **we need to achieve an optimal trade-off ...**
- **... between security and functionality**
- **... with minimum cost**

The security index



THE THREE COMMANDMENTS OF FIREWALL

I. the FW must be the **only** contact point of the internal network with the external one

II. only the “authorized” traffic can traverse the FW

III. the FW must be a highly secure system itself

D.Cheswick
S.Bellovin

Authorization policies

“All that is not explictly permitted, is forbidden”

- higher security (gatekeeper)
- more difficult to manage

Default-deny, permitlist, allowlist

Default-allow blocklist, denylist

“All that is not explictly forbidden, is permitted”

- lower security (open gates)
- more easy to manage

Default-deny vs. default-allow policies

- **Default-deny: by default, every network service is denied, unless is has been specifically listed as allowed (allowlist)**
  - start off with a list of few known servers that need to be visible to the outside worls and that have been judged to be reasonably safe
    - external users will be denied access to any service not on the list
    - needed services (e.g. a internal server used for some operations) may be added to the allowlist (after checking that they are reasonable safe and properly configured)
  - less convenient (services really needed in a company might be blocked) but is more secure
    - if some service that is safe has been mistakenly omitted from the allowlist, then the result is a loss of functionality (availability) but NOT a security breach

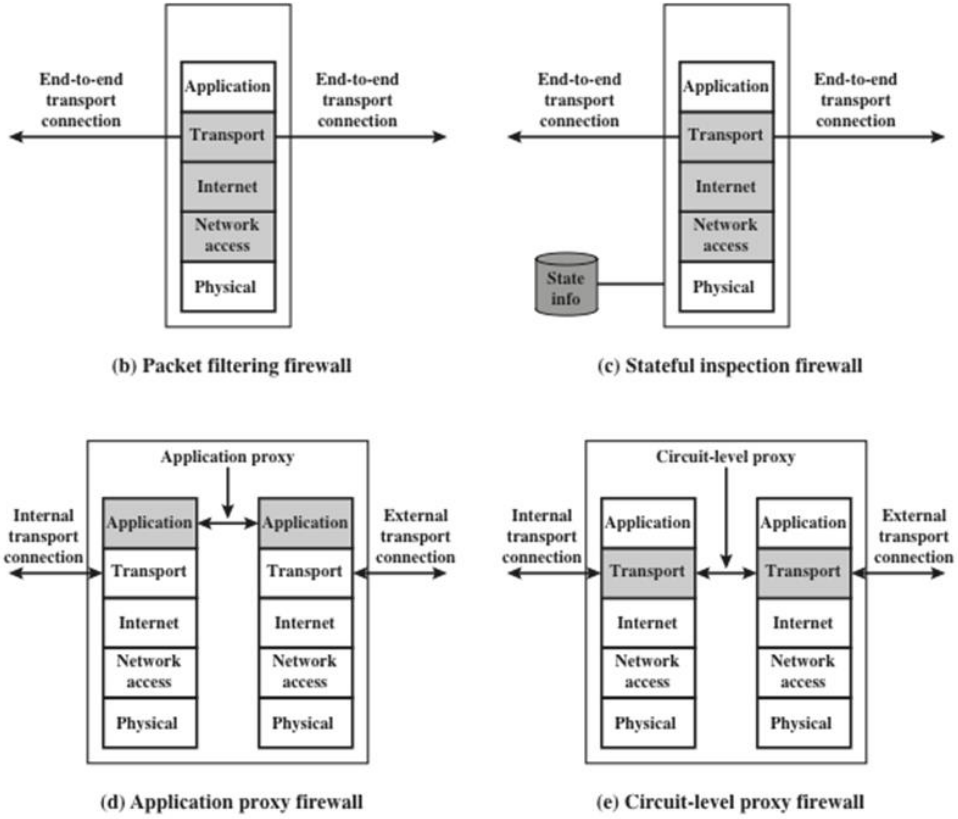
Default-allow vs. default-deny

- **Default-allow: be default, every network service is allowed, unless is has been specifically listed as denied (denylist).**
  - start off by allowing outside users access to all internal services, and then block a few that are known to be unsafe (blocklist)
  - requires knowing which services are potentially dangerous (e.g. affected by bugs) in order to block them
  - more convenient (everything typically stays working) but is more dangerous
    - if there is some dangerous service (present or future) that you ‘forget’ to add to the denylist, then the result will be a security failure/breach

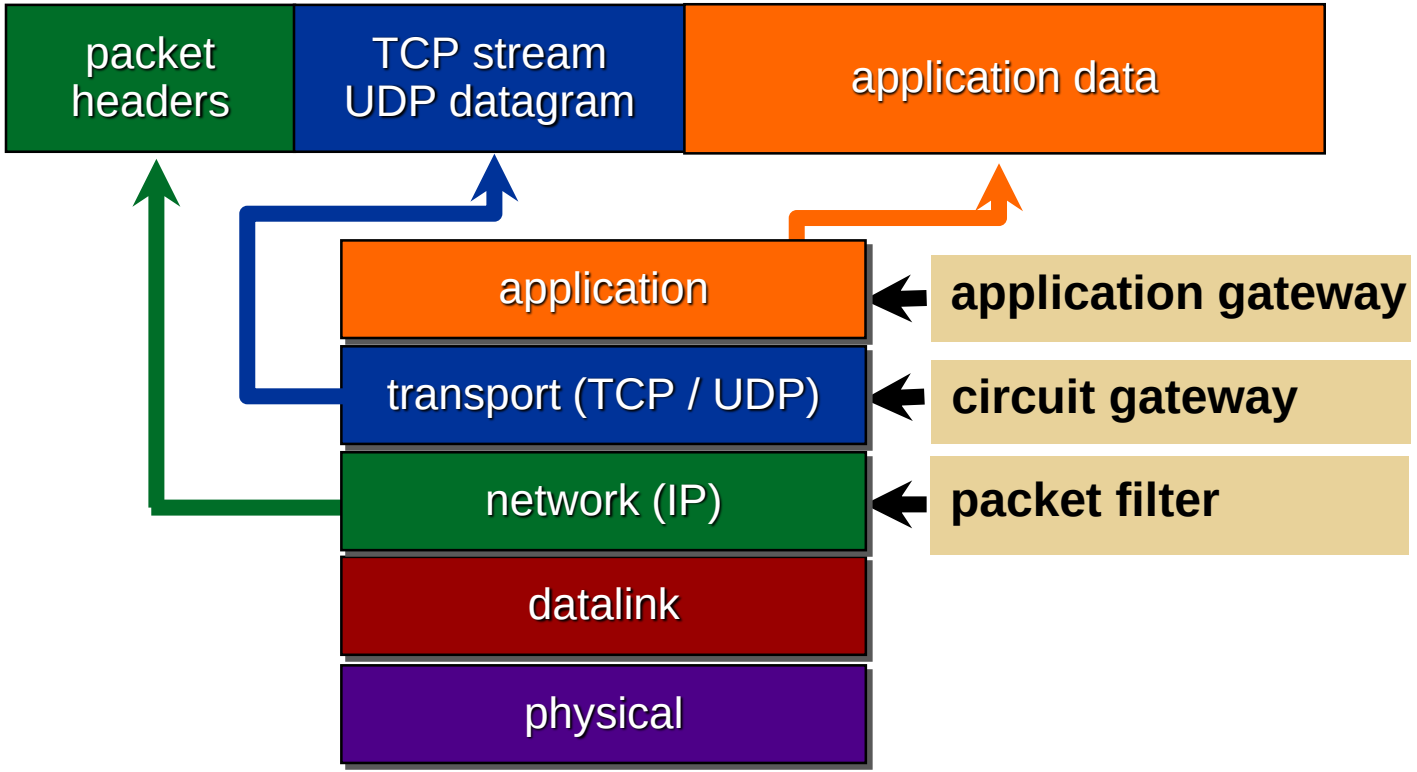
Firewall types

Types of firewalls

- **We discuss firewalls under two broad headings**
  - packet filters
  - proxy-type firewalls (gateways)



At which level are made the controls?



Firewall technologies/types

- **different controls at various network levels:**
  - (static/stateless) packet filter
  - stateful (dynamic) packet filter
  - cutoff proxy
  - circuit-level gateway / proxy
  - application-level gateway / proxy
  - stateful inspection
- **differences in terms of:**
  - performance
  - protection of the firewall O.S.
  - keeping or breaking the client-server model

(Stateless) Packet filter

- **historically available on routers, configured by an administrator**
- **checks each packet at network level against a list of rules**
  - IP header
  - transport (TCP, UDP) header
- **contains a list of rules of the form <condition, action>**
- **in a ‘first-matching rule’ packet filter, the action taken for a packet is that specified by the first matching rule whose condition is satisfied. Primary actions:**
  - allow (permit packet to pass)
  - drop (silently discard the packet)
  - reject (drop but also try to inform the source). Might result in sending a TCP RST packet, or for UDP an ICMP «destination unreachable»
- **each packet processed independently of the others**

Packet filter

- **a rule (typically) lists:**
  - source IP address, source port number, destination IP address, destination port number, protocol (e.g. tcp, udp), action (allow, drop)
  - a rule can use wildcards (*) to indicate ‘any’
- **rule example:**
  - permit incoming connections to our webserver:
    - ‘*src any dst 10.1.2.3/0.0.0.0 tcp 80 allow*’
  - only our internal DNS server can query DNS external servers:
    - ‘*src 10.1.2.1/0.0.0.0 dst any udp 53 allow*’
  - typically there is an (implicit) ‘deny all’ rule at the end
  - order is important (first match principle)

Packet filter: pros and cons

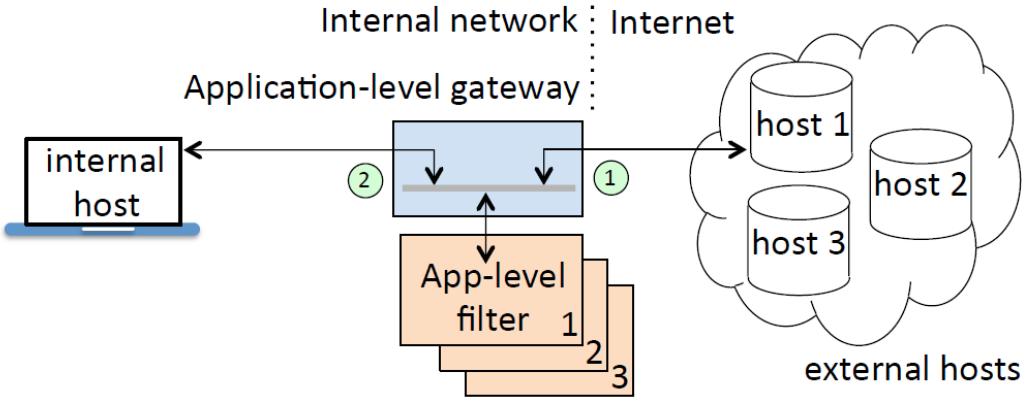
- **advantages:**
  - independent of applications
    - good scalability
    - approximate controls: easy to “fool” (e.g. IP spoofing, fragmented packets)
  - good performance
  - low cost (available on routers and in many OS)
- **disadvantages:**
  - difficult to support services with dynamically allocated ports (e.g. FTP)
  - complex to configure
  - difficult to perform user authentication

Stateful (dynamic) packet filter

- **similar to packet filter but maintains a *state***
  - keeps track of all open connections
  - if you see a packet from X to Y: is this packet on a connection initiated by X to Y or is it a response on a connection initiated by Y to X?
    - TCP flags (SYN, ACK) useful for this purpose
  - can distinguish new connections from those already open
    - state tables for open connections
    - packets matching one row in the table are passed without any further control
- **better performance than (stateless) packet filter**
- **still has many of the static packet filter limitations**

Application-level gateway (1)

- **composed by a set of app-level filters**
  - inspecting the payload at application level
  - they can transform data on the fly
- **every application needs a specific proxy (app-level filter)**
- **completely breaks the client/server model**



Application-level gateway (2)

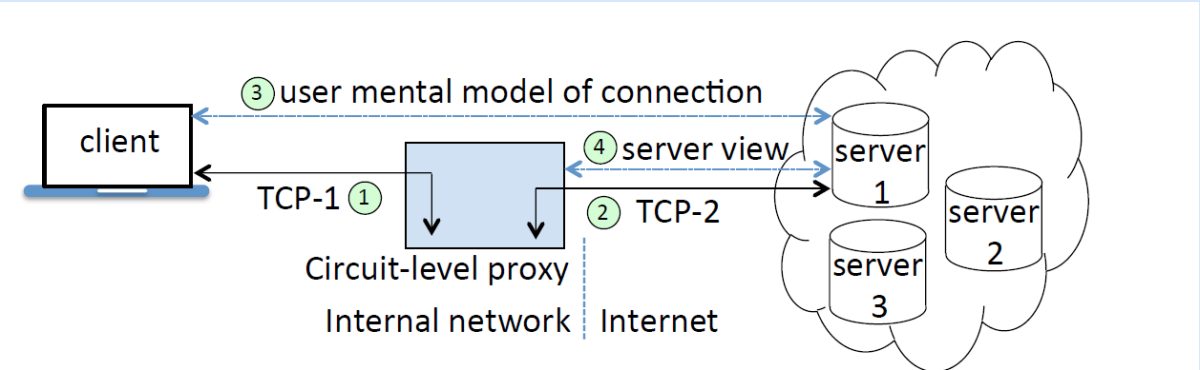
- **rules are more fine-grained and simple than packet filter rules**
- **rule example:**
  - deny dangerous HTTP methods ‘PUT, DELETE deny’
- **advantages:**
  - may optionally mask / renumber the internal IP addresses
  - more protection for the server (against attacks)
  - may authenticate the client
- **disadvantages:**
  - often requires modifications to the client application
  - the firewall OS may be exposed to attacks
  - problems with application-level security techniques that do not permit traffic inspection (e.g. TLS)

Application-level gateway (3)

- **disadvantages (cont.) :**
- **Remember: every application needs a specific proxy !**
  - delay in supporting new applications
    - if the application-level gateway does not implement the proxy code for a specific application, the service is not supported and cannot be forwarded across the firewall
  - heavy on resources (many processes)
  - low performance (user-mode processes)

Circuit-level gateway (1)

- **a generic proxy (i.e. not “application-aware”)**
  - creates a transport-level circuit between client and server …
  - … but it doesn’t understand or manipulate in any way the payload data
  - … it just copies between its two interfaces the TCP segments or UDP datagrams (if they match the access control rules)
  - … but, in doing this, it will re-assemble the IP packets and hence it will provide protection against some L3/L4 attacks



Circuit-level gateway (2)

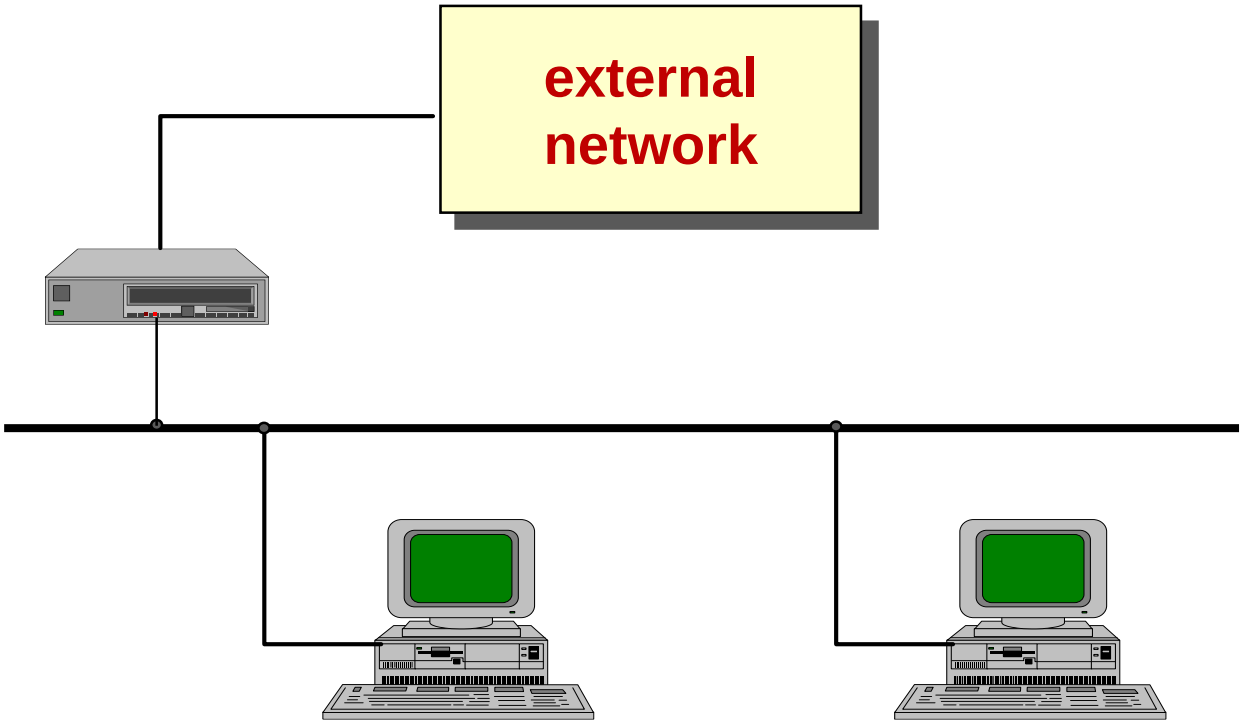
- **breaks the TCP/UDP-level client/server model during the connection**
  - more protection for the server
  - may authenticate the client
    - but this requires modification to the application
- **still exhibits many limitations of the packet filter**
- **SOCKS is the most famous one**

Firewall architectures

FW: basic components

- **screening router (choke)**
  router that filters traffic at network level
- **bastion host**
  secure system, with auditing
  - term borrowed from medieval castles: a bastion = a fortified cornerpoint or angled wall where to place defensive firepower
  - here, defensive host exposed to hostile network
- **application gateway (proxy)**
  service that works on behalf of an application, with access control
- **dual-homed gateway**
  system with two network cards and routing disabled

"Screening router" architecture

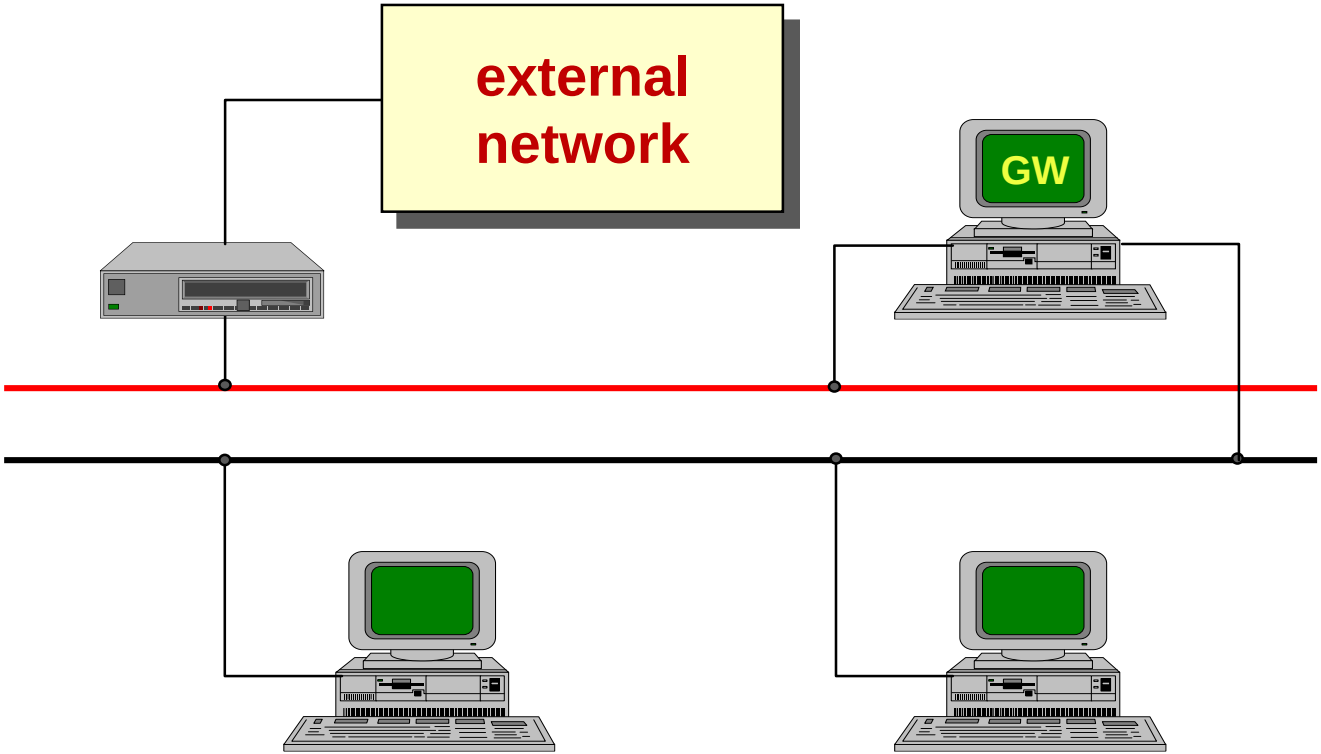


"Screening router" architecture

- **exploits the router to filter the traffic both at IP and upper levels**
- **no need for dedicated hardware**
- **no need for a proxy and hence no need to modify the applications**

- **simple, easy, cheap and ... insecure!**

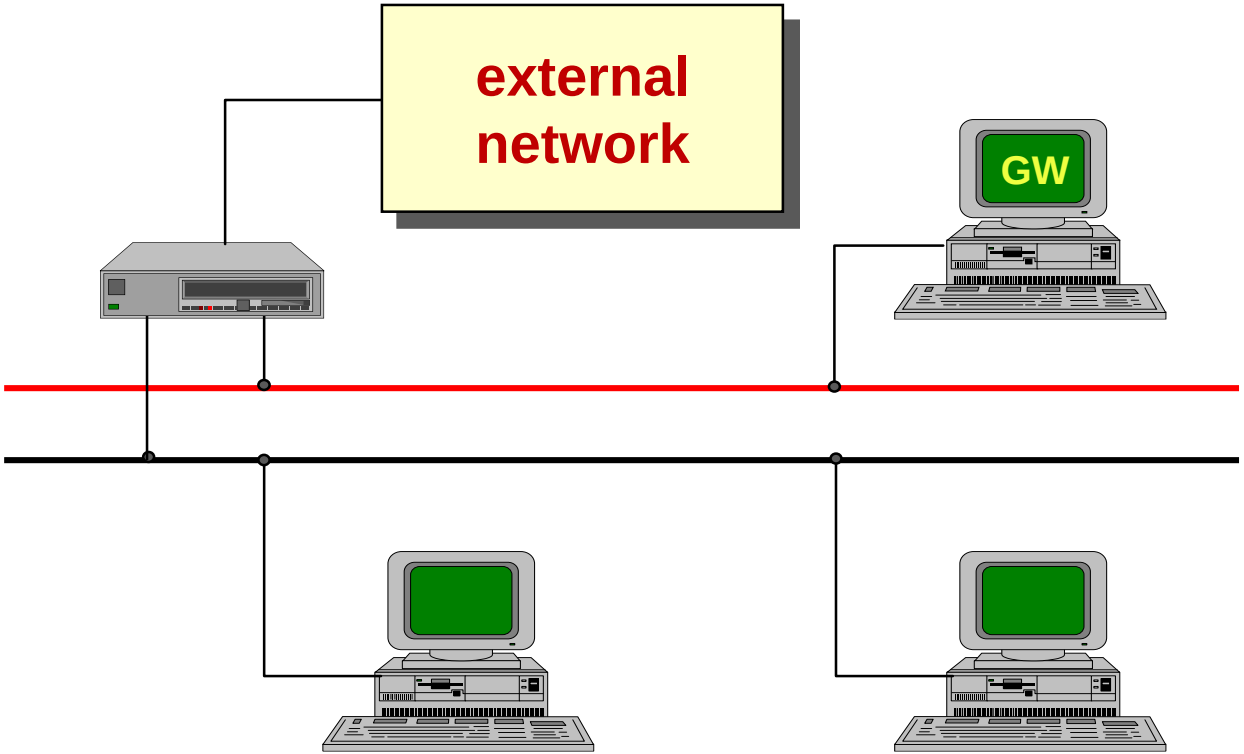
"Dual-homed gateway" architecture



"Dual-homed gateway" architecture

- **easy to implement**
- **small additional hardware requirements**
- **the internal network can be masqueraded**
- **unflexible**
- **large work overhead**

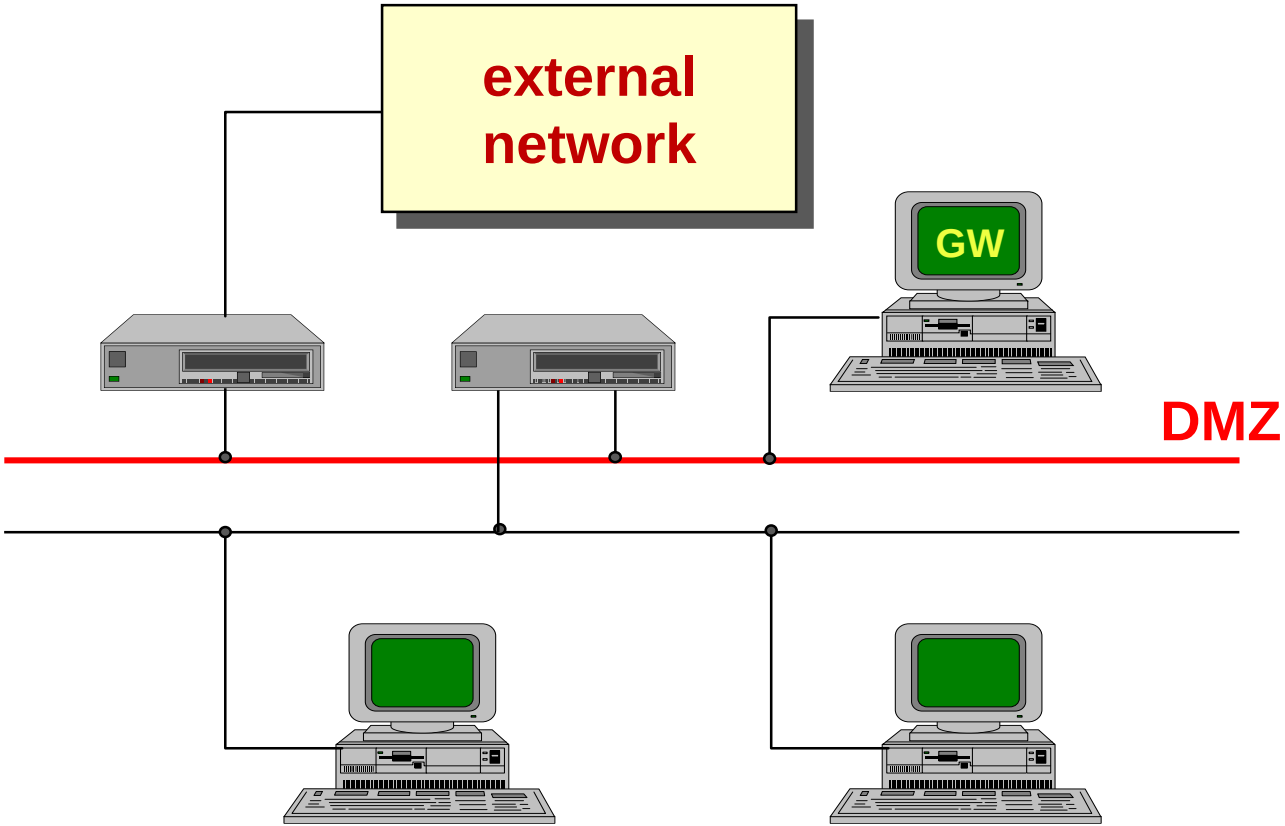
"Screened host" architecture



"Screened-host" architecture

- **router:**
  - blocks traffic INT > EXT unless from the bastion
  - blocks traffic EXT > INT unless goes to the bastion
  - exception: directly enabled services
- **bastion host runs circuit/application gateway to control the authorized services**
- **more expensive and complex to manage (two systems rather one)**
- **more flexible (no control over some services / hosts)**
- **only the hosts/protocols passing through the bastion can be masked (unless the router uses NAT)**

"Screened subnet" architecture

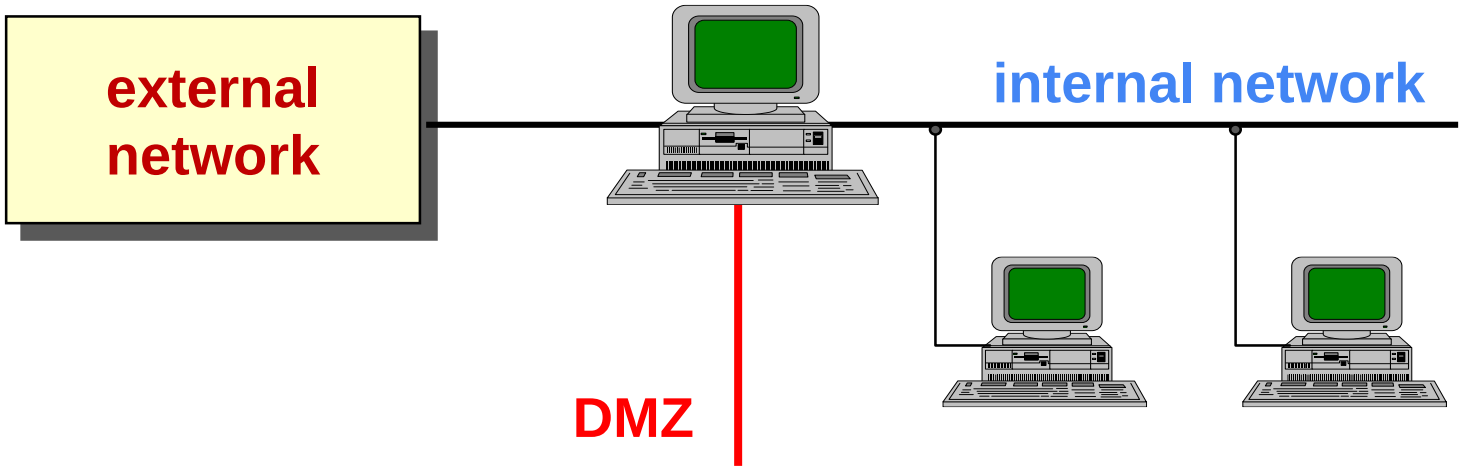


"Screened subnet" architecture

- **DMZ (De-Militarized Zone)**
- **the DMZ is home not only to the gateway but also to other hosts (tipically the public servers):**
  - Web
  - remote access
  - . . .
- **the routing may be configured so that the internal network is unknown**
- **expensive**

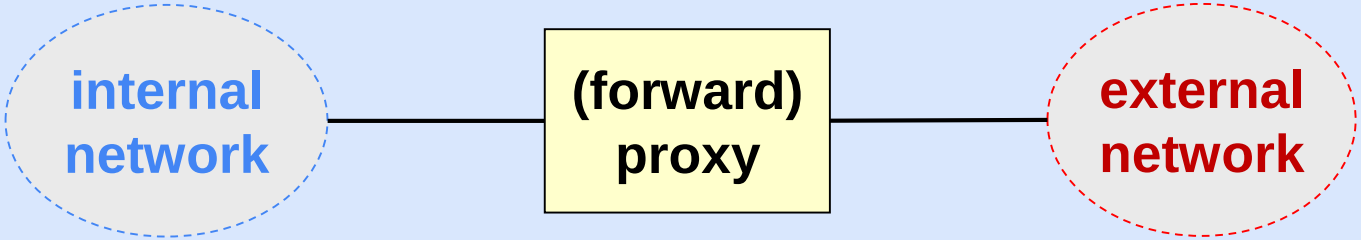
"Screened subnet" architecture

- to reduce costs and simplify management often the routers are omitted (and their function incorporated into the gateway)
- AKA “three-legged firewall”



HTTP (forward) proxy

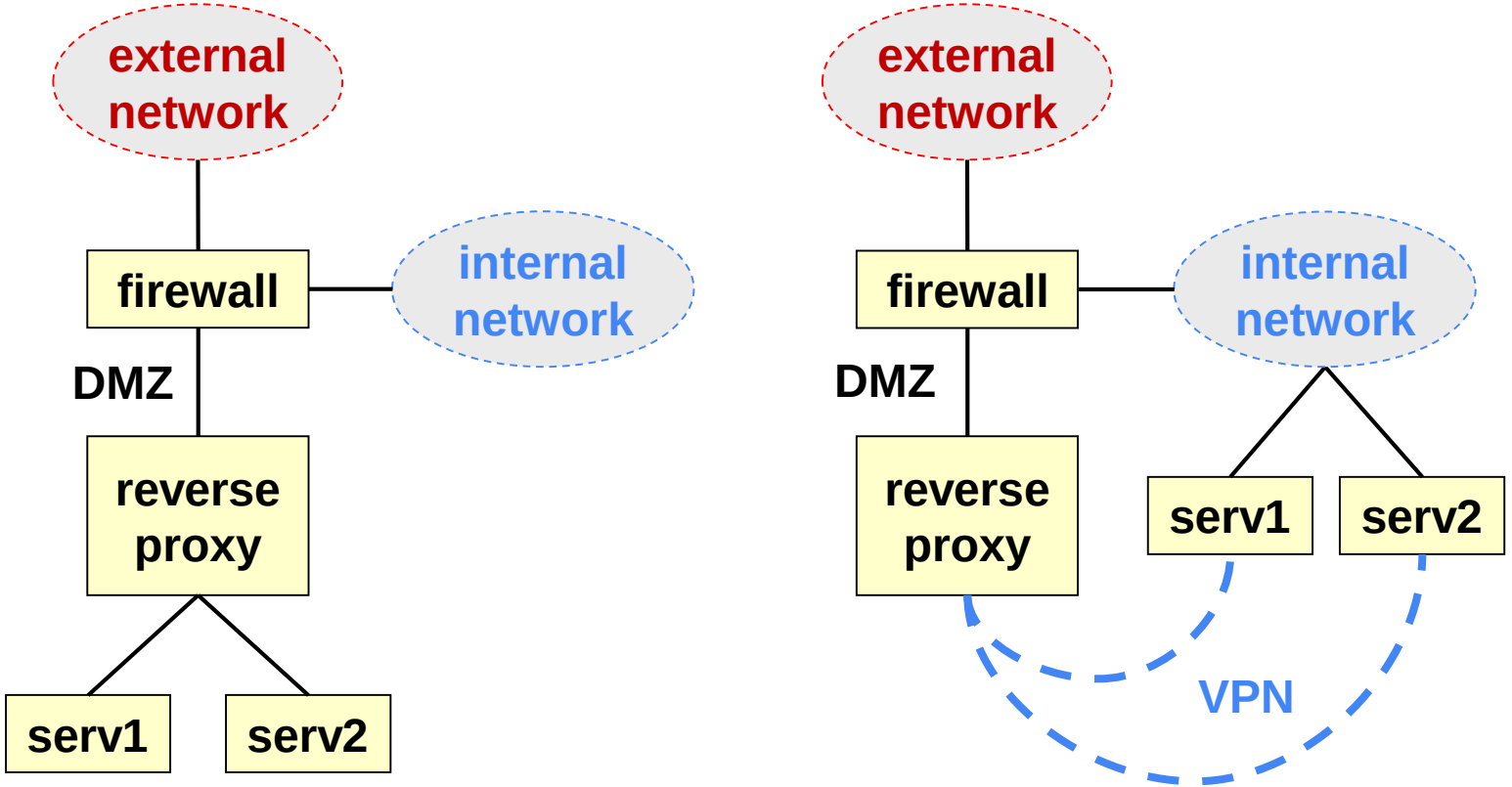
- **a HTTP server acting just as a front-end and then passing requests to the real server (external)**
- **it's an egress control**
- **benefits (besides network ACL):**
  - shared cache of external pages for all internal users
  - authentication + authorization of internal users
  - various controls (e.g. allowed sites, transfer direction, data types, …)



HTTP reverse proxy

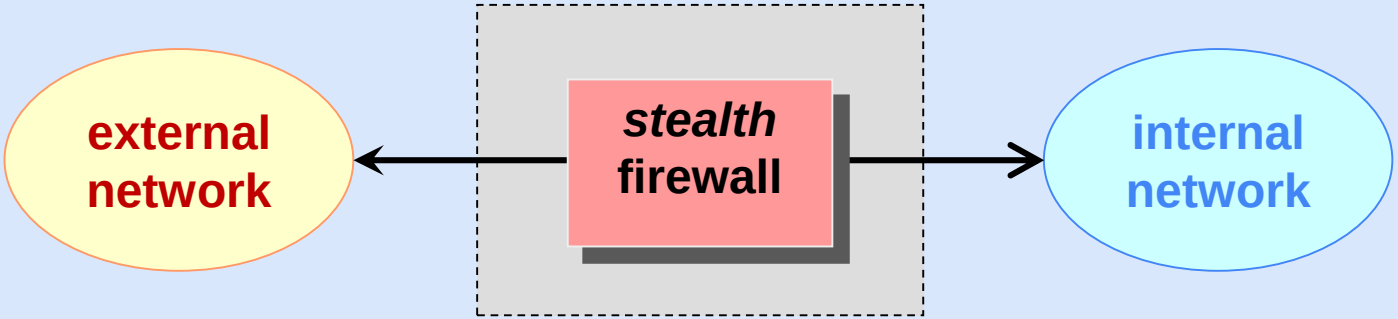
- **HTTP server acting just as a front-end for the real server(s) which the requests are passed to**
- **implements network ACL & content inspection**
- **… plus additional benefits:**
  - obfuscation (no info about the real server)
  - TLS accelerator (with unprotected back-end connections …)
  - load balancer
  - web accelerator (=cache for static content)
  - spoon feeding (gets from the server a whole dynamic page and feeds it to the client according to its speed, so unloading the application server)

Reverse proxy: possible configurations



Stealth firewall

- **firewall without an IP address, so that it cannot be directly attacked**
- **physical packet interception (by setting the interfaces in promiscuous mode)**
- **copies or discard network traffic (based upon its security policy) but does not alter it in any way**



Local / personal firewall

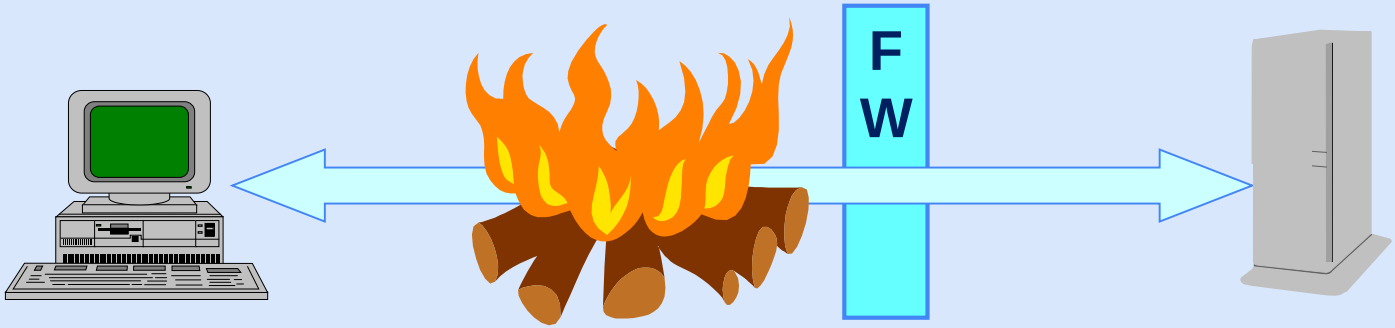
- **firewall directly installed at the node to be protected**
- **typically a packet filter**
- **w.r.t. a normal network firewall, it may limit the processes that are permitted:**
  - to open network channels towards other nodes (i.e. act as a client)
  - to answer network requests (i.e. act as a server)
- **important to limit the diffusion of malware and trojans, or plain configuration mistakes**
- **beware: in order to be effective, the firewall management must be separated from the system management**

Firewall management

- **network security policy manual:**
  - rule no. X
  - required functionality
  - temporal duration of the rule
  - requestor and implementor
  - firewall rule(s) no. Y
- **periodic (semi-)automatic control of the match between the policy and the actual firewall rules:**
  - dump + diff against ‘golden’ configuration

Protection offered by a firewall

- **a firewall is 100% effective only for attacks over/against blocked channels**
- **the other channels require other protection techniques:**
  - VPN
  - “semantic” firewall / IDS
  - application-level security



Intrusion Detection System (IDS)

- **definition:**
  - system to identify individuals using a computer or a network without authorization
  - extendable to identify authorized users violating their privileges
- **hypothesis:**
  - the behavioural “pattern” of non-authorized users differs from that of the authorized ones

IDS: functional features

- **passive IDS:**
  - cryptographic checksum (e.g. tripwire)
  - pattern matching (“attack signature”)
- **active IDS:**
  - “learning” = statistical analysis of the system behaviour
  - “monitoring” = active statistical info collection of traffic, data, sequences, actions
  - “reaction” = comparison against statistical parameters (reaction when a threshold is exceeded)

IDS: topological features

- **HIDS (host-based IDS)**
  - log analysis (OS, service or application)
  - internal OS monitoring tools
- **NIDS (network-based IDS)**
  - network traffic monitoring tools

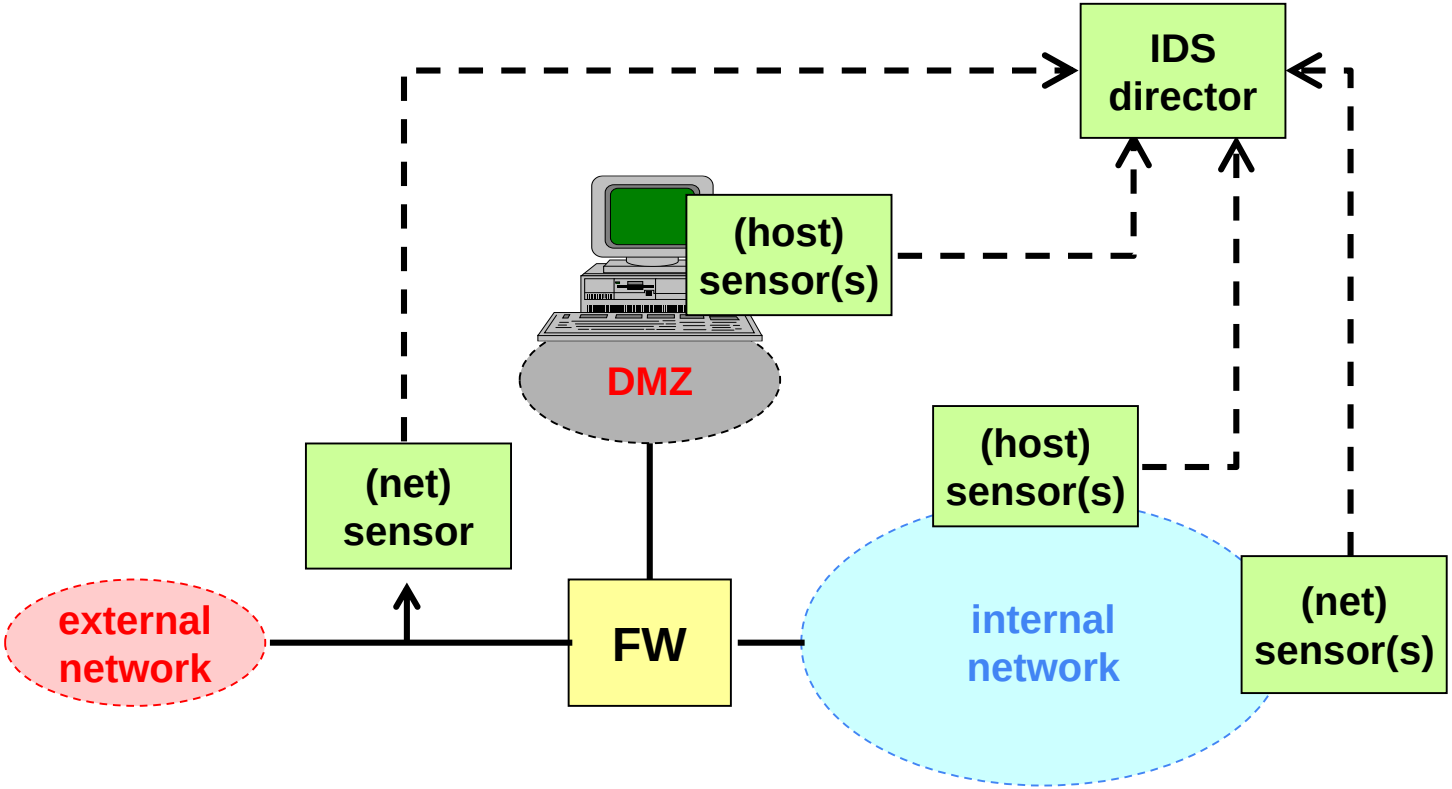
SIV and LFM

- **System Integrity Verifier**
  - checks files / filesystems looking for changes
  - e.g. changes to Windows registry, cron configuration, user privileges
  - e.g. tripwire
- **Log File Monitor**
  - checks the log files (OS and applications)
  - looks for known patterns of successful attacks or attempts
  - e.g. swatch

NIDS components

- **sensor**
  - checks traffic and logs looking for suspect patterns
  - generated the relevant security events
  - interacts with the system (ACLs, TCP reset, ...)
- **director**
  - coordinates the sensors
  - manages the security database
- **IDS message system**
  - secure and reliable communication among the IDS components

NIDS architecture



IPS

- Intrusion Prevention System
- to speed-up and automate the reaction to intrusions = IDS + distributed dynamic firewall
- a technology, not a product, with large impact on many elements of the protection system
- dangerous! may take the wrong decision and block innocent traffic
- often integrated in a single product IDPS

Unified Threat Management (UTM)

- **integration of several products in a single device**
  - UTM- or security-appliance
- **firewall, VPN, anti-malware, content-inspection, IDPS, …**
- **the actual capabilities depend upon the manufacturer**
- **mainly targeted to reduce the number of different systems, hence the management complexity and the cost**

Honey pot / Honey net