Politecnico di Milano
Dip. Elettronica e Informazione
Milano, Italy

Firewalling and secure network architectures

What's a firewall ?

- A firewall is a network access control system which verifies all of the traffic flowing through it
- Therefore, a firewall must be the single enforcement point between a screened network and outside networks
- Its main functions are usually:
  - IP packet filtering
  - Network address translation (NAT)
- A side note: it's a firewall, not a “fire wall”, and “un firewall” and not “una firewall” in Italian
  - A firewall is a wall designed to partition a building and stop fire spreading, “wall of fire” is a 4th level spell.

Firewall is not omnipowerful

- Let's repeat: a firewall checks all the traffic flowing through it, and only the traffic flowing through it
- Insider abuse = firewall is powerless (unless the network is partitioned somehow)
- Unchecked paths
  - E.g. a modem connection of a LAN
- The firewall itself is a computer: it could have vulnerabilities and be violated
  - However, usually offers few or no services, so less attack surface

Security policy (firewall rules)

- A firewall is a stupid bouncer at the door
  - Just applies rules
  - Bad rules = no protection
- The rules of the firewall must correspond to a higher-level security policy
  - E.g. "I want no clients to be able to download email from external email servers!"
- The policy must be built on a default deny base
  - Everything is forbidden, except what is explicitly allowed
- We will look at some examples, but now we need to understand how to translate high level policies into real rules; and we must understand the technologies

Politecnico di Milano
Dip. Elettronica e Informazione
Milano, Italy

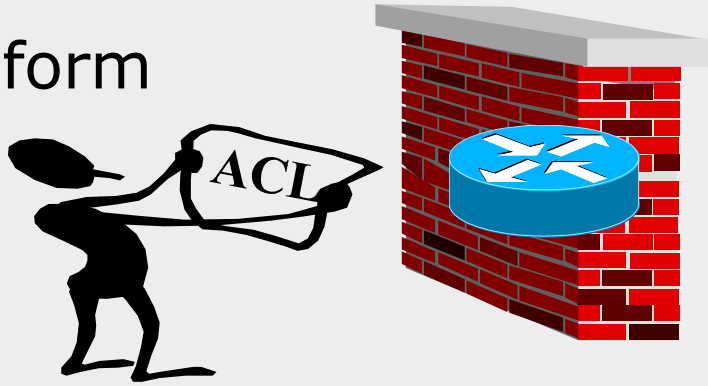
Firewall technologies

Firewall taxonomies

- Network layer firewalls
  - Packet Filters
  - Stateful Packet Filters (amidst network and transport)
- Application layer Firewalls
  - Circuit level firewalls (amidst transport and application)
  - Application proxies

Packet Filters

- Packet by packet processing
- Uses header info for filtering
  - Src and dst address
  - Src and dst ports
  - Protocol type
  - IP options
- Cannot track connections across packets
- Cannot examine content, except packet by packet
- Many routers implement a form of packet filtering



What can we express with these rules

- Based on addresses, we can open or close traffic from and to specific sources
- Based on port numbers we can block or allow known services (e.g. port 25 for SMTP)
- I can completely block a protocol (e.g. ICMP)
- Discard on some options (e.g. source routing)
- We can use some creativity
  - If packet comes in from external interface and has a src address apparently from the IP network I can drop it as spoofed (spoofing)
  - If packet comes from a trusted network I can let it in
  - Are these good ideas?

Stateful (Dynamic) Packet Filtering

- Same as previous, plus...
- Keeping track of the state of the TCP connection
  - e.g. after a SYN packet, a SYN-ACK must follow, with specific field values; any other packet is unacceptable
  - I can track connections, and allow through packets; whereas with a packet filter I needed to add a response rule!
- Better expressivity
- Performance and load become connection based and not just packet based
  - Parallel connections are just as important as packets-per-second

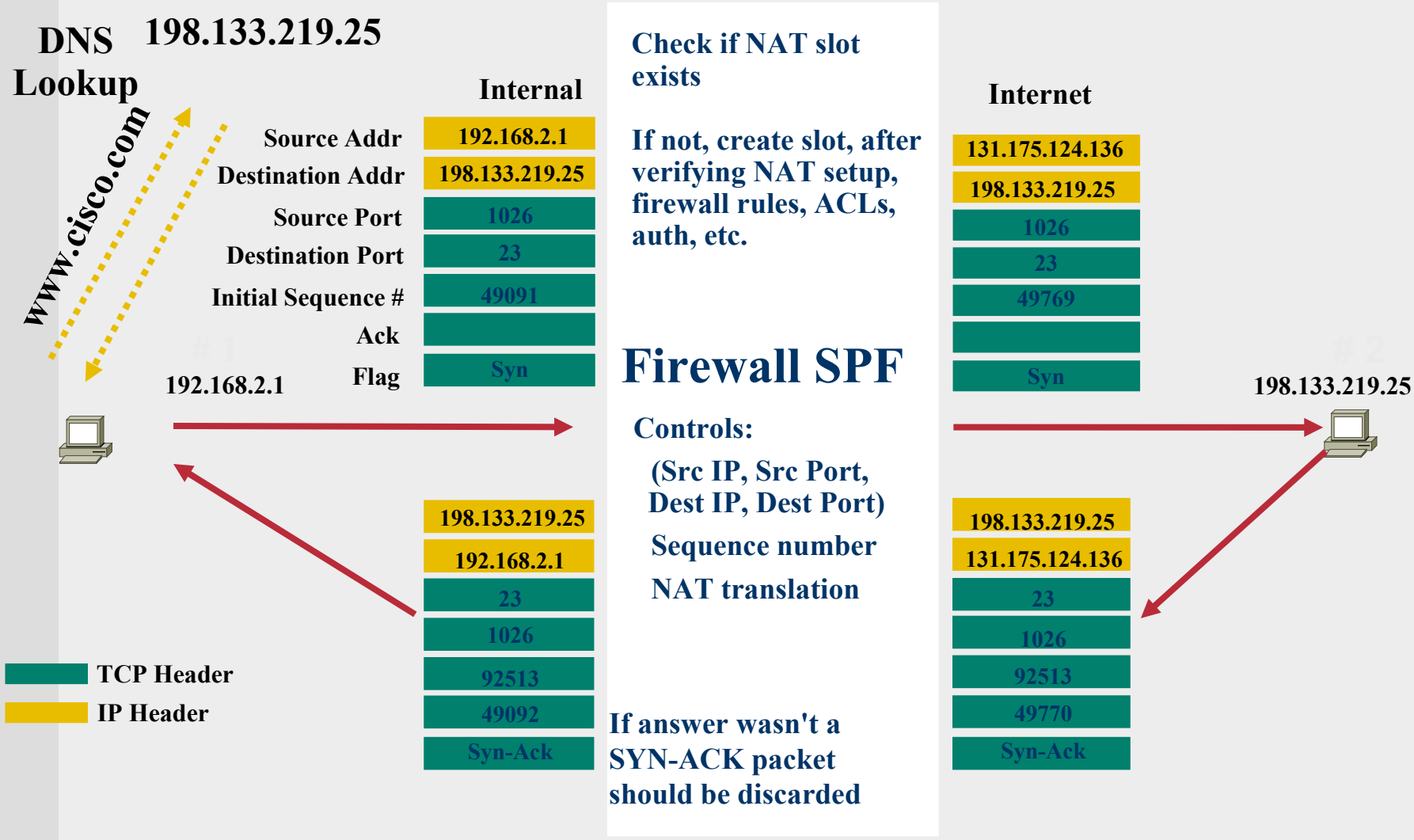
A few other benefits and capabilities

- Content inspection can reach application layer
  - Can reconstruct application-layer protocols such as HTTP
  - Can perform application-layer filtering, e.g. ActiveX content in HTTP connections
- Can perform logging and accounting on connections
- Can perform Network Address Translation (NAT)
- Defragmenting and reassembling packet (helps avoiding pathological fragmented packets)

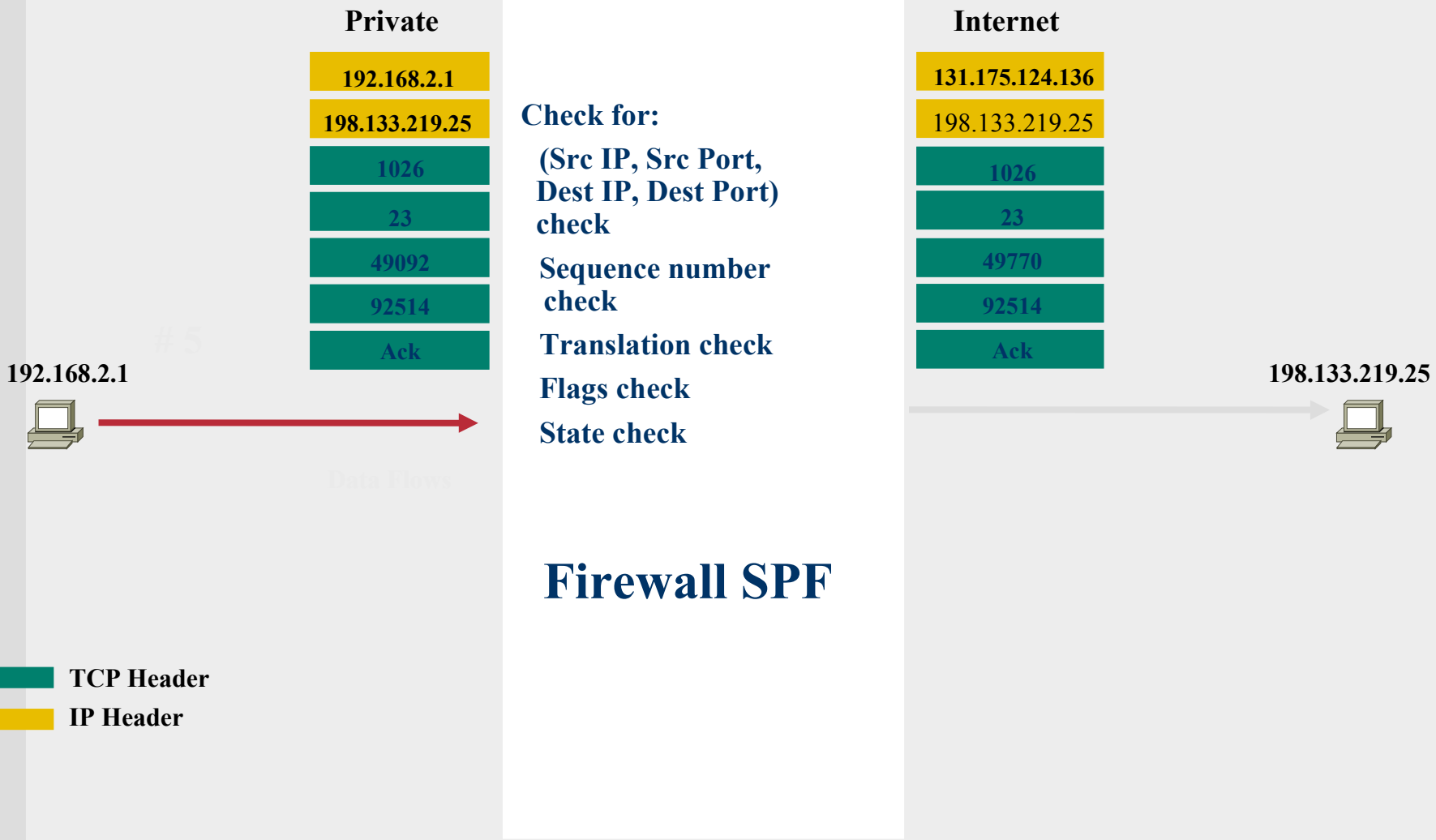
Handling sessions

- A “session” can be roughly approximated as “any atomic data exchange” between two hosts over the Internet
- Two main protocols
  - TCP (Transmission Control Protocol)
  - UDP (User Datagram Protocol)
- For TCP a session almost maps to a single “connection”, while in UDP no such concept
- Session handling is particularly important for NAT translations

NAT session initialization



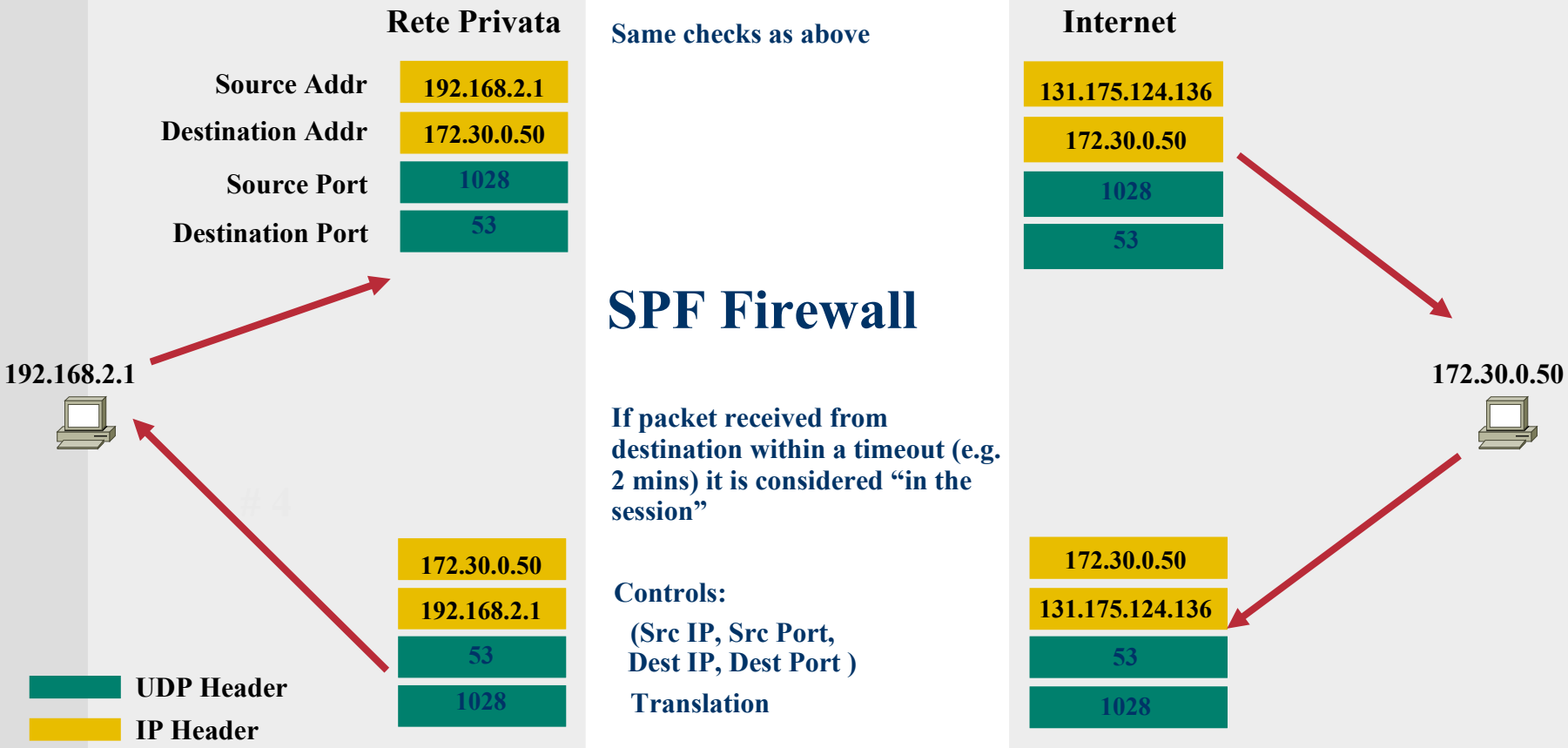
Connection goes on



What about UDP ?

- Connectionless
- Used in several services, so we can't dismiss it
  - E.g. DNS
  - Performance-based services: VoIP H.323, streaming video
- Difficult to secure and handle, because there are no connections
- “session” concept exists, and it can be used for NAT and controls

UDP and NAT

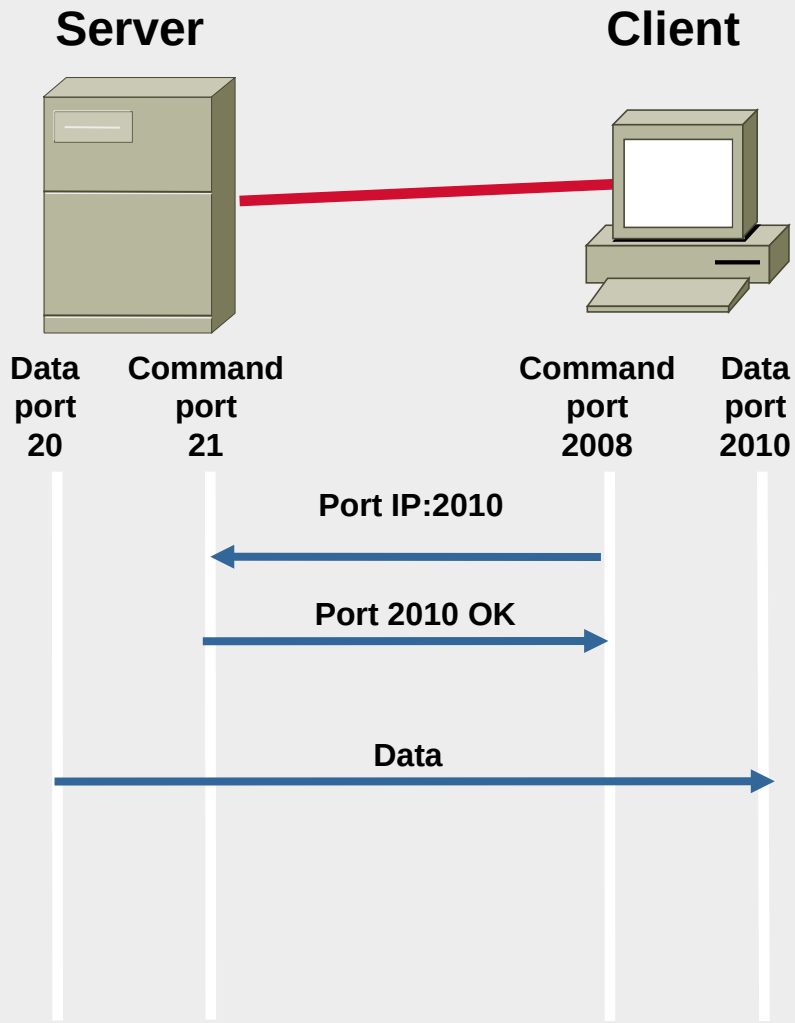

Rete Privata
Source Addr
Destination Addr
Source Port
Destination Port
192.168.2.1
172.30.0.50
1028
53
Same checks as above
SPF Firewall
If packet received from destination within a timeout (e.g. 2 mins) it is considered "in the session"
Controls:
(Src IP, Src Port, Dest IP, Dest Port)
Translation
Internet
131.175.124.136
172.30.0.50
1028
53
192.168.2.1
172.30.0.50
172.30.0.50
192.168.2.1
53
1028
172.30.0.50
131.175.124.136
53
1028
UDP Header
IP Header

Application-layer inspection for NAT

- Some weird protocols (e.g. NAT, DCC, some instant messengers) transmit network layer info at application layer
- E.g., FTP uses dynamic connections
  - Allocated for file uploads, downloads, output of commands
  - "PORT" syntax is used to control such channels
  - I need to substitute parameters in PORT command, inside the application-layer data

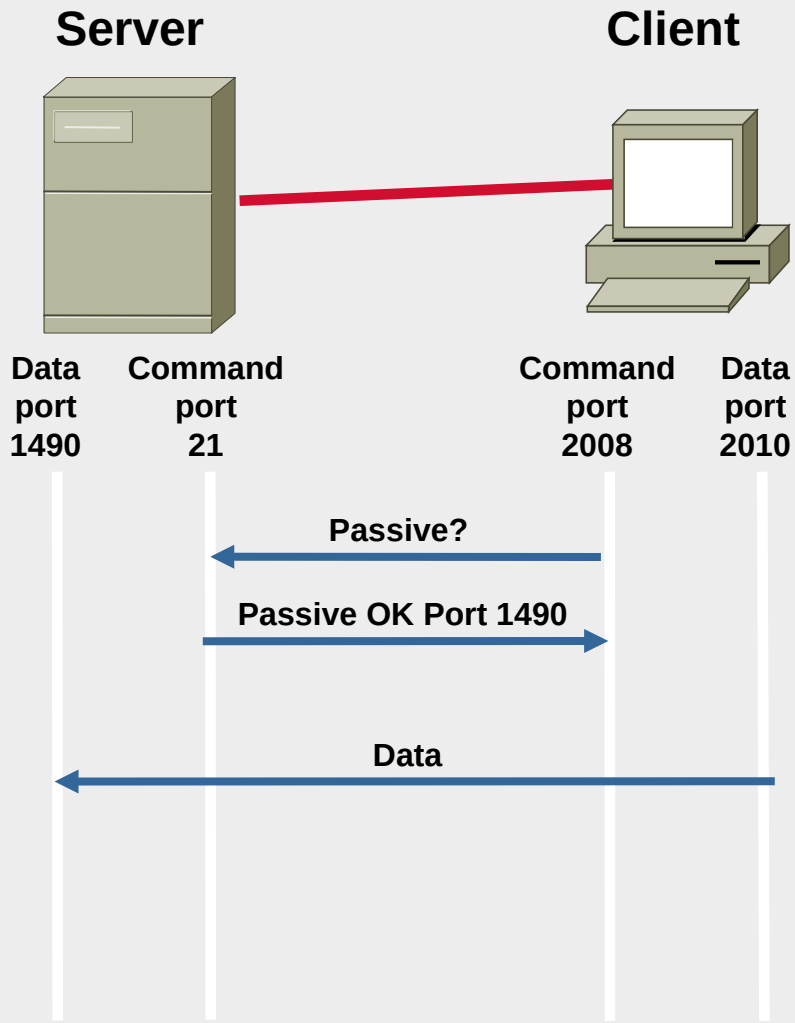
FTP standard mode

- Two channels
  - Client-initiated command channel
  - Server-initiated data channel
- If the firewall protects the client
  - Must allow outbound connections to port 21
  - Must inspect app layer to open inbound data connection port
  - If NAT is used, must detect app layer address, change it, open port and map it temporarily
- If the firewall protects the server
  - Port 21 must be open inbound
  - If server can initiate outbound connections, no further rules needed, otherwise need temporary outbound connection rule via app layer inspection



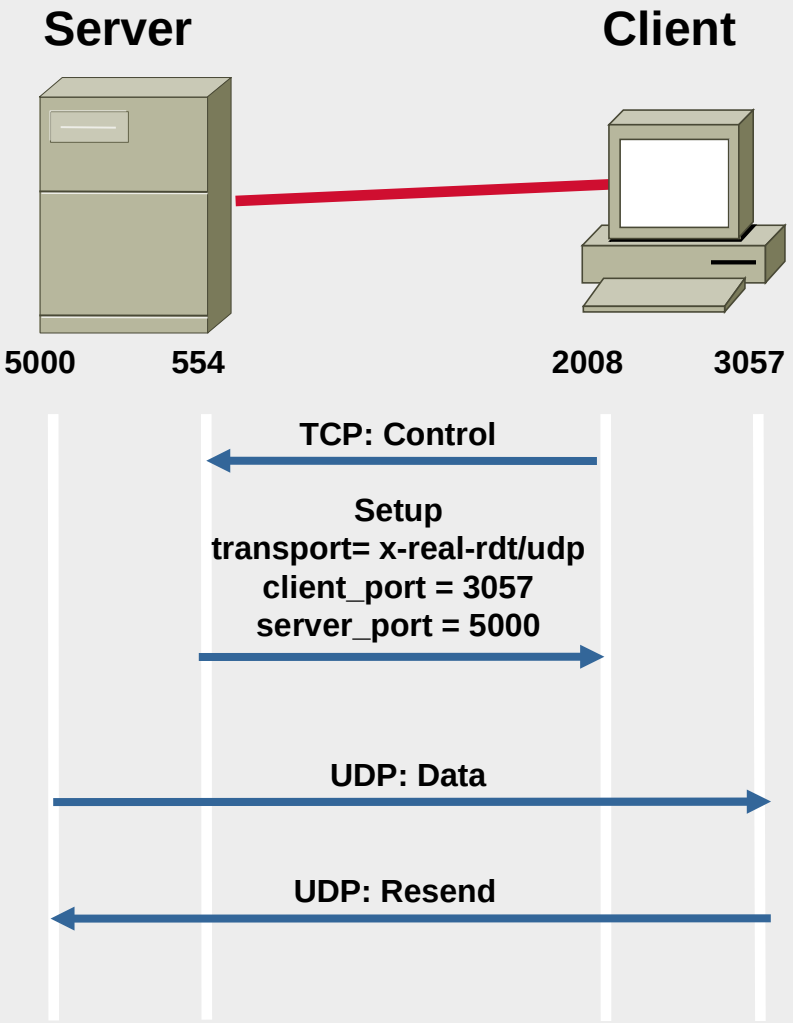
FTP passive mode

- Both channels are client-initiated (yeah!)
- If the firewall protects the client
  - If outbound connections are allowed, no further rules needed
  - Otherwise, applayer inspection to temporarily open outbound port
- If the firewall protects the server
  - Need applayer inspection to temporarily open and map the data port



RealNetworks RDT Mode

- Threeeee channels, ladies 'n gentlemen!
  - Control connection (TCP)
  - UDP data (UDP)
  - UDP resend (UDP)
- If the firewall protects the client
  - If outbound traffic is allowed: need to open port for UDP data
  - Otherwise, need also to open port for UDP resend and TCP control
- If the firewall protects the server
  - If outbound traffic is allowed: need to open port for TCP control, and a temp port for UDP resend
  - Otherwise, need also a temp port for UDP data



Deep Inspection / Intrusion Prevention

- Modern packet filters can go even more in depth in the analysis of sessions and protocols
  - E.g. recognize MIME multipart attachments in SMTP flows and send data to antiviruses!
- Intrusion Prevention
  - Add a set of "known attack packets" that the firewall can drop based on signatures
  - Update problems
  - Zero-days
- We'll see more in detail talking about IDS

Politecnico di Milano
Dip. Elettronica e Informazione
Milano, Italy

Application Layer Firewalls

Circuit Firewall

Application/Proxy Firewall

Circuit Firewall

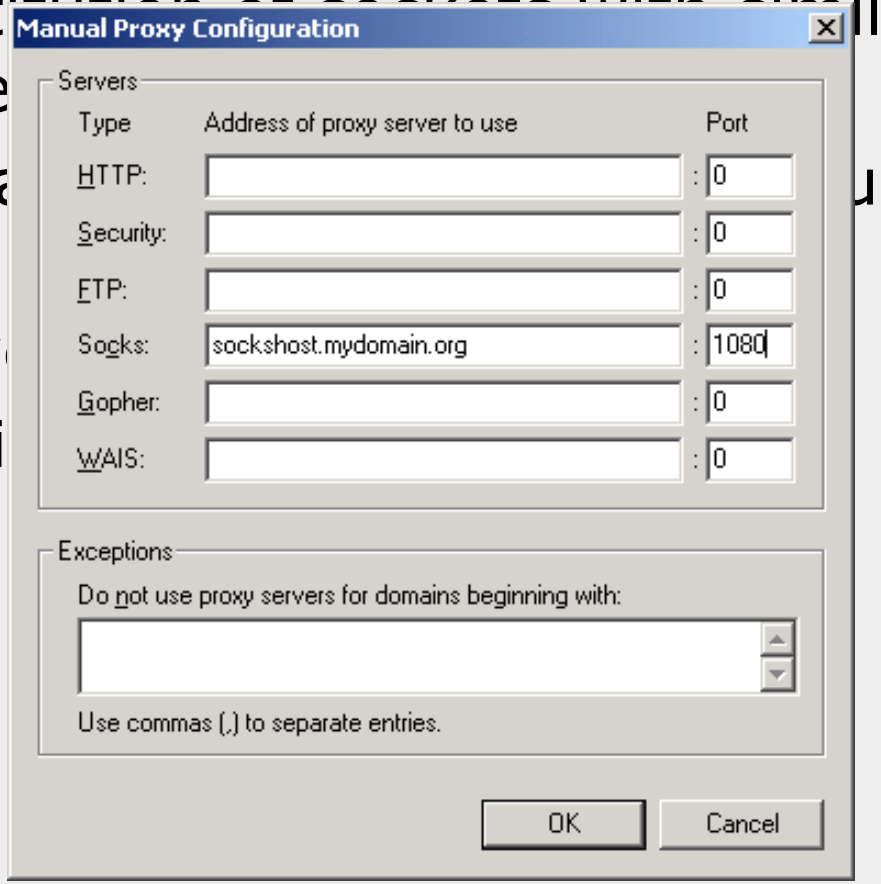
- Relays TCP connections
- Legacy
- Client connects to a specific TCP port on the fw, which then connects to the address and port of the desired server
- In general, no deep inspection when they were created
- Creating the virtual circuit on behalf of the client it can check the handshake and then forward data
- Packets can be either:
  - Connection requests, or
  - Packets belonging to an established connection

How is it implemented

- The firewall uses a connection table with:
  - Connection src address
  - Connection dst address
  - State (handshake, established, closing)
  - ACK numbers
  - Physical interface for ingress and egress
- Data can pass through only if they belong to a valid connection
- When connection is dropped, entry is removed

Cons

- Requires modification to applications
- Substitution of sockets with similar calls which connect
- Firewalls must be configured on each client
- IP level
- Only i



http://www.socks.nec.com/socksfaq.html

Application Proxy Firewall

- Proxy: acts as client towards server; acts as server towards client
- Validates the protocol/application level data!
- Almost never transparent to users or applications
  - May require modifications
  - Each protocol needs its own proxy server
- May authenticate users, apply specific filtering policies, perform advanced logging, perform content filtering/virus scanning
- Gives the user access to a subset of the server functions...
- This means that it can be used both
  - To defend clients, or
  - To defend servers (“reverse proxy”)
- Usually implemented on COTS OSs.

Un esempio di proxy HTTP

www.cisco.com 198.133.219.25

GET http://www.cisco.com/index.html

DNS Lookup

Proxy Client

Application Protocol Analysis

Proxy Server

Public Network

GET /index.html

HTTP: ----- Hypertext Transfer Protocol -----

HTTP:

HTTP: Line 1: HTTP/1.0 302 Found

HTTP: Line 2: Server: Netscape-Enterprise/2.01

HTTP: Line 3: Date: Tue, 08 May 2001 20:52:20 GMT

Pros and cons of proxies

- Pro
  - Logging
  - Caching
  - Authentication/authorization/policies
  - Masking internal network (besides NAT)
  - Content filtering
  - Protection for weak apps
- Cons
  - New or custom services need custom proxies
  - Not transparent to clients
  - General OS: needs to be defended from lower level attacks (e.g. by a packet filter!)

Politecnico di Milano
Dip. Elettronica e Informazione
Milano, Italy

Dual-zone architecture

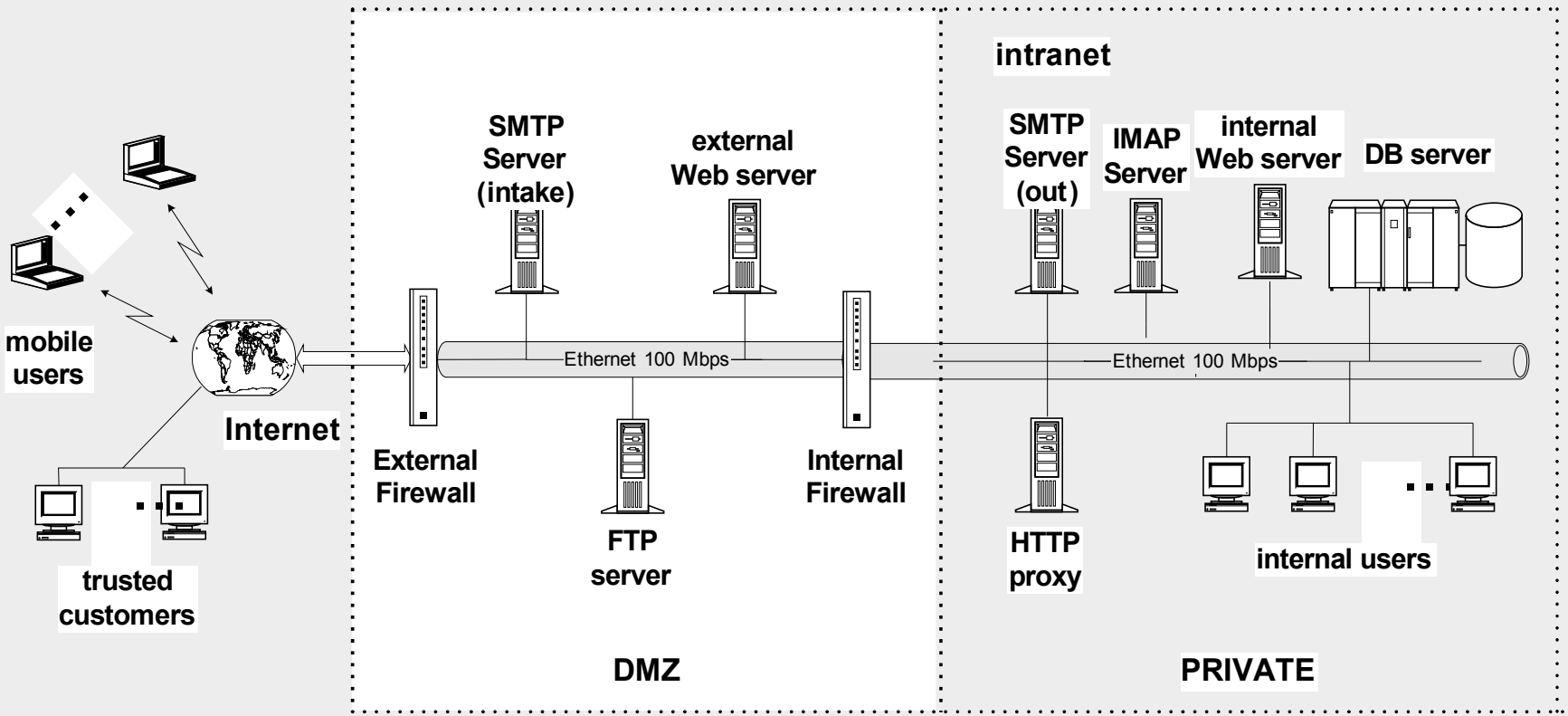
Ensuring access to the internal network

- In most cases, the perimeter defense works on the assumption that what is “good” is inside, and what's outside should be kept outside if possible
- There are two counterexamples
  - Access to resources from remote (i.e. to a web server, to FTP, mail transfer)
  - Access from remote users to the corporate network
- We will see the solutions
  - Dual zone architecture
  - Virtual Private Network (VPN)

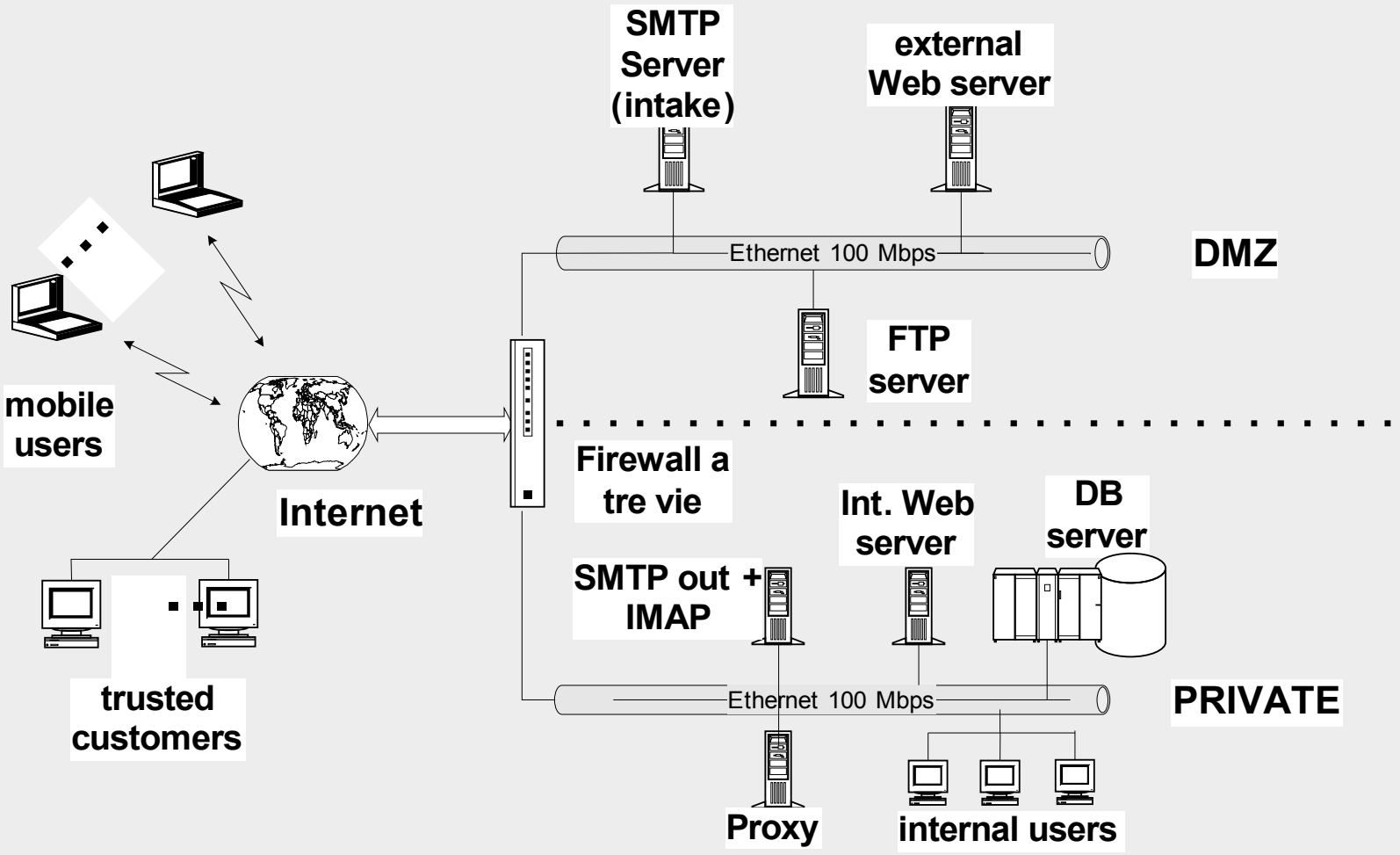
Dual zone architecture

- Issue: if we mix externally accessible servers with clients, we lower the security of the internal net
- Base idea: we allow external access to the accessible servers, but not to the internal network
- Creation of a semi-public zone called DMZ (demilitarized zone)
- The DMZ will host the public servers (web, FTP, public DNS server, intake SMTP)
- On the DMZ no critical or irreplaceable data
- We will represent "theoretically" the DMZ as contained between two dual-homed firewalls, but the real (and equivalent) implementation is with a single multihomed firewall
- Assumption during design: DMZ is almost as risky as the Internet

Dual zone architecture logical example



Dual zone architecture real example



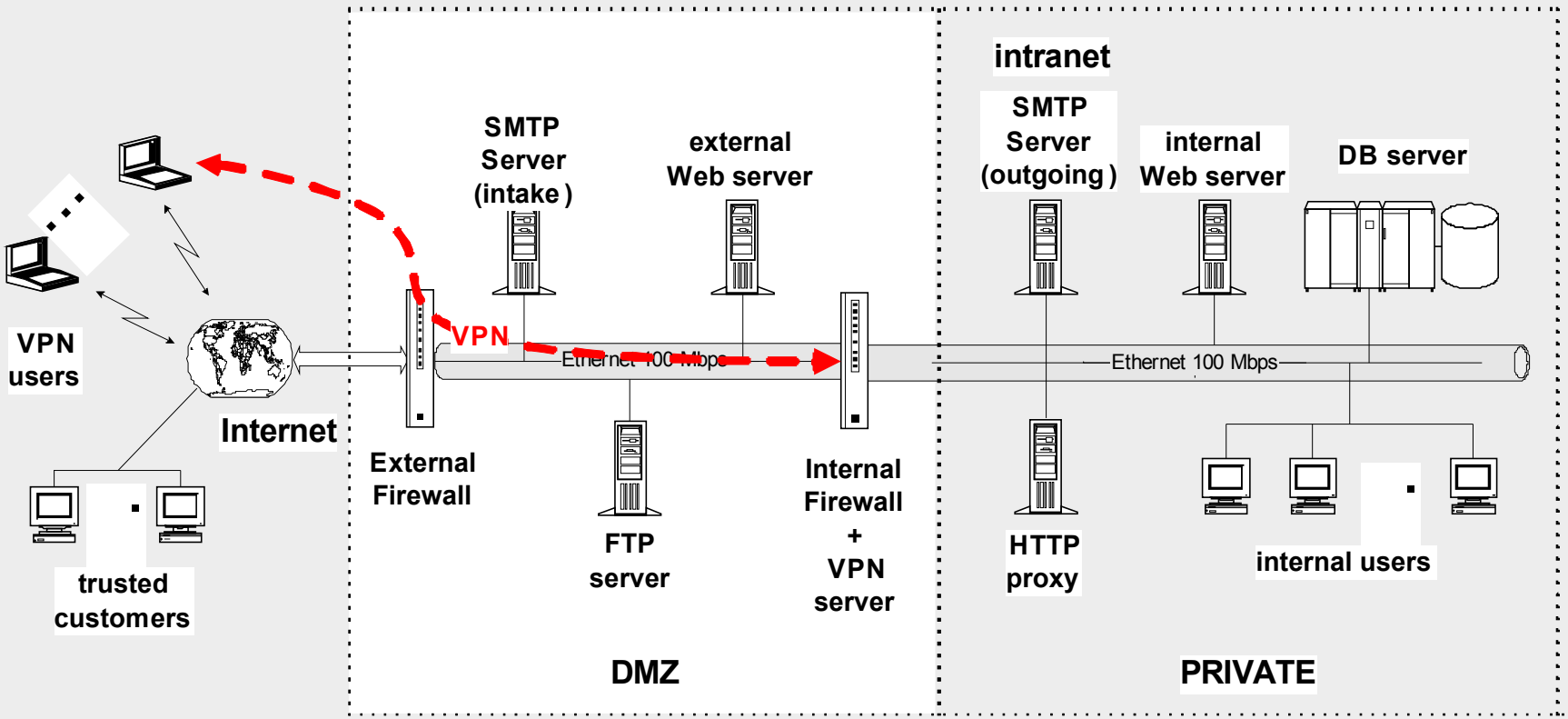
Politecnico di Milano
Dip. Elettronica e Informazione
Milano, Italy

Virtual Private Networks

What is a VPN?

- Needs:
  1. Road-warriors need to work "as if" they were in the office, accessing resources on the internal zone
  2. Connecting remote sites without using leased lines
- Which means: need to ensure confidentiality and integrity + authentication to data transmitted over a public network
- VPN, Virtual Private Network, an encrypted overlay connection over a public network
- Many different technologies, but always the same basic idea

VPN interacting with dual-zone arch



Two possible policies for VPN

- Full tunnelling
  - Every packet goes through the tunnel
  - Traffic multiplication
  - Single point of control and application of all security policies as if the client were in the corporate network
- Split tunnelling
  - Traffic to the corp network: in VPN; traffic to the Internet: directly to ISP
  - More efficient, less control
  - Just similar to the case of the PC connected via modem to the Internet

Technologies

- PPTP
  - Point-to-point tunnelling protocol, a proprietary Microsoft protocol
  - A variant of the PPP protocol with authentication and cryptography
- VPN over SSL / Tunnel SSH
  - We will see the SSL protocol in detail
- **IPSEC**
  - Security extensions of IPv6, backported to IPv4
  - Authentication and cryptography at IP layer

IPSec: keywords

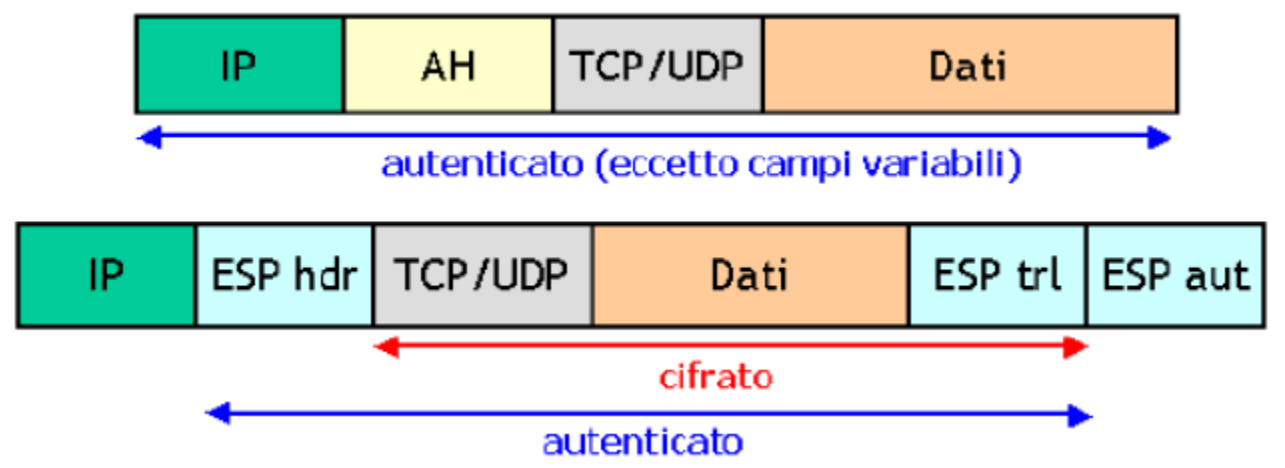
- **AH** (Authentication Header)
  - Authentication + Integrity
- **ESP** (Encapsulating Security Payload)
  - Authentication + Integrity + Confidentiality
- **IKE** (Internet KeyExchange) + **ISAKMP** (Internet Security Association and Key Management Protocol)
  - Security parameter agreement, definition of security associations, generation and renewal of master keys

Security Associations

- Security Association (SA) define the IPSec parameters
- A single SA is a unidirectional connection which defines services used by the traffic flowing through
- A single SA can adopt **either** AH **or** ESP, but not both, and either tunnelling or transport mode (see in the following)
- To secure a full-duplex connection between two point, we need two symmetric SA, one in each direction

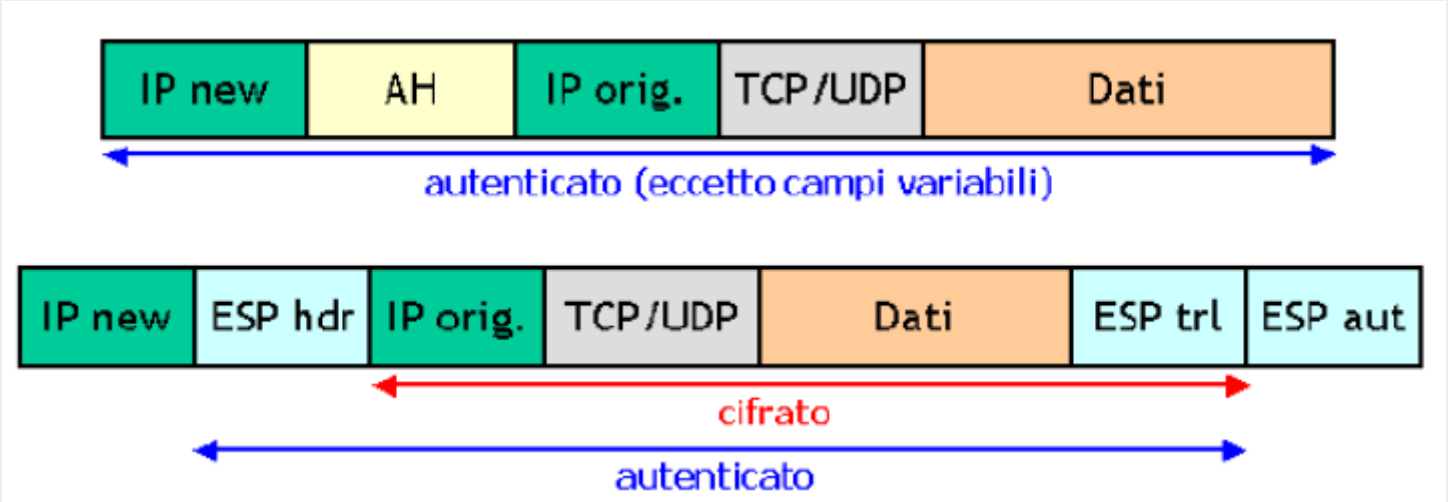
IPSec: transport mode

- SA which can work only between two hosts, not between security gateways
- AH or ESP header inserted between IP header and transport header. IP header is directly authenticated except for TTL, and thus this protocol cannot pass through NAT



IPSec: tunnel mode

- SA can work between security gateways (or hosts)
- The whole IP packet is encapsulated in a new packet
- Can traverse NAT



IPSec: basic protocol structure

- Phase I:
  - Choice of encryption protocols and algorithms; choice between modes (AH / ESP, tunnelling, etc)
  - Master Key exchange (usually Diffie-Hellman), to derive encryption keys
  - Identity verification (e.g. through certificates)
- Phase II:
  - Encryption Protocol setup
  - Subkeys generation
  - Key re-generation