La Sapienza
Università degli Studi di Roma
Dipartimento di Informatica e Sistemistica

Computer Networks II
NAT - VPN - FIREWALLS

Luca Becchetti
Luca.Becchetti@dis.uniroma1.it

A.A. 2009/2010

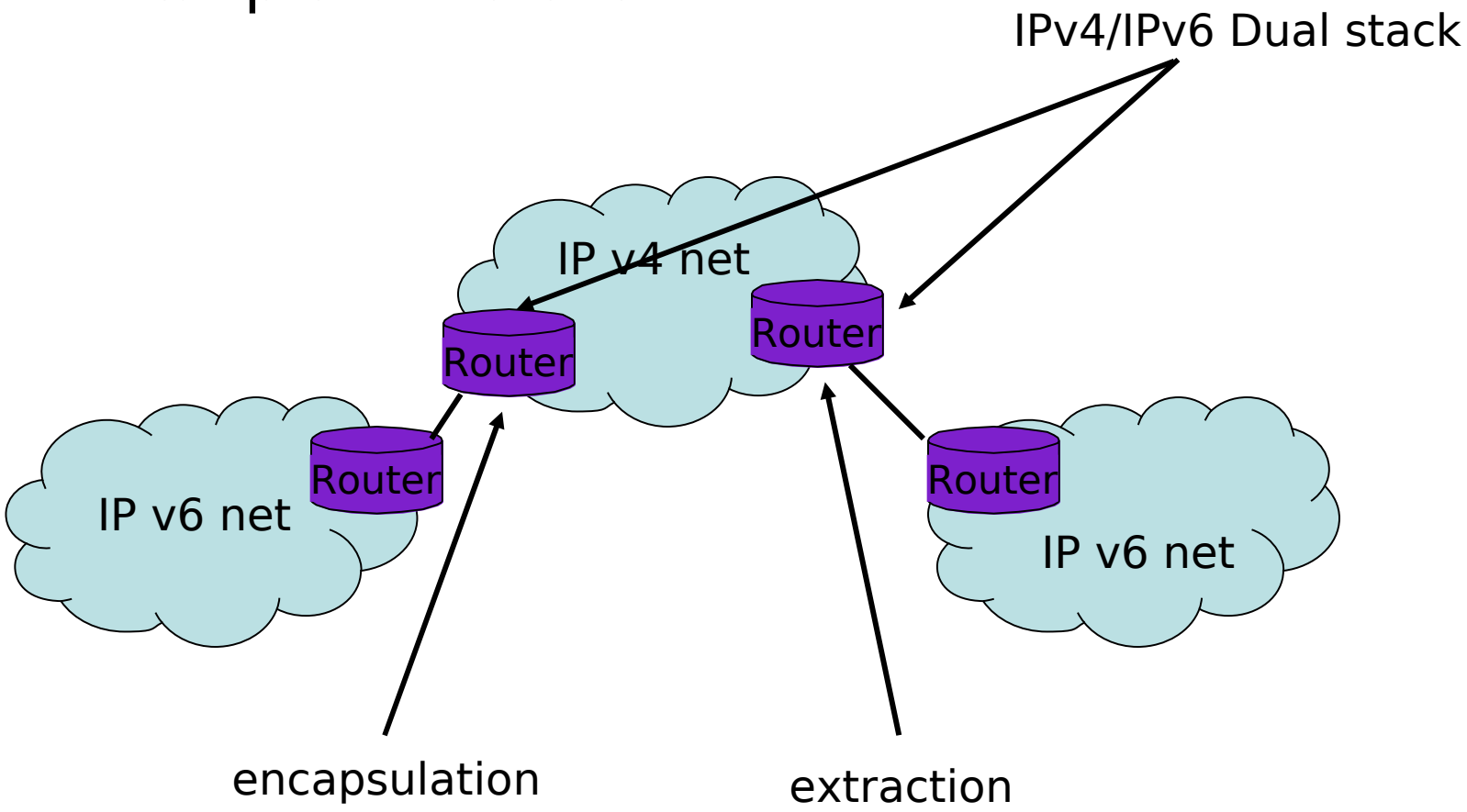
Overview

- Tunneling
  - Generic Route Encapsulation
  - Virtual Private Networks (VPN)
- Private addressing
- **N**etwork **A**ddress **T**ranslation
  - Private and public addresses
  - Main requirements
    - Reuse of IP addresses
    - Protection of subnets against external attacks
- Application gateway e firewall

Tunneling

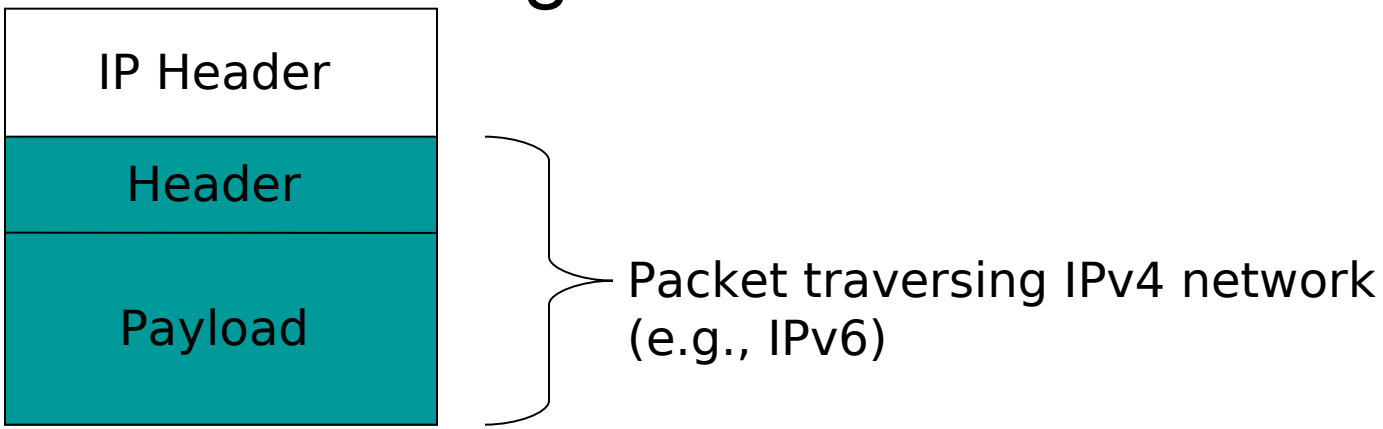
Tunneling

- Example: IPv6 over IPv4



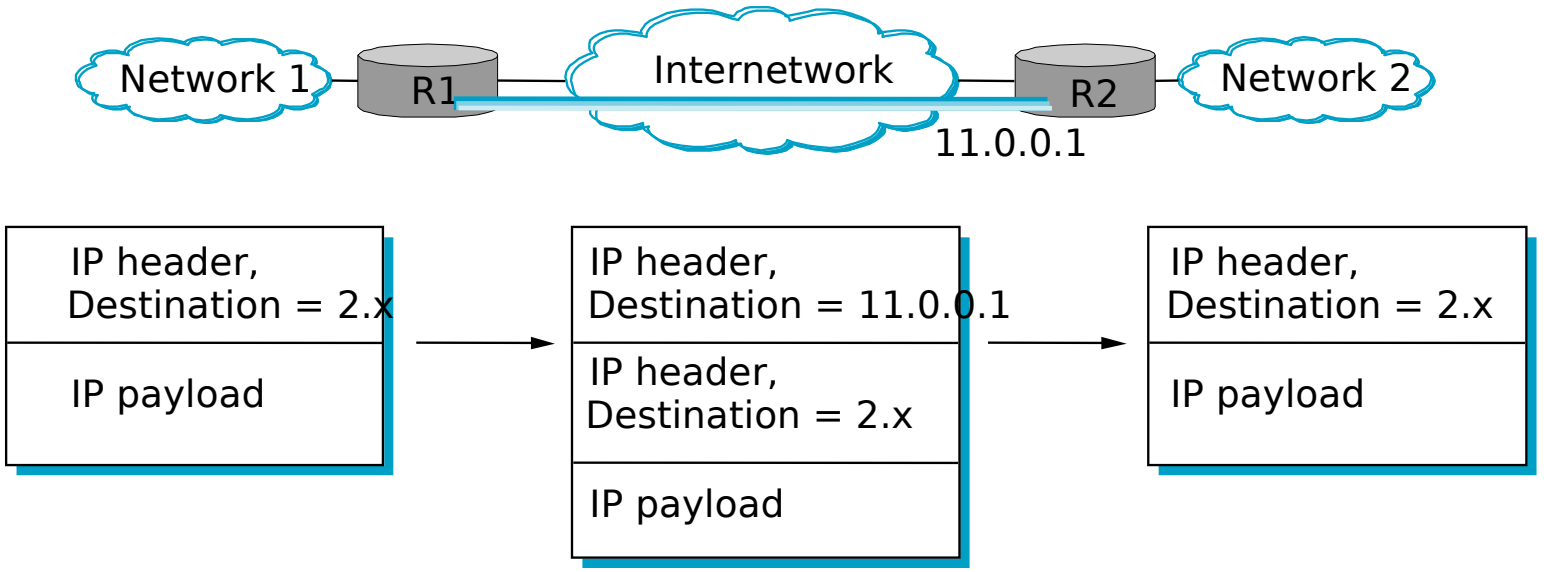
Tunneling

- General technique
- For IP networks: technique to carry non IP (or special purpose IP traffic - see further) traffic over an IP
- Main tools: encapsulation of non IP packets within IP datagrams



IP tunneling: example 1/3

- IP Tunnel over IP: allow a point-to-point virtual link [VC] between arbitrarily far apart nodes
- In the example: R1 and R2 are end points of IP tunnel

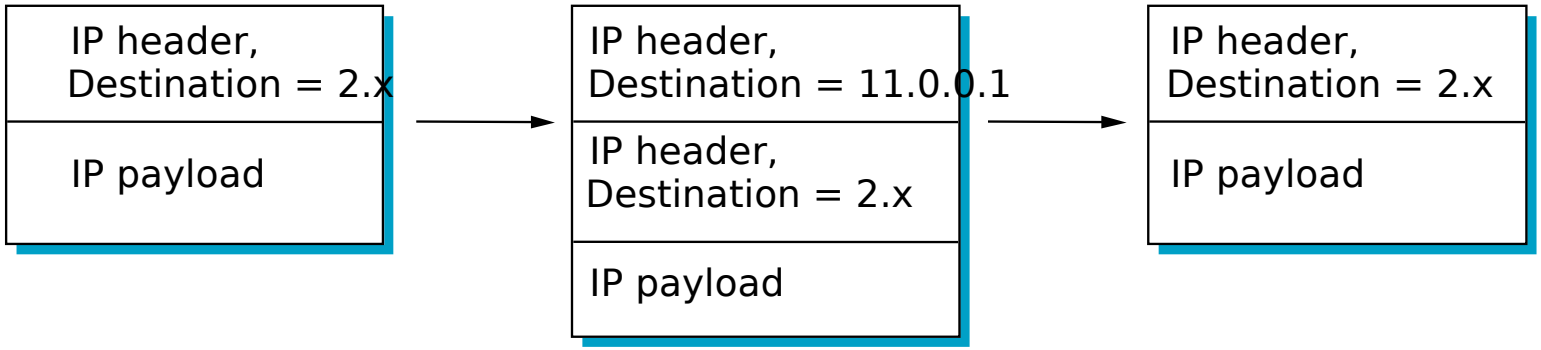


IP tunneling: example 2/3

RT at R1

NetNum	NextHop
1	Intf0
2	VIntf1
Default	Intf1

- Intf1 connects to Internetwork
- R1 encapsulates packets directed to virtual link within IP datagrams towards R2

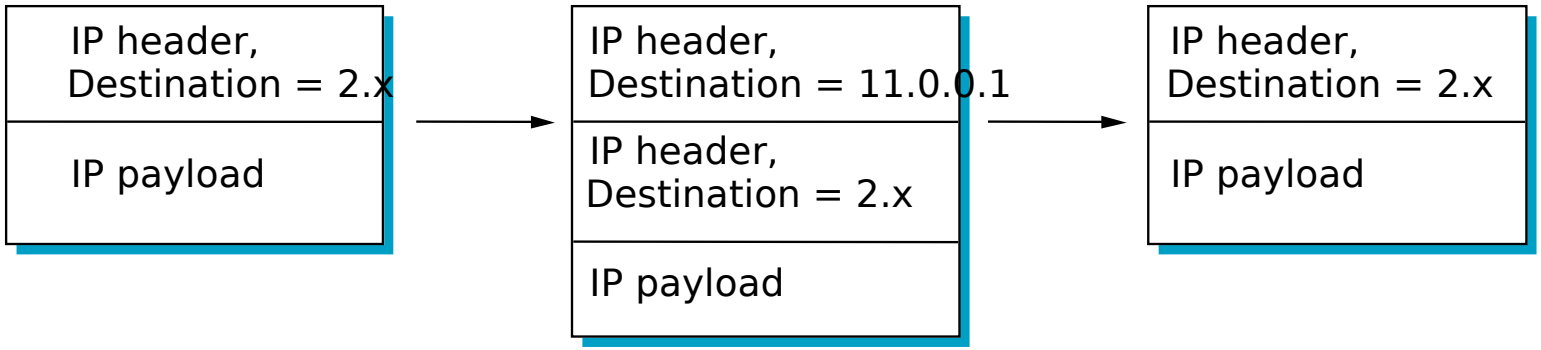


IP tunneling: example 3/3

RT di R1

NetNum	NextHop
1	Intf0
2	VIntf1
Default	Intf1

- R2 extracts encaspulated packet and “sees” that it si directed at network 2
- R2 delivers packet to destination



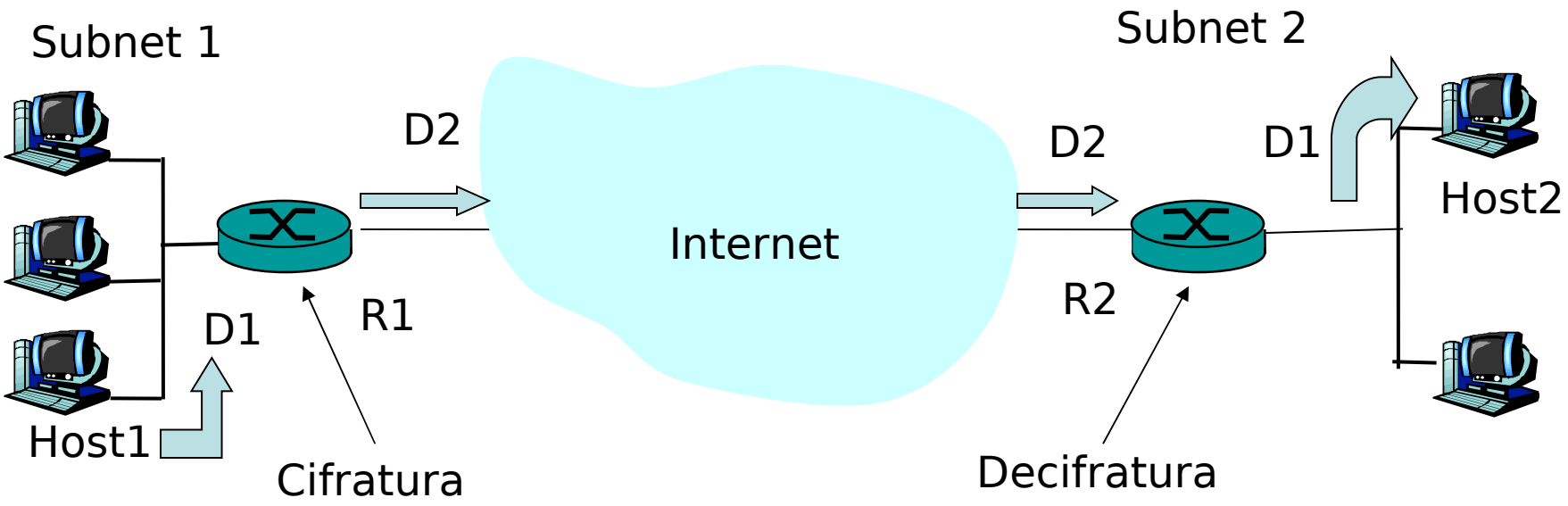
Why IP tunneling

- security
  - Tunneling + encription allow safe connectivity over public networks
    - Solution often used in practice
    - Corresponds to most common definition of VPN
- Link to routers with capabilities not available elsewhere in the network
  - E.g., Multicast
- Traversing of IP segments by non IP traffic
  - E.g.: IPv6
- Mobile IP

Cryptography and tunneling

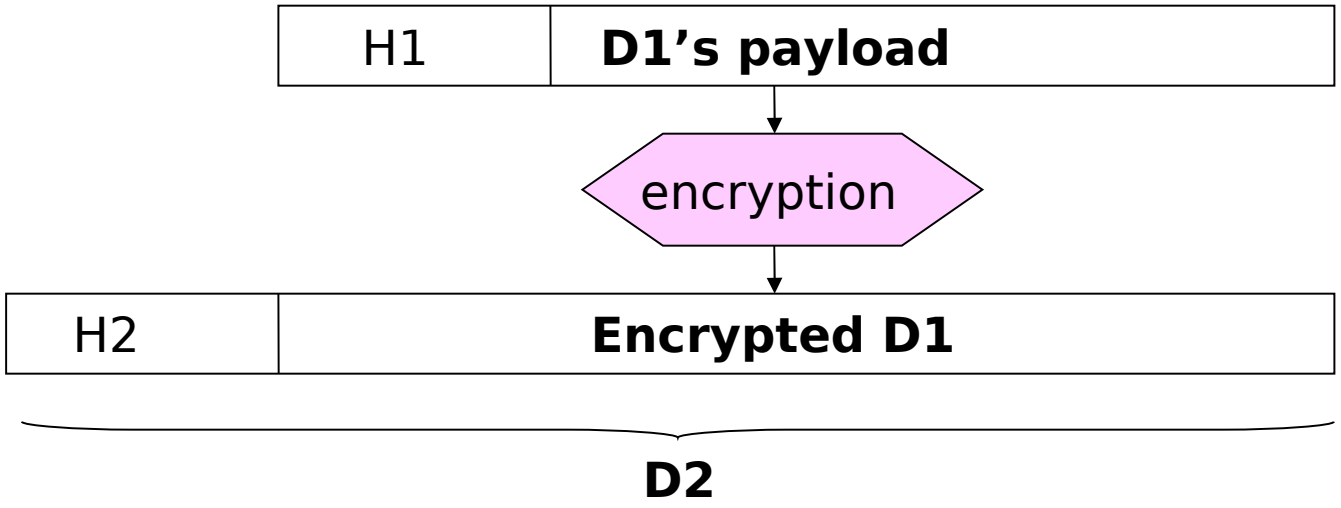
- Cryptography
  - Datagrams encrypted at source and decrypted at destination
- Tunneling
  - Encrypted datagrams encapsulated within standard IP datagrams as payload
- Only visible part: external datagram's header
  - Payload containes oryginal, encrypted packet
  - Source and destination IP addresses (of original packet) encrypted

E.g.: D'gram from Host1 to Host2



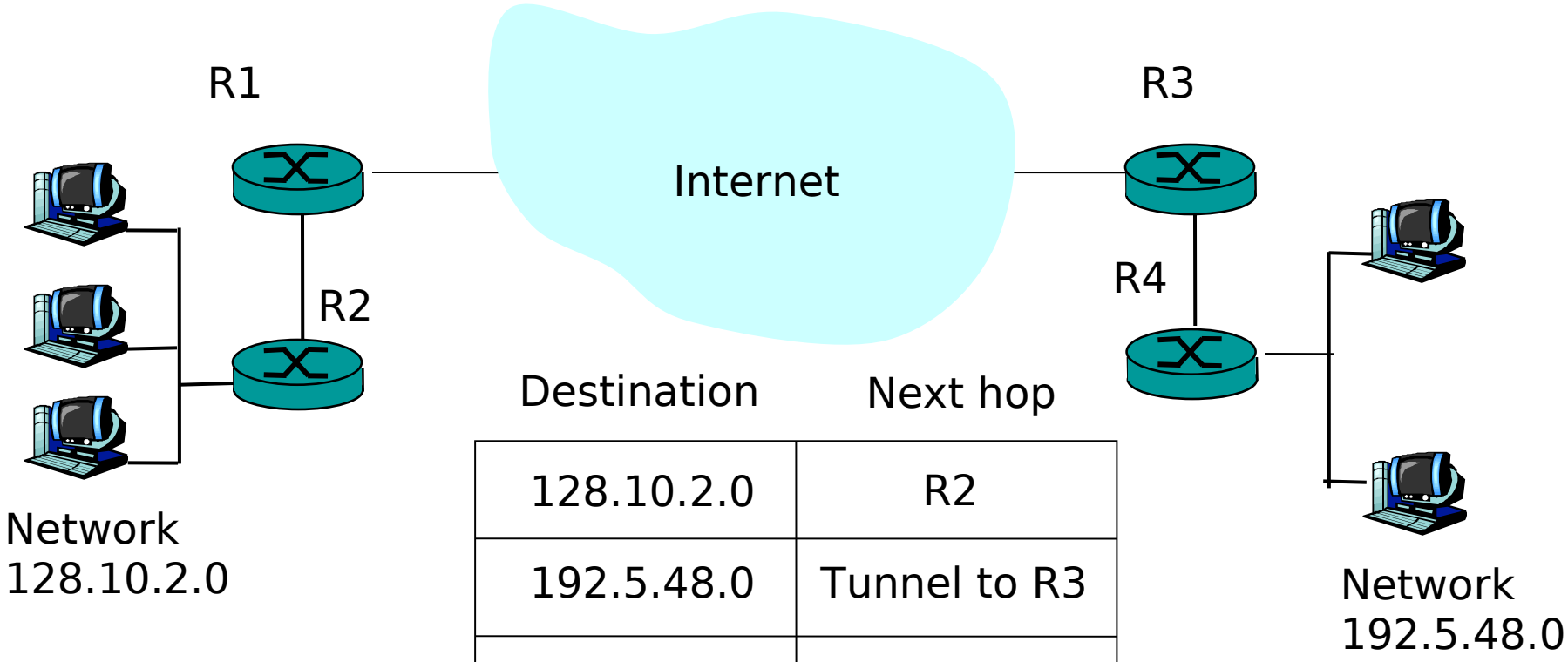
- Datagram D1 from Host1 to Host2
- D1 encrypted at R1 $\Rightarrow$ D2
- D2 reaches R2 and is decrypted $\Rightarrow$ D1
- D1 is delivered to Host2

Example - cont.



- H2 not encrypted
- IP addresses in H2 are those of R1 and R2 respectively
- Addresses of Host1 and Host2 in H1 [hence encrypted]

E.g.: R1's routing table

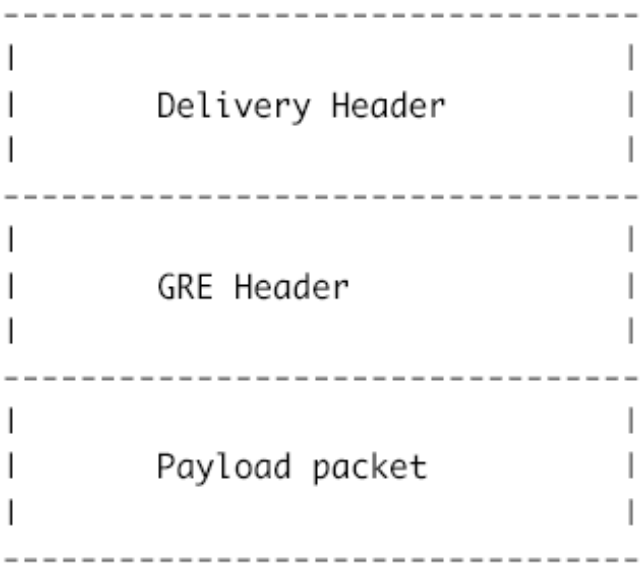


Destination	Next hop
128.10.2.0	R2
192.5.48.0	Tunnel to R3
default	ISP Router

- Default entry indicates that Datagram is sent to the Internet without encryption and tunneling

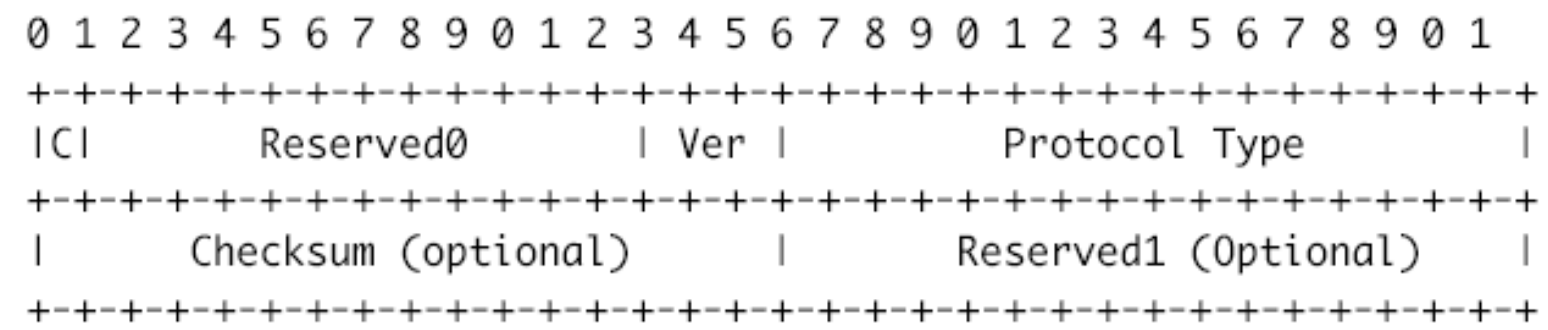
Case study - GRE

- Generic Routing Encapsulation (Cisco)
  - RFC 1701 (first version)
  - RFC 2784 and 2890
- “This document specifies a protocol for encapsulation of an arbitrary network layer protocol over another arbitrary network layer protocol.” - RFC 2784



GRE/cont.

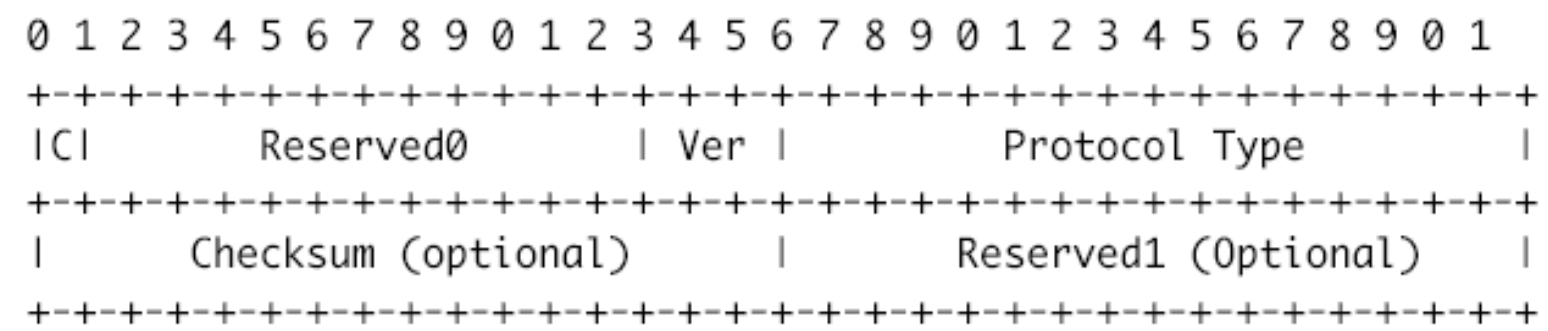
The GRE packet header has the form:



- Bit C: if 1 --> Checksum and Reserved1 fields are valid
- Reserved0: bit 1-5 used if receiver implements RFC 1701
- Ver: must be 0

GRE/cont.

The GRE packet header has the form:



- Protocol type: identifies protocol of encapsulated packet
- Checksum: Internet checksum over GRE header and payload (excluding checksum itself)
  - Present if C = 1
- Reserved1
  - Present if C = 1
  - Used in RFC 1701 compliant implementations

Payload - IPv4

- From RFC 2784
  - “When a tunnel endpoint decapsulates a GRE packet which has an IPv4 packet as the payload, the destination address in the IPv4 payload packet header MUST be used to forward the packet and the TTL of the payload packet MUST be decremented. Care should be taken when forwarding such a packet, since if the destination address of the payload packet is the encapsulator of the packet (i.e., the other end of the tunnel), looping can occur. In this case, the packet MUST be discarded.”

- IPv4 over IPv4 obviously possible. Example: VPN’s using IP tunneling
  - Example: RFC 1701 compliant tunneling

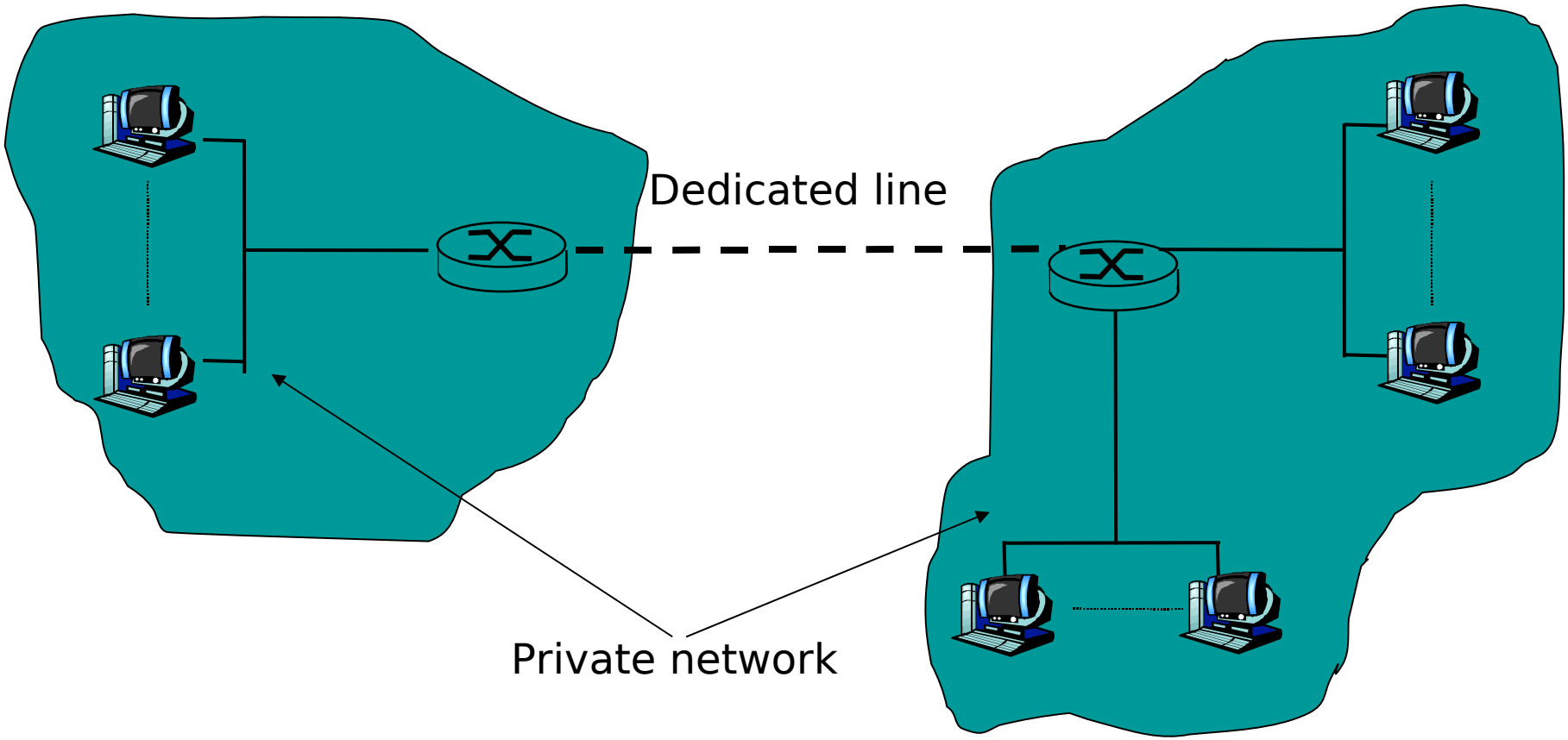
Extensions

- Set of options in first protocol version (RFC 1701)
  - Valid when flag C = 1
- Possible to specify routes (sequence of routers to traverse)
- Encryption
  - Key field
- Sequence numbers
- Checksum

Private networks

- Private network: group of nodes (servers, hosts, routers) not accessible to other networks' traffic
- Typical implementation: rental of lines interconnecting nodes of private network
- Potential advantages
  - Address reuse
  - Privacy
- May be hard/expensive to implement

Completely isolated network

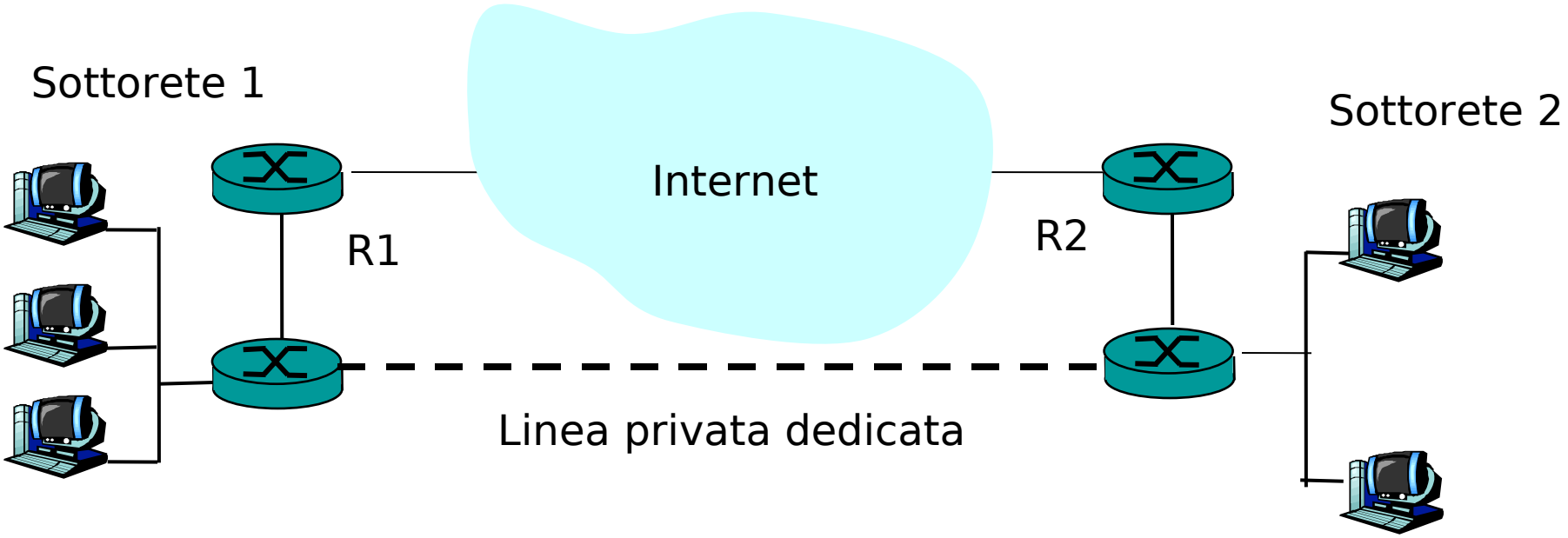


Not possible to access network from outside

Reti ibride

- Linee dedicate connettono i diversi gruppi di nodi che formano la rete privata
- Indirizzi IP pubblici
- Alcuni o tutti i gruppi sono connessi al resto di Internet
- Possibilta' di riservatezza. Es.:
  - Traffico privato su linee dedicate
  - Traffico da/verso il resto di Internet su linee condivise

Reti ibride - esempio



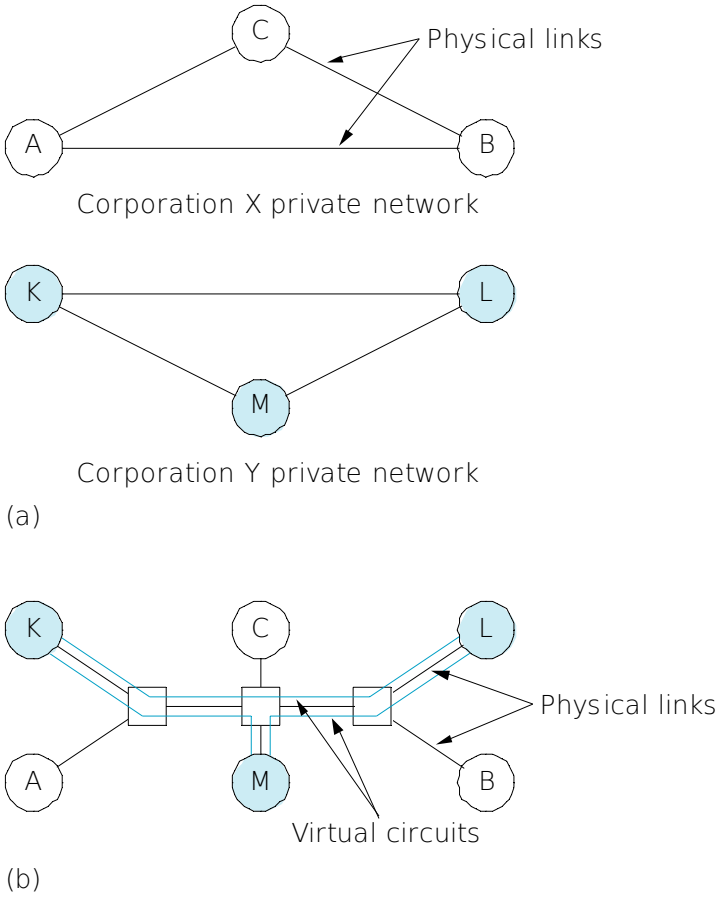
- La Sottorete 1 puo' comunicare con la Sottorete 2 in modo riservato usando la linea dedicata
- L'accesso a Internet avviene attraverso i router R1 e R2
- La rete usa indirizzi IP validi

Virtual Private Networks (VPN)

- Virtual Private Network (VPN)
  - No use of dedicated lines
  - Traffic to/from nodes of private network “virtually” separated from traffic to/from other networks
    - E.g.: by encryption
- Different solutions possible
  - Widespread solution: encryption + tunneling

Esempio - circuiti virtuali

- Nella figura (a) sono mostrate due reti private fisiche
- Nella figura (b) sono mostrate le corrispondenti reti private virtuali
- Si usano circuiti virtuali (es. ATM o Frame-Relay)
- I pacchetti della rete X non possono essere intercettati dalla rete Y se non esiste almeno un VC comune tra esse



Private addressing

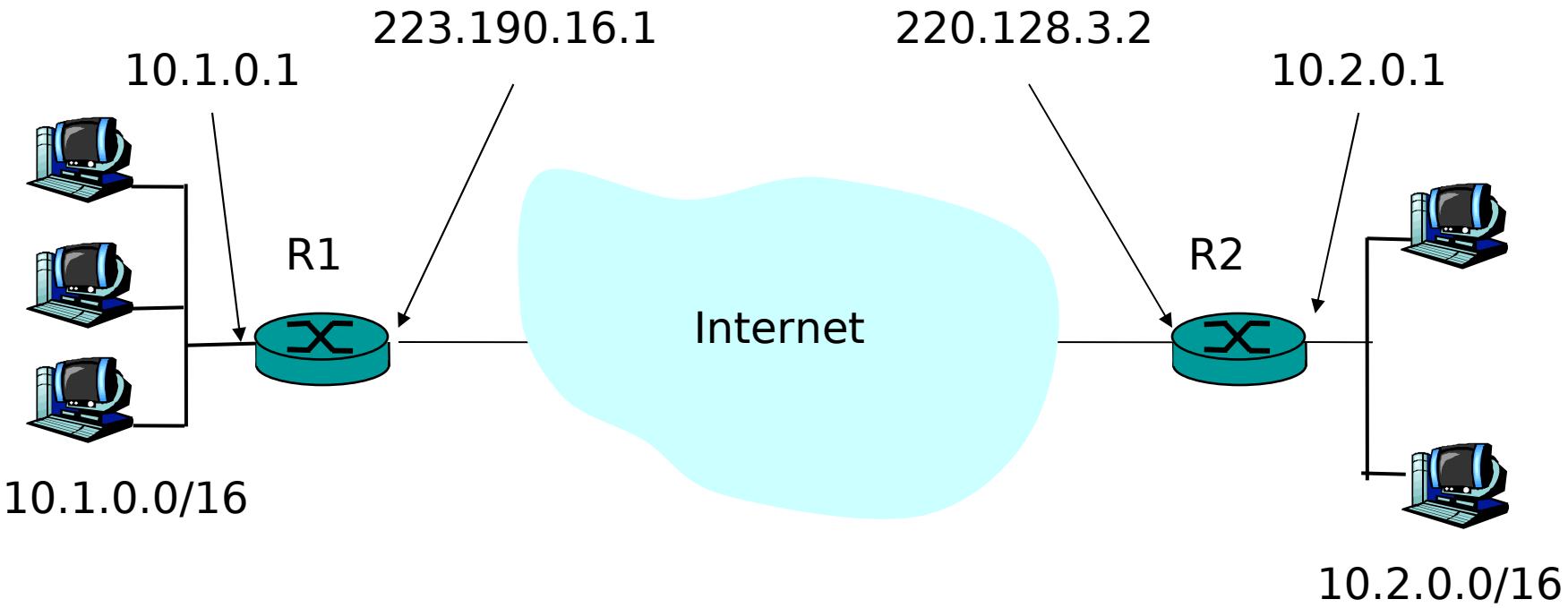
Private addresses

- CIDR specifies a set of address blocks for use in private networks, not directly connected to the Internet
- **Non routable** addresses
  - A datagram with a private destination address is recognised and dropped by an Internet router

Private addresses - cont.

- Private address blocks
  - 10.0.0.0/8
  - 172.16.0.0/12
  - 192.168.0.0/16
  - 169.254/16 (Link-local addresses)
    - Used when no available DHCP server
- More Internet subnets may use these address blocks, as long subnets are not directly accessible from outside
- VPNs often use private addressing for internal nodes

VPN with private addresses



- 2 subnets forming VPN use private address blocks
- R1 and R2 need 2 *global* addresses for tunneling

Pros/cons

- Less global addresses needed
- Problem: hosts of private subnets have no direct Internet access
  - Global IP address needed to this purpose
- Fixes
  - Application gateways
  - NAT: Network Address Translation

Network **A**ddress **T**ranslation

NAT - Network Address Translation

- Association (mapping) between private and public addresses
- NAT allows IP level access to the Internet
- Requirements
  - Single access to the Internet
  - At least one global IP address available
- Application Gateways allow hosts to access specific services (e.g., HTTP, SMTP etc.) but not at IP level

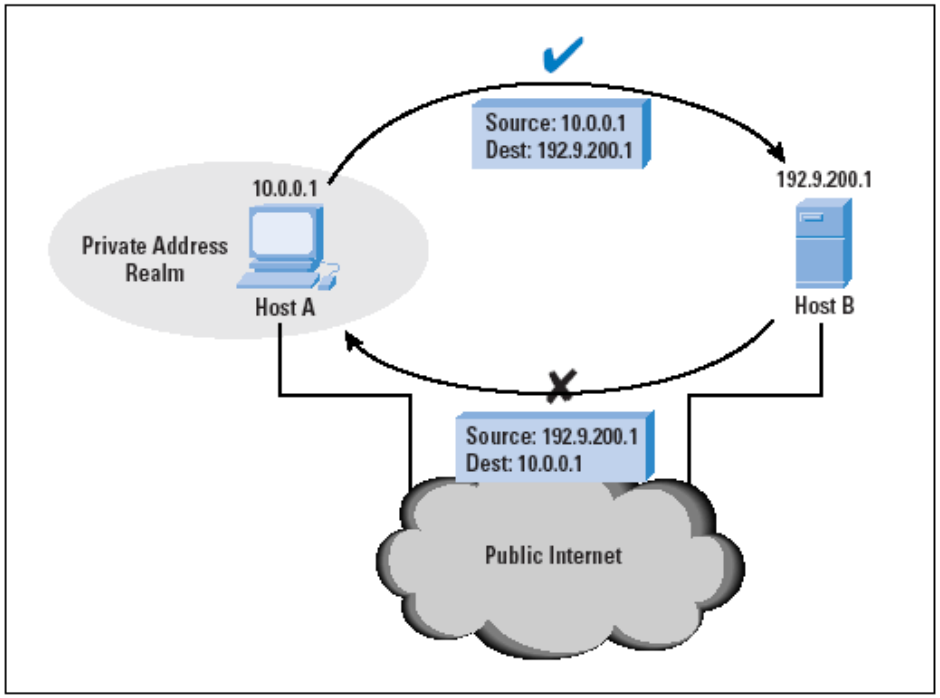
Motivations and history

- First proposed in 1994 to address IP address scarcity
  - Idea: at any given moment, only a subset of internal nodes needs access to the Internet
- Nowdays, NAT used for many purposes
  - Design networks that are not visible outside
  - Security
  - Special cases:
    - Network masquerading
    - Native address translation
    - Port address translation [**PAT**]

NAT

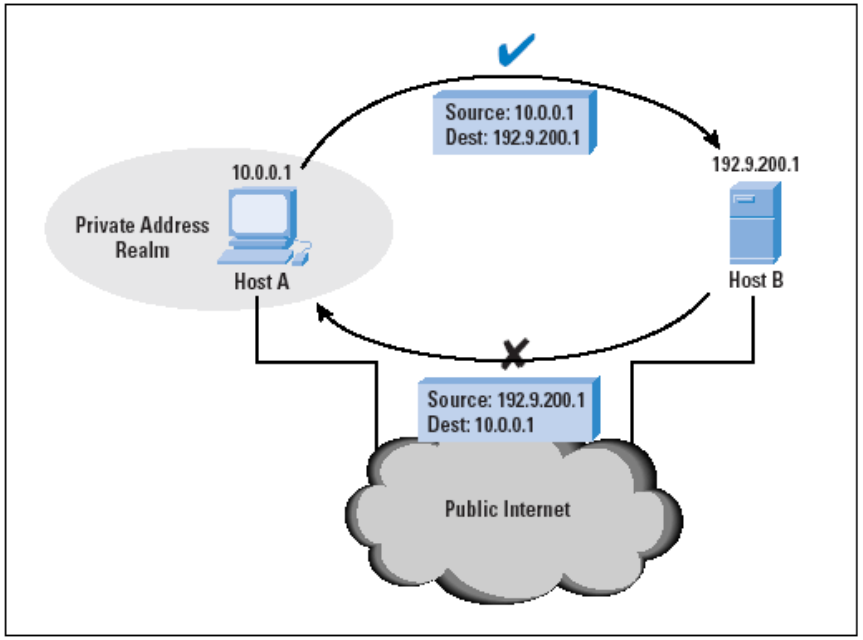
- Leaving private address space is not the problem…
- ….return path is the mess

Figure 2: Public/Private Communication



Basic mechanisms

- Pool of global IP addresses assigned to NAT
- Global addresses mapped to private ones when needed
  - Inbound packets directed to a non-mapped global address ⇒dropped
- Packets sent from source or sent as replies from destination are **suitably manipulated**
  - **Which IP header fields?**
  - **Is payload modified?**

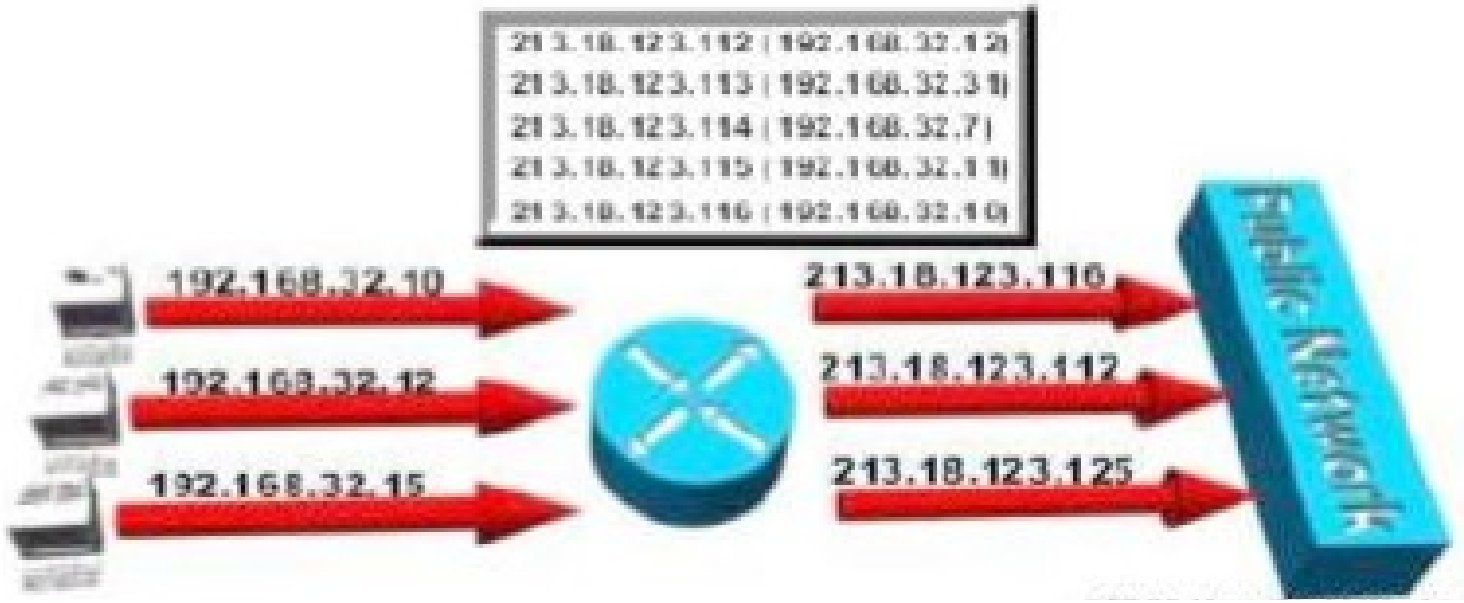


Static NAT

- Static association between global and private address
- Used when a node has to be accessible from outside
- NAT router in this case offers protection

DynamicNAT

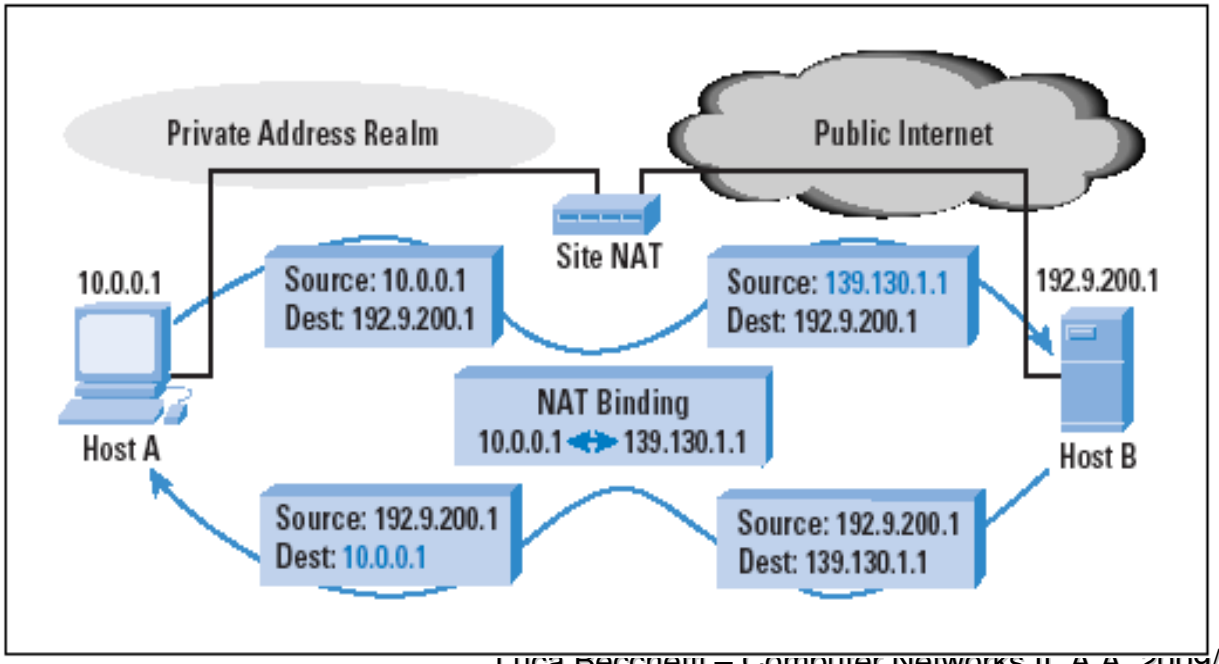
- Dynamic binding between global and local address
- NAT has a pool of global IP addresses
- Same private address can be associated to *different* global addresses over time



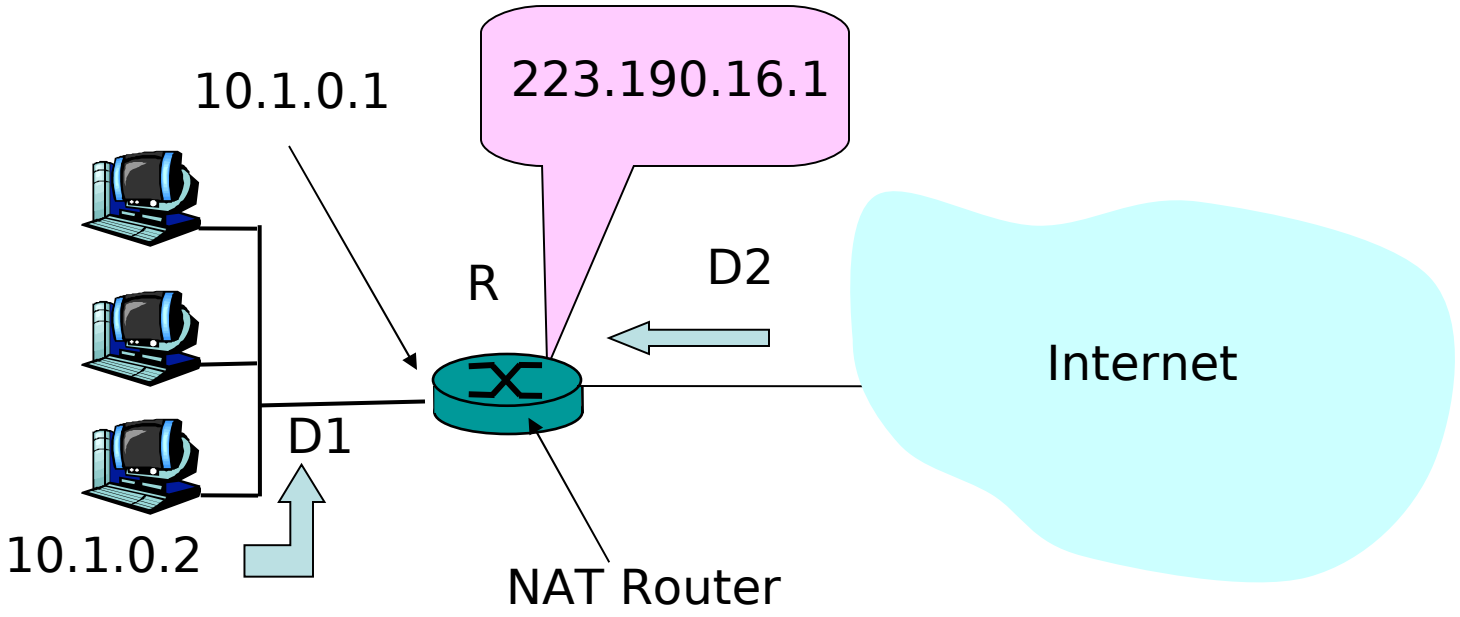
Dynamic NAT

- Dynamic binding
- Binding's expiry: timer-dependent

Figure 3: NAT Traversal



Dynamic NAT: example



"Translation" for datagrams to/from 10.1.0.2

- D1's IP source: 10.1.0.2 ⇒223.190.16.1
- D2's IP destination: 223.190.16.1 ⇒R
- D: how does R understand that D2 is directed to 10.1.0.2?

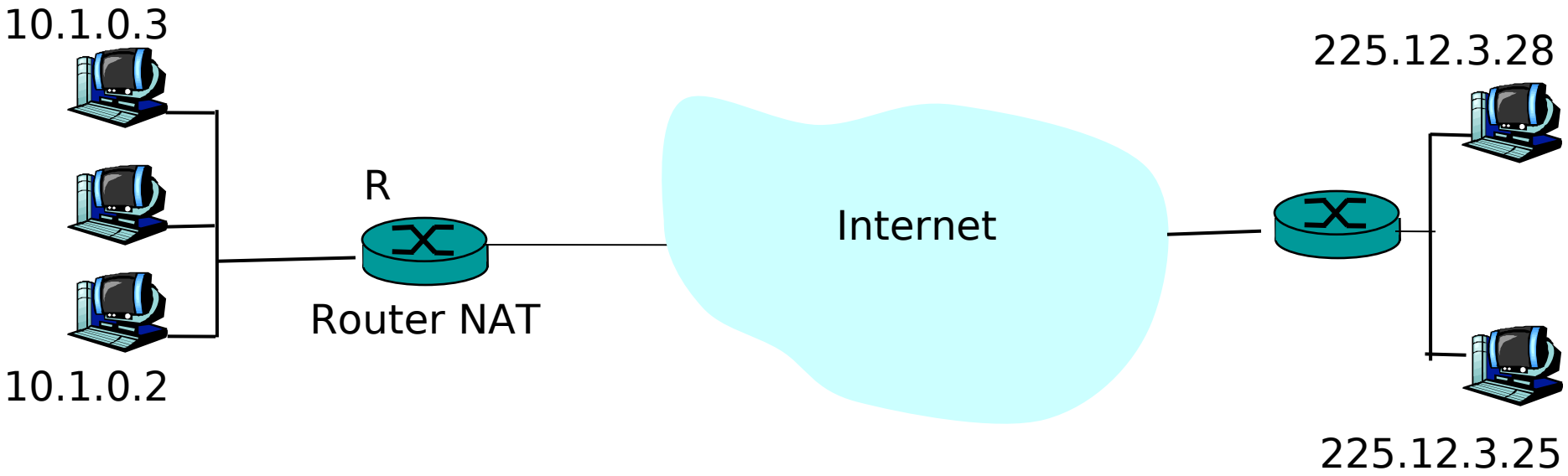
Implementation issues

- All Internet packets reaching R have same destination address (in the example: 223.190.16.1)
- How to map global destination address to corresponding private one within local network?
- E.g.: D2 is reply to D1, so internal destination for D2 is 10.1.0.2
- **Solutions**
  - **Tables**
  - **Multi-address NAT**
  - **NAT with port mapping**

NAT - translation tables

- Needed to route incoming datagram to correct private address
  - All incoming datagrams have *same* destination address
- Based on the following assumption:
  - If (internal) host H sends datagram to (Internet) host A then datagrams coming from H are destined to H
- Works if at most 1 internal host at a time can exchange datagrams with the same global host
  - Remark: no distinction based on transport protocol

NAT tables – example 1/2



Source	Dest. host
225.12.3.28	10.1.0.2
225.12.3.25	10.1.0.3

R's translation table

- 10.1.0.2 communicates with 225.12.3.28
- 10.1.0.3 communicates with 225.12.3.25

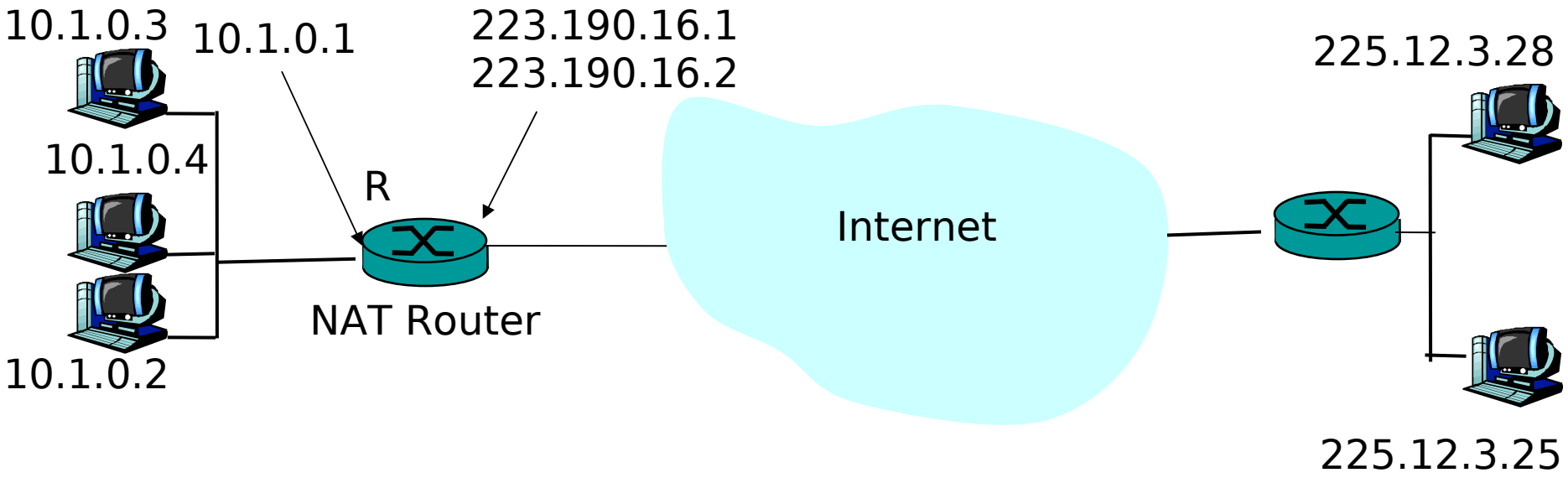
NAT tables – example 2/2

- NAT tables - initialization
  - Manual
  - Outbound datagram: first time that R receives datagram from 10.1.0.2 to 225.12.3.28 it creates entry (225.12.3.28, 10.1.0.2) in its NAT table - most common solution
  - Domain name based initialization
- Main drawback: at most 1 local host can access to the same Internet machine

NAT with multi-addressing

- Idea: use NAT tables with a pool of global IP addresses (instead of one)
  - NAT router's external interface has more than 1 associated global address
  - Global addresses assigned round-robin to outbound datagrams
  - K global addresses allow *at most* K internal hosts to exchange info with the same Internet host

E.g.: K=2



Source	Global dest. IP.	Dest. host
225.12.3.28	223.190.16.1	10.1.0.2
225.12.3.28	223.190.16.2	10.1.0.4
225.12.3.25	223.190.16.1	10.1.0.3

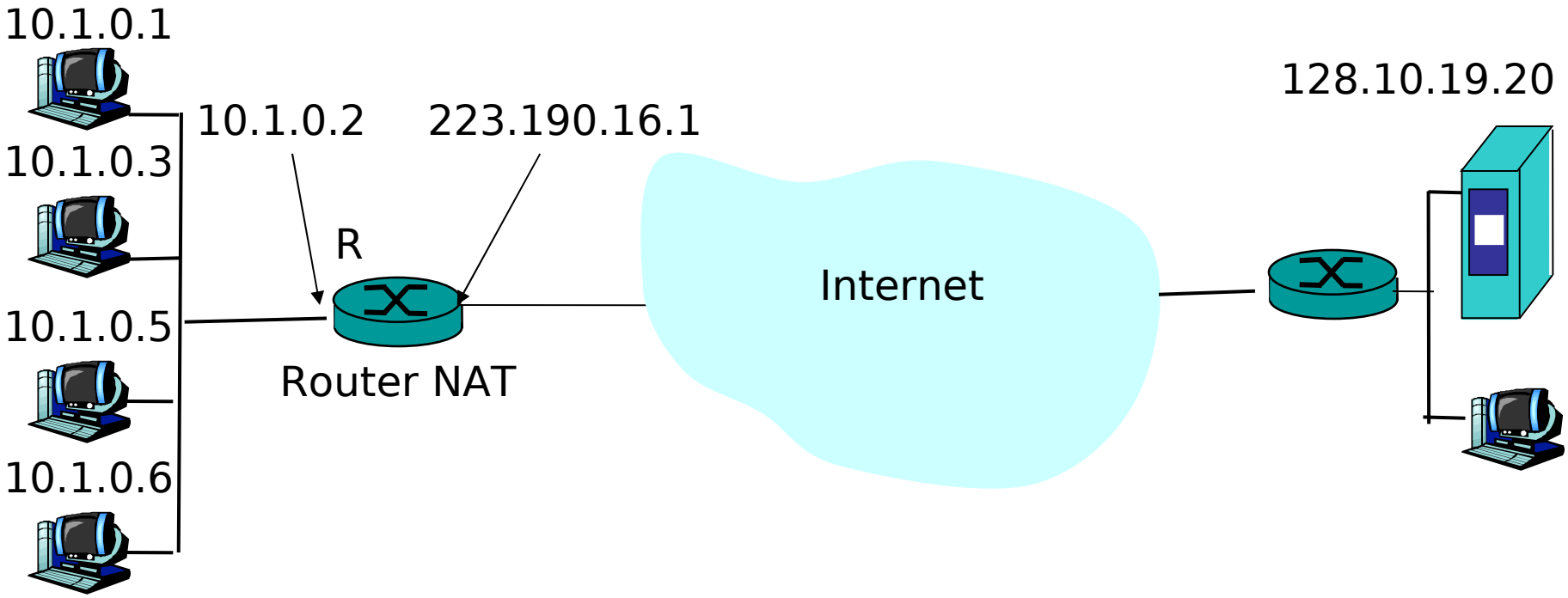
R's translation table

- 10.1.0.2 and 10.0.0.4 communicate with 225.12.3.28
- 10.1.0.3 communicates with 225.12.3.25

NAT with Port mapping

- NAT router distinguishes (internal) destinations of incoming datagrams also on the basis of transport protocol information
- Used when datagrams carry TCP or UDP segments
- It also uses port numbers
- Necessary to extend entry fields of NAT tables
- More processing complexity at NAT routers

Port mapping example 1/3



- 10.1.0.1 e 10.1.0.5 both have ongoing HTTP connection with Web server 128.10.19.20 on port 80
  - 2 TCP connections to 128.10.19.20 on port 80
- Other hosts have ongoing TCP connections to other machines not shown in the picture

Port mapping example 2/3

Local side		Protocol	Remote side		NAT info	
Address	Port		Address	Port	Port	Protocol
10.1.0.5	21023	TCP	128.10.19.20	80	14003	TCP
10.1.0.1	2386	TCP	128.10.19.20	80	14010	TCP
10.1.0.6	26600	UDP	207.200.75.12	21	14012	UDP
10.1.0.3	1274	UDP	128.10.19.20	80	14003	UDP

- NAT table
  - First 2 lines correspond to previous picture
  - Notice *explicit* management of transport protocol

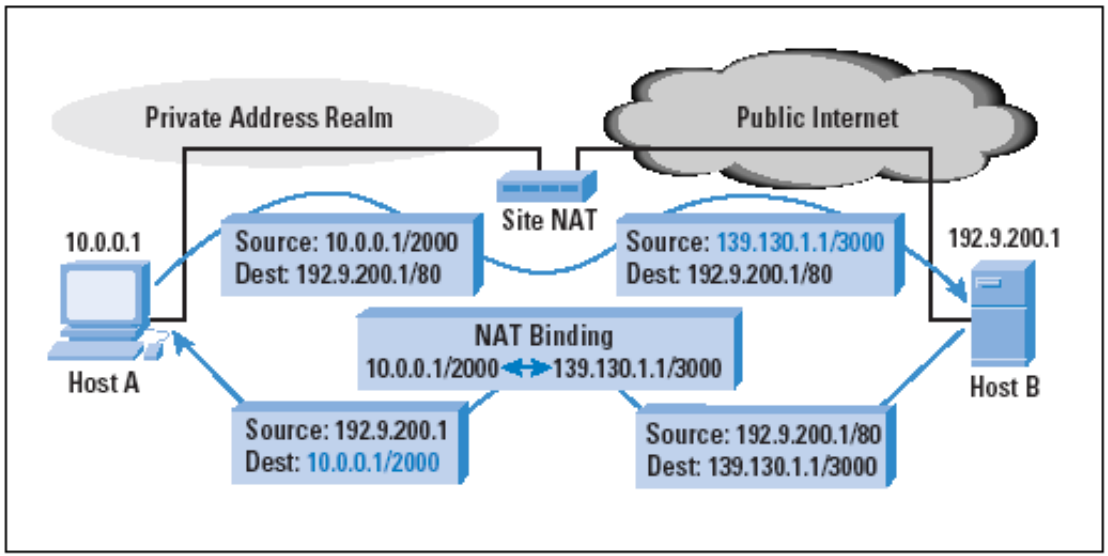
Port mapping example 3/3

- NAT maintains *distinct* NAT port for every information exchange [**binding**]
  - Two local hosts might choose same private port number
  - Host 128.10.19.20 can distinguish the two TCP connections corresponding to the first two lines of the previous table:
    - (**223.190.16.1, 14003**, 128.10.19.20, 80)
    - (**223.190.16.1, 14010**, 128.10.19.20, 80)

Port mapping

- Binding expires according to associated timer
- Up to 65536 sessions for every <external address,protocol> pair

Figure 4: NAPT Traversal



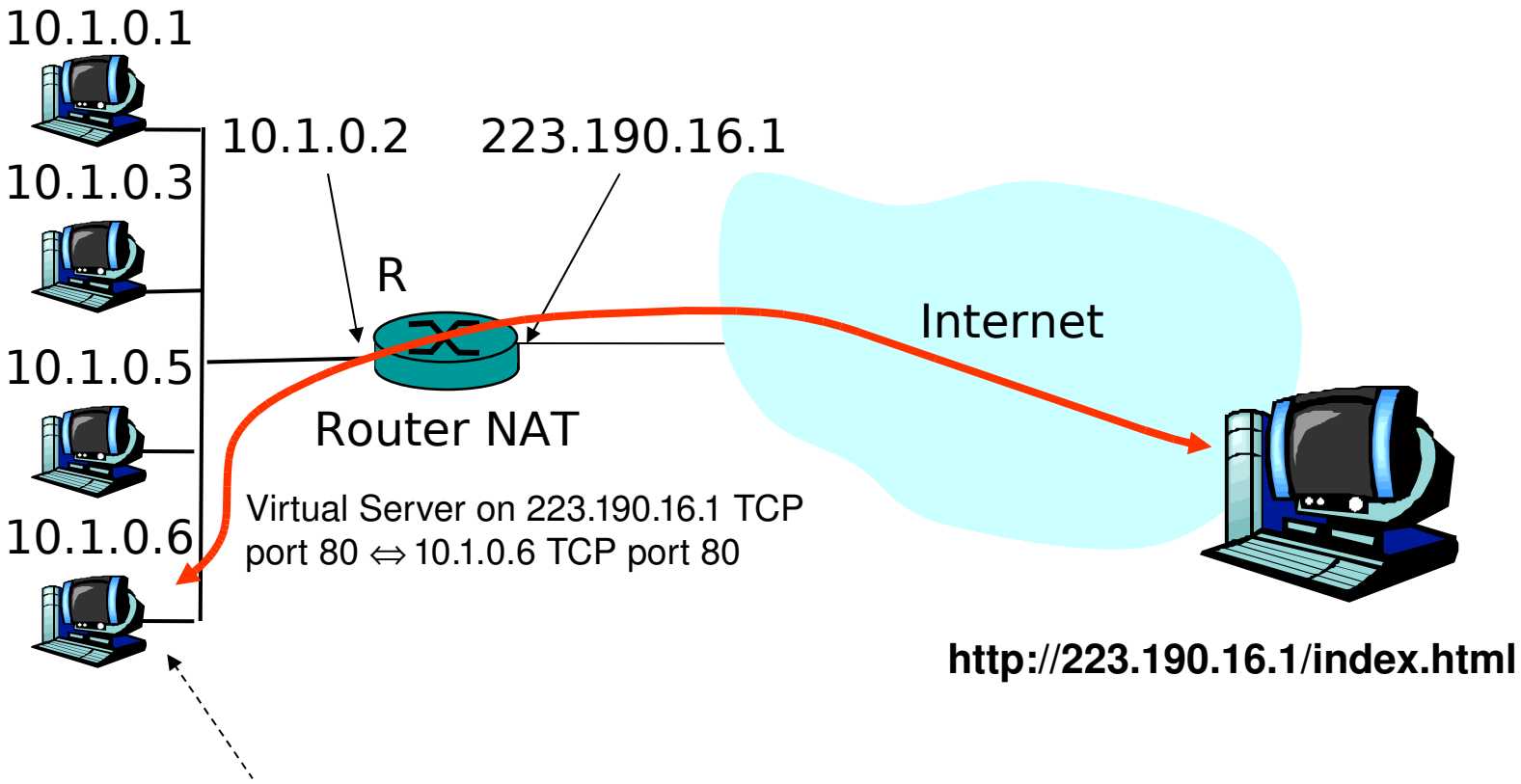
Virtual Server

- For a given transport protocol [TCP and/or UDP]: static binding between <global_address,global_port> and <local_address, local_port>

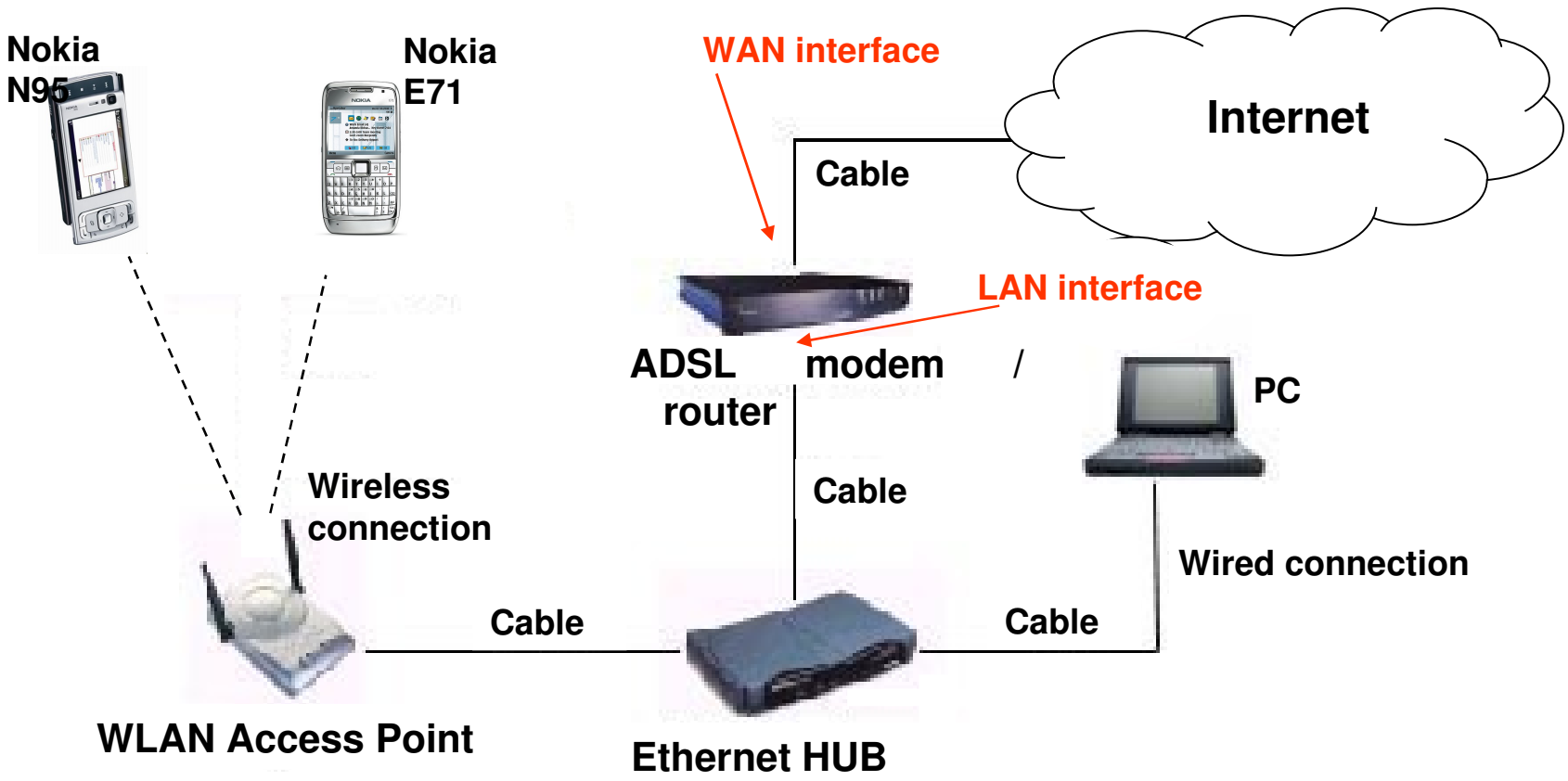
- Allows active services on a host with private address to be reachable over the Internet
  - Examples: Web Server, UDP server,…

- Virtual Server vs dynamic Port mapping
  - Dynamic Port mapping activates binding only AFTER private host initiates transmission and binding is removed when associated timer expires (after binding not used for a time longer than timer)

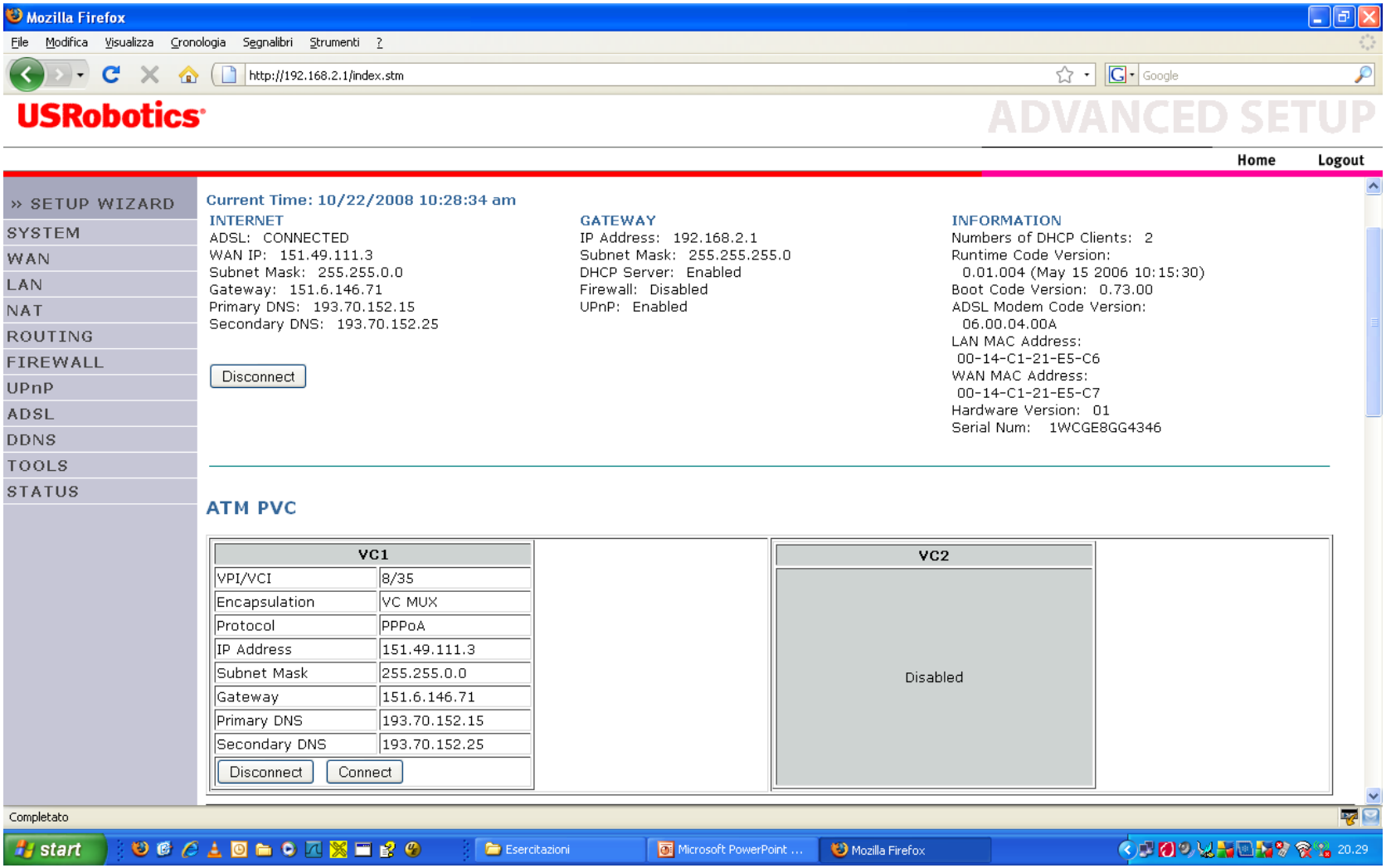
Virtual Server: example



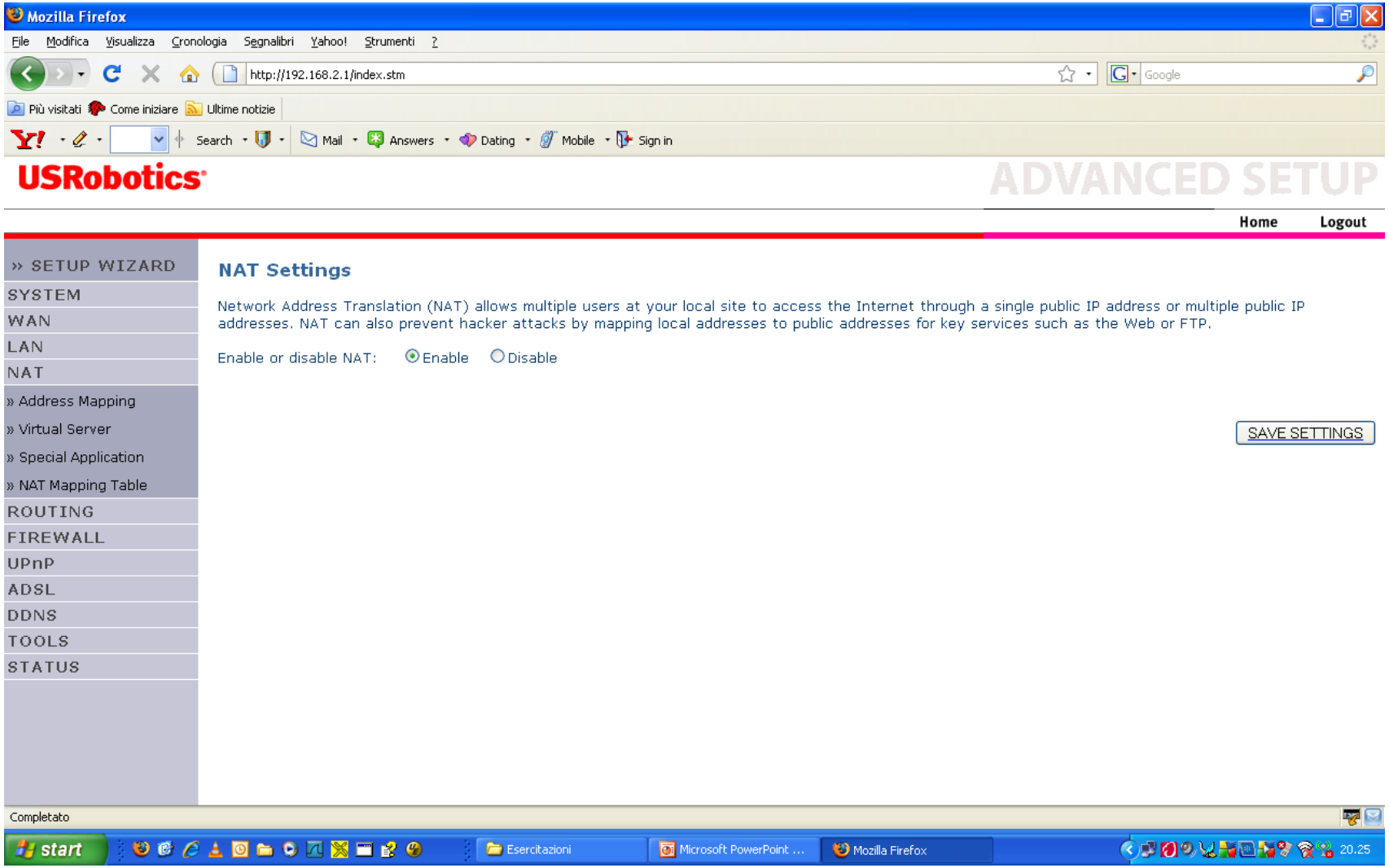
NAT example 1/5: network scenario



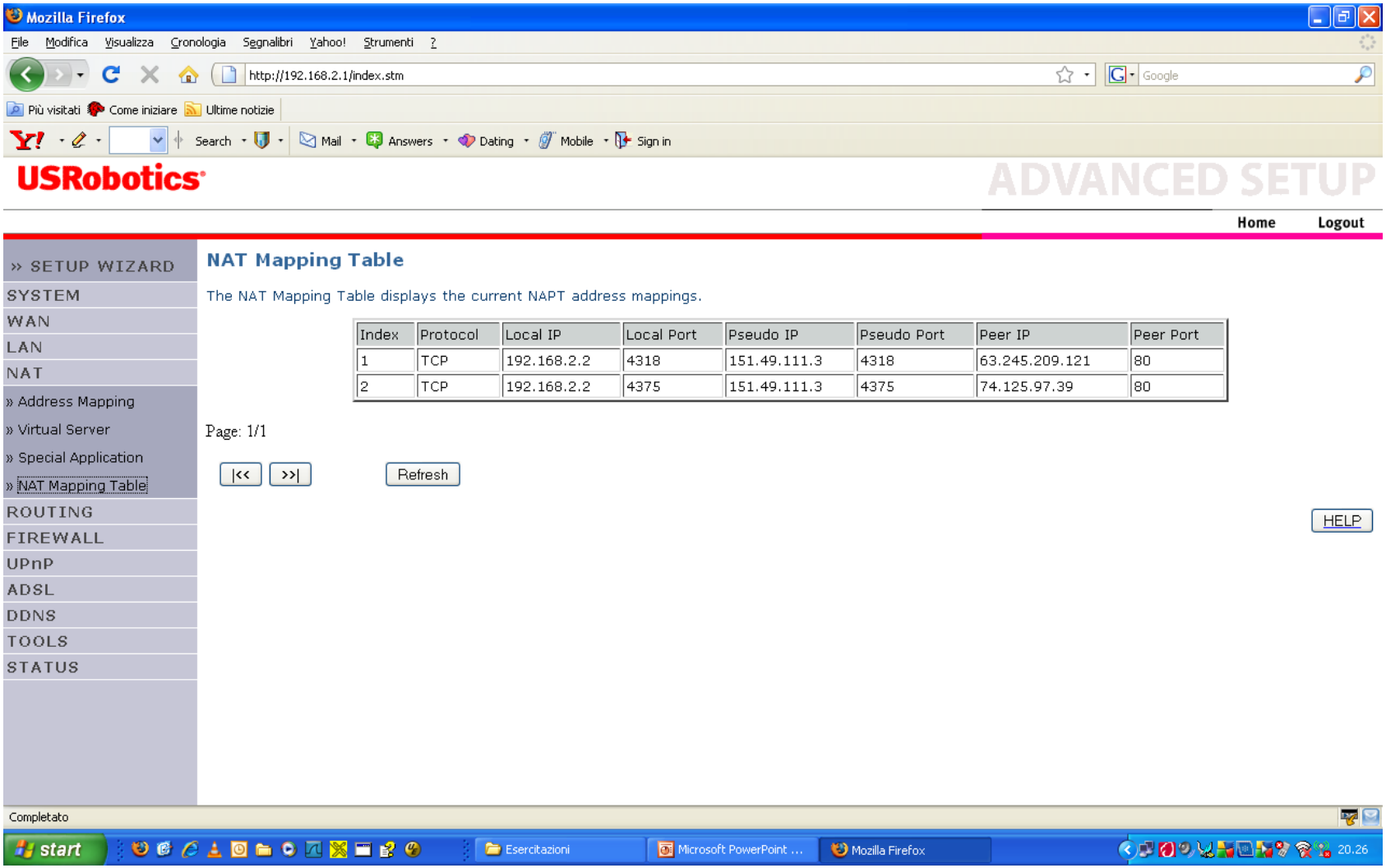
NAT example 2/5: router status



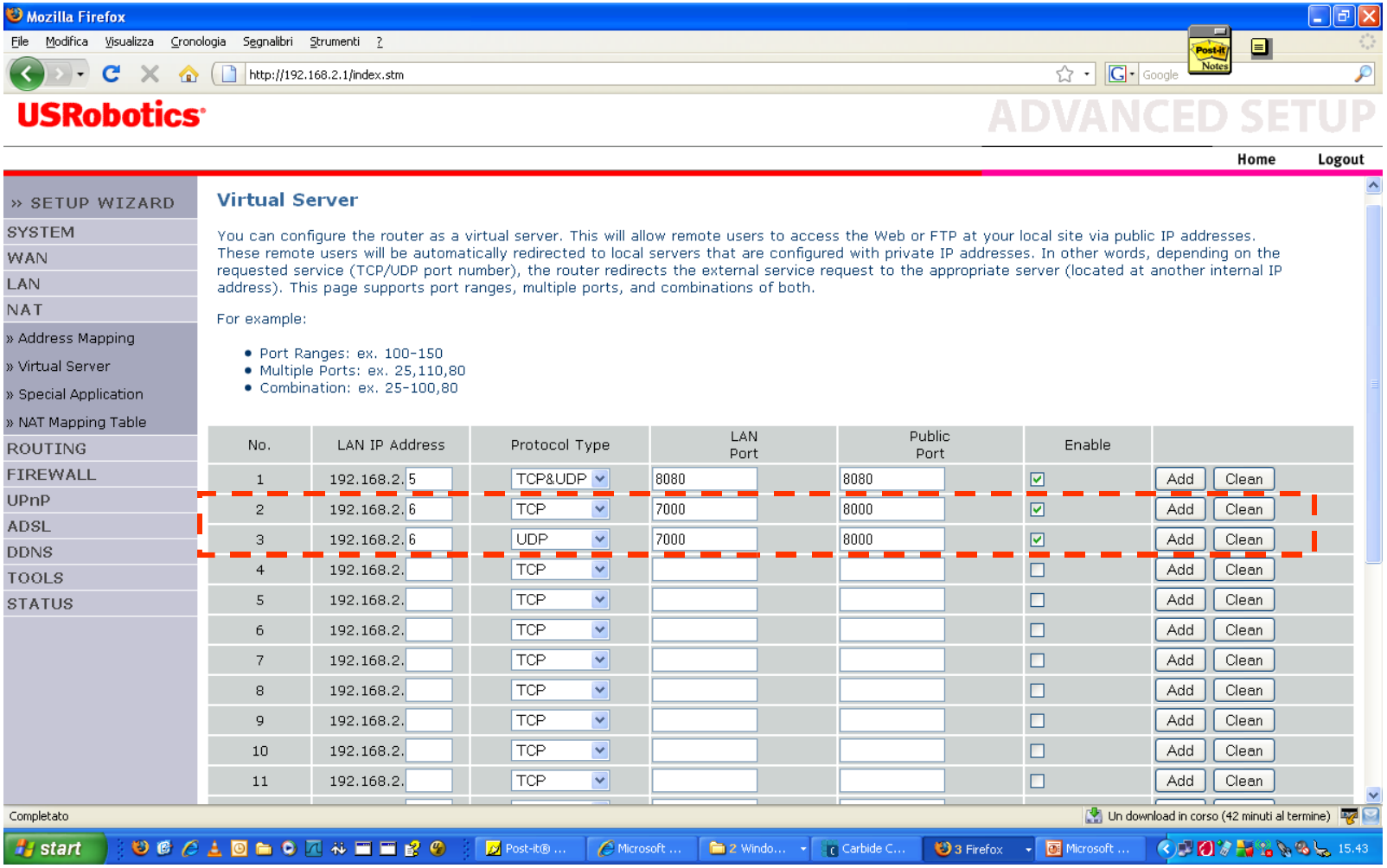
NAT example 3/5: router NAT service



NAT example 4/5: router NAT mapping table



NAT example 5/5: router virtual servers



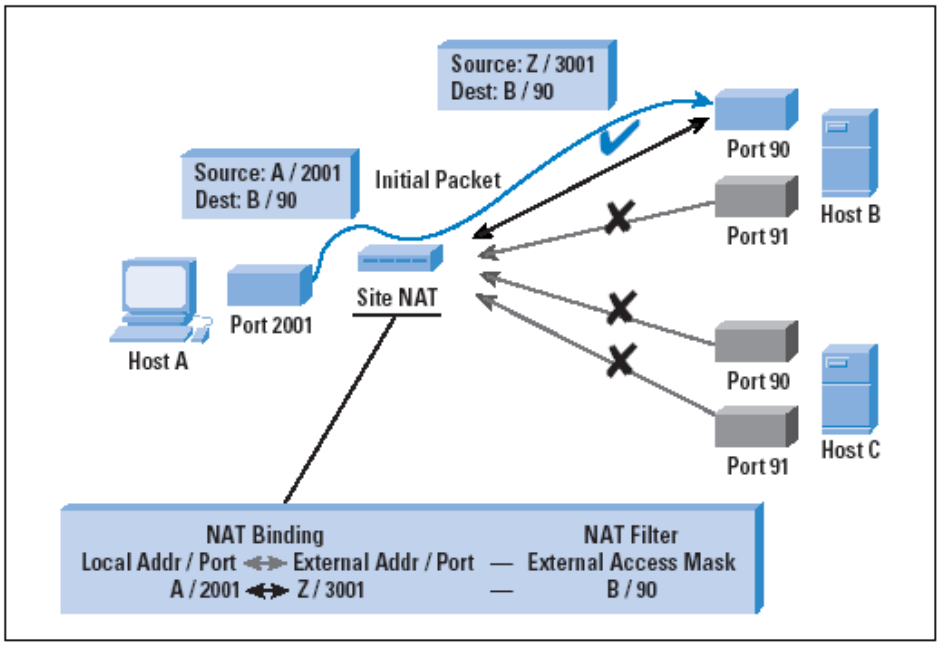
NAT behaviour

- Many differenceses among proposed solutions
  - Even between different solutions by *same* vendor
- Some applications require info about “client IP address” [see example further]
  - What happens with NAT? Often it is necessary to understand which binding scheme is used
  - Specific protocol to this purpose: STUN [Simple traversal of UDP over NATs];
  - STUN classifies binding strategies as follows:
    - Simmetric
    - Full-cone
    - Address restricted-cone
    - Port-restricted-cone
  - In practice: NAT devices can actually combine different strategies :-)

Symmetric NAT

- Mapping determined by destination address/port
- Incoming (external) packets filtered according to source address/port
- Only filtered packets are allowed

Figure 5: Symmetric NAT



Each request from the same internal IP address and port to a specific destination IP address and port is mapped to a unique external source IP address and port.

If the same internal host sends a packet even with the same source address and port but to a different destination, a different mapping is used.

Only an external host that receives a packet from an internal host can send a packet back

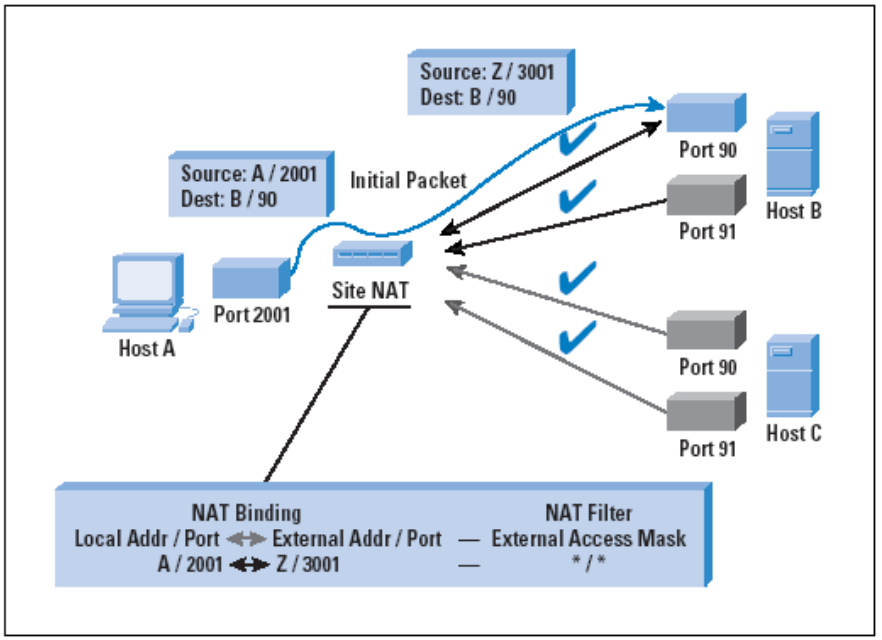
Full-cone NAT

- Also known as NAT 1:1
- Incoming (external) packets are allowed whenever destination address/port pair is currently used

Figure 6: Full Cone NAT

Once an internal address (iAddr:port1) is mapped to an external address (eAddr:port2), any packets from iAddr:port1 will be sent through eAddr:port2.

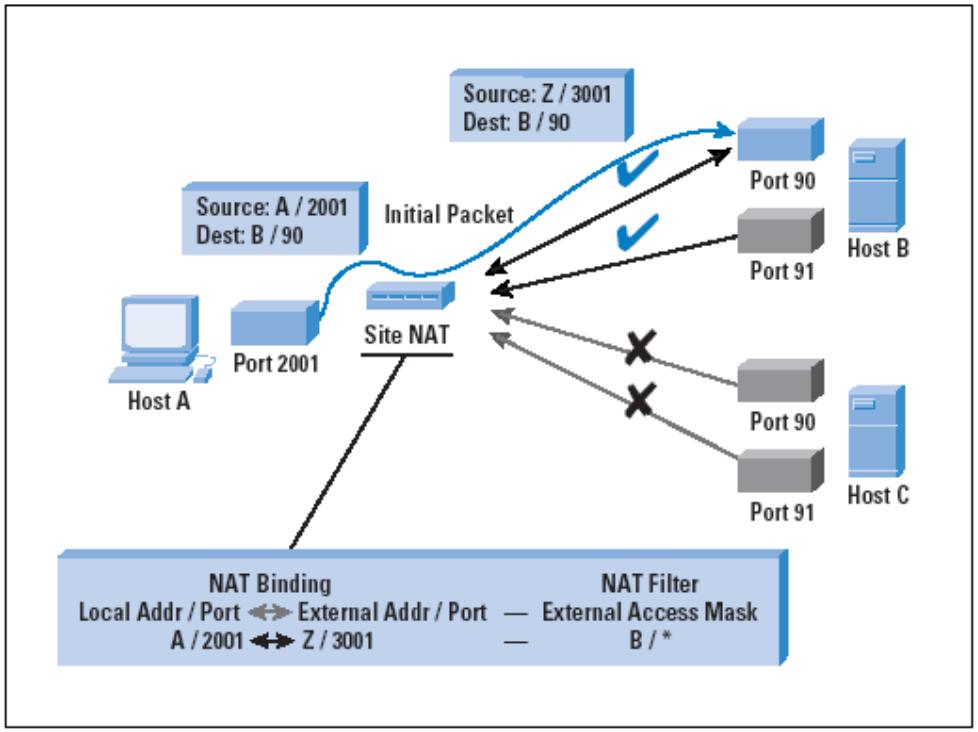
Any external host can send packets to iAddr:port1 by sending packets to eAddr:port2.



Address-restricted-cone or Restricted-cone

- Incoming (external) packets allowed whenever destination address/port is currently used and source address is associated to the pair in NAT binding

Figure 7: Restricted-Cone NAT



Once an internal address (iAddr:port1) is mapped to an external address (eAddr:port2), any packets from iAddr:port1 will be sent through eAddr:port2.

An external host (hostAddr:any) can send packets to iAddr:port1 by sending packets to eAddr:port2 only if iAddr:port1 had previously sent a packet to hostAddr:any; "any" means that port number doesn't matter

Port-restricted-cone

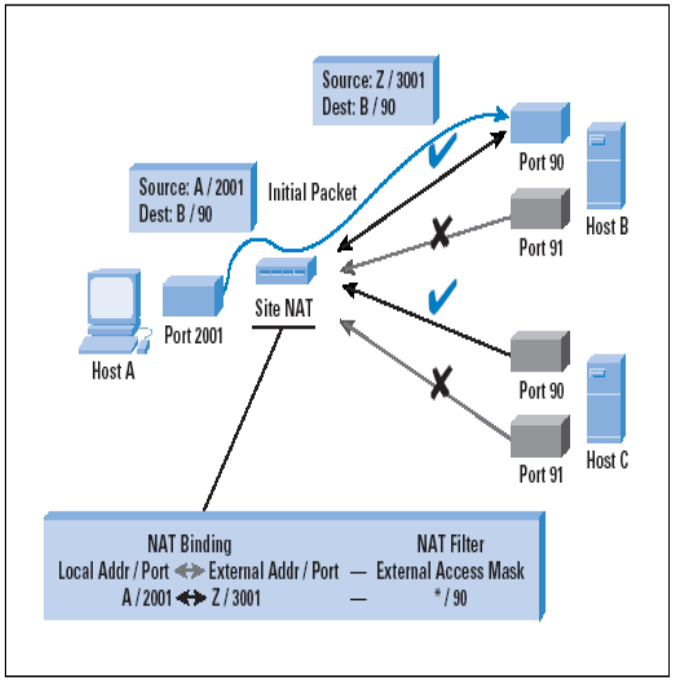
- Incoming (external) packets allowed whenever destination address/port is currently in use and source address is the one associated to the pair under consideration in NAT binding

Like a restricted cone NAT, but the restriction includes port numbers.

Once an internal address (iAddr:port1) is mapped to an external address (eAddr:port2), any packets from iAddr:port1 will be sent through eAddr:port2.

An external host (hostAddr:port3) can send packets to iAddr:port1 by sending packets to eAddr:port2 only if iAddr:port1 had previously sent a packet to hostAddr:port3.

Figure 8: Port-Restricted-Cone NAT



Any external host can send packets to internal_address / internal_port by sending packets to external_addres / external_port

Interaction with other protocols

- NAT interacts with upper (layer) protocols and ICMP
- NAT has to:
  - Modify IP headers
  - Modify TCP and UDP port numbers
  - Recompute checksums (because headers change)
  - Explicitely manage ICMP messages (e.g., ping)
- Which impact at the application level?
  - E.g., FTP, SIP,…

Interactions with other protocols/cont.

- Some applications need “client IP address” info [see example further]
  - What happens with NAT? which <IP,port> info does client communicate to application’s server side?
    - If private info communicated --> client could be unreachable
    - If global info communicated [NAT] it needs to know it

Interaction with other protocols
VoIP / SIP Example 1/13

- scenario:
  - VoIP / SIP softphone [⇔VoIP client] registers for the services of a SIP *Registrar* server
  - SIP / UDP / IP
  - Softphone uses UDP port 25992 for SIP signalling
  - Registrar SIP uses UDP port 5061 for SIP signalling
  - SIP registration message [REGISTER] contains header *Contact: ……* which provides to the *registrar* client's reachability info in the case of incoming calls: < client_ip_address, client_UDP_port>
  - Upon a phone call, client must communicate to SIP Proxy which <ip_address, UDP_port> it is going to use for RTP audio session

Interaction with other protocols

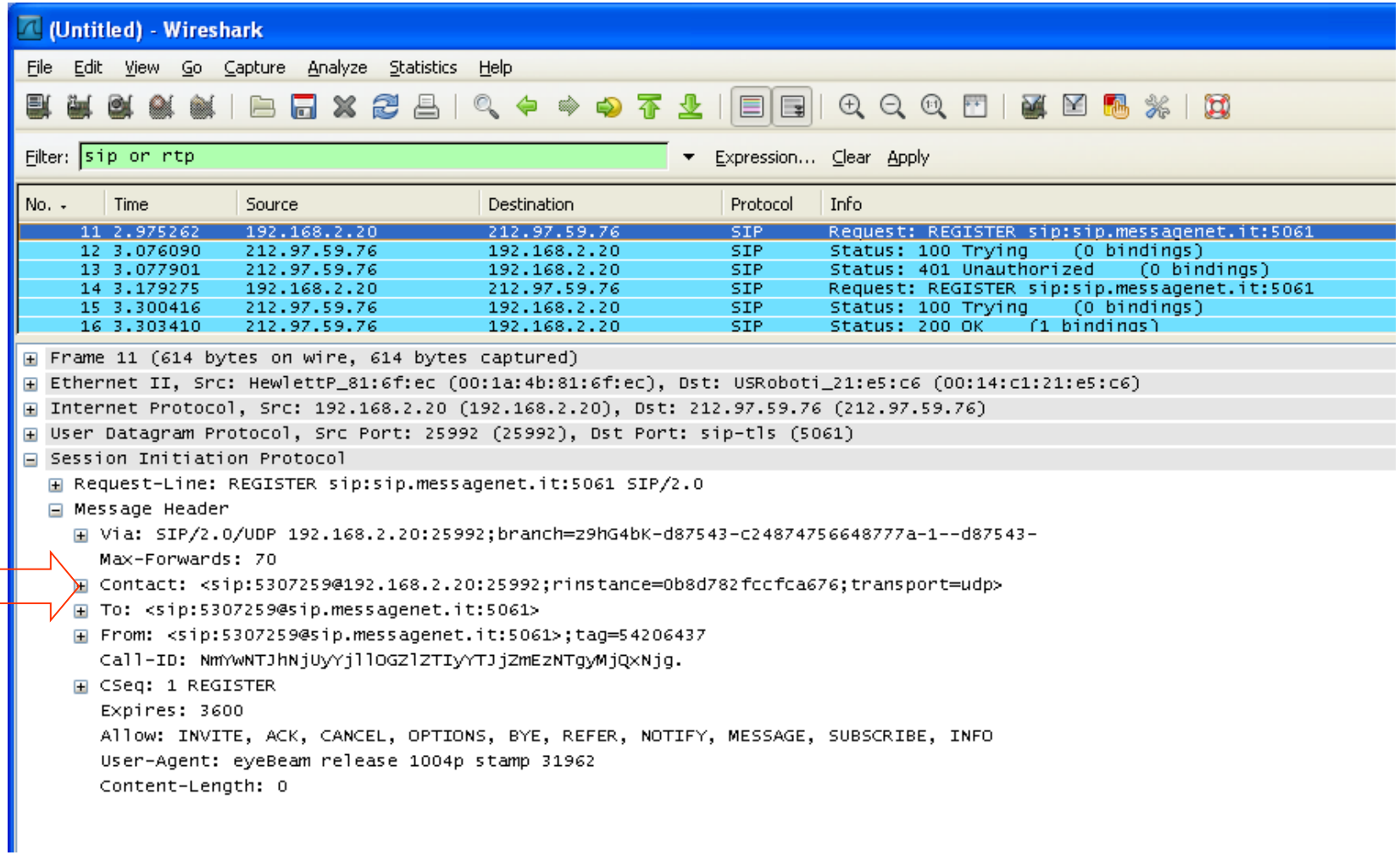
VoIP / SIP Example 2/13

- I 2 aspetti evidenziati rientrano nelle problematiche di gestione del **NAT traversal**:
- 2 Alternative:
  - Gestione lato server:
    - Il client comunica al registrar ed al proxy il proprio indirizzo privato ed i numeri di porta UDP locali
    - il registrar ignora le info di raggiungibilità comunicate dal client e le recupera invece dal pacchetto UDP ricevuto [external_address, external_port]
    - Il proxy deve aspettare il primo pacchetto RTP inviato dal client per conoscere su quale coppia <IP,UDP_port> il NAT ha mappato l'indirizzo IP e la porta UDP per il traffico RTP del client
  - Gestione lato client:
    - Il client utilizza il protocollo STUN [o equivalente] per conoscere su quali coppie <IP,UDP_port> sono mappate:
      - la sua porta SIP locale; questo mapping deve restate invariato nel tempo $\Rightarrow$**necessario refresh del binding in assenza di traffico**
      - la porta UDP per la sessione RTP; questo mapping è in generale diverso nel tempo

Interaction with other protocols

VoIP / SIP Example 3/13

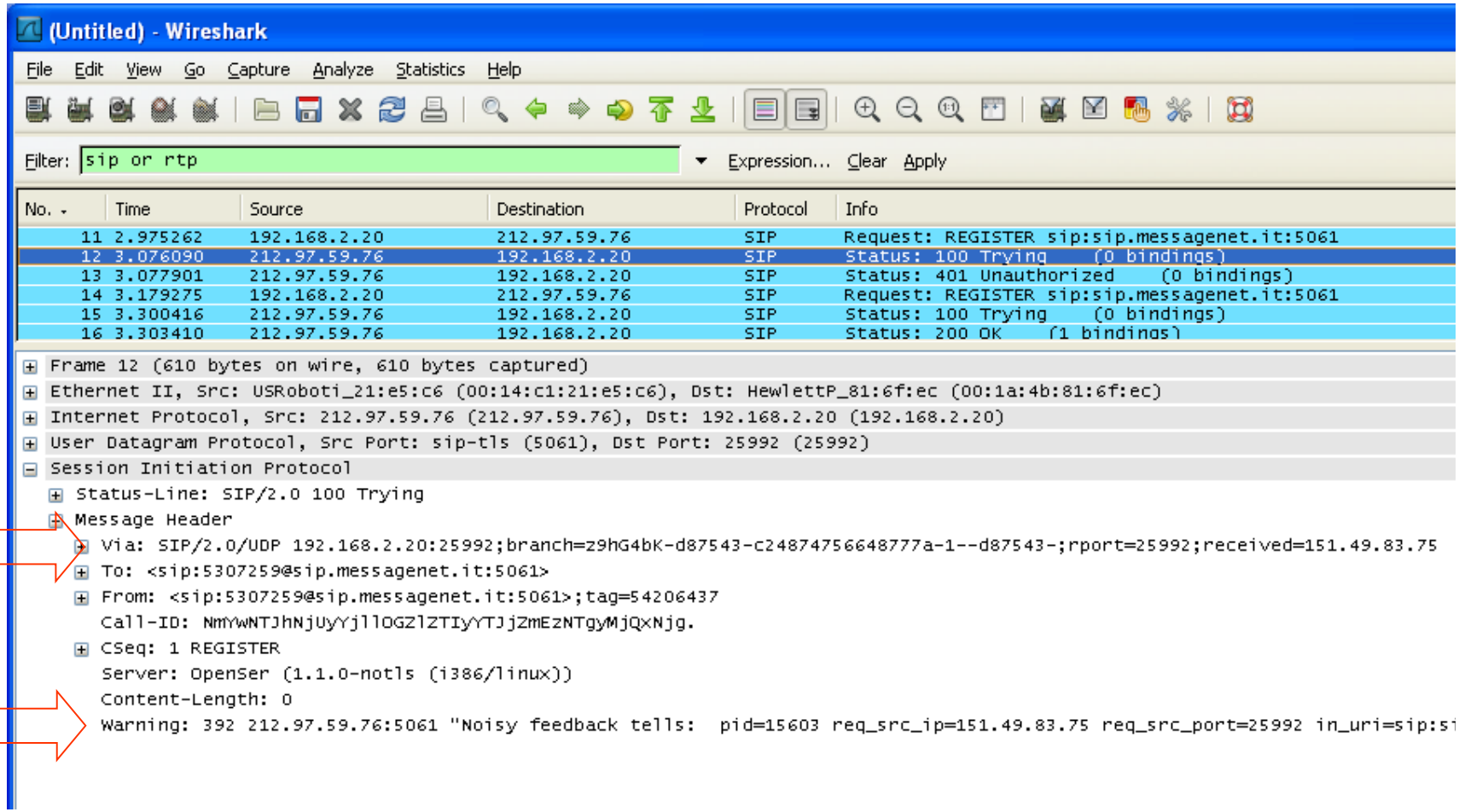
Server side management:



Interaction with other protocols

VoIP / SIP Example 4/13

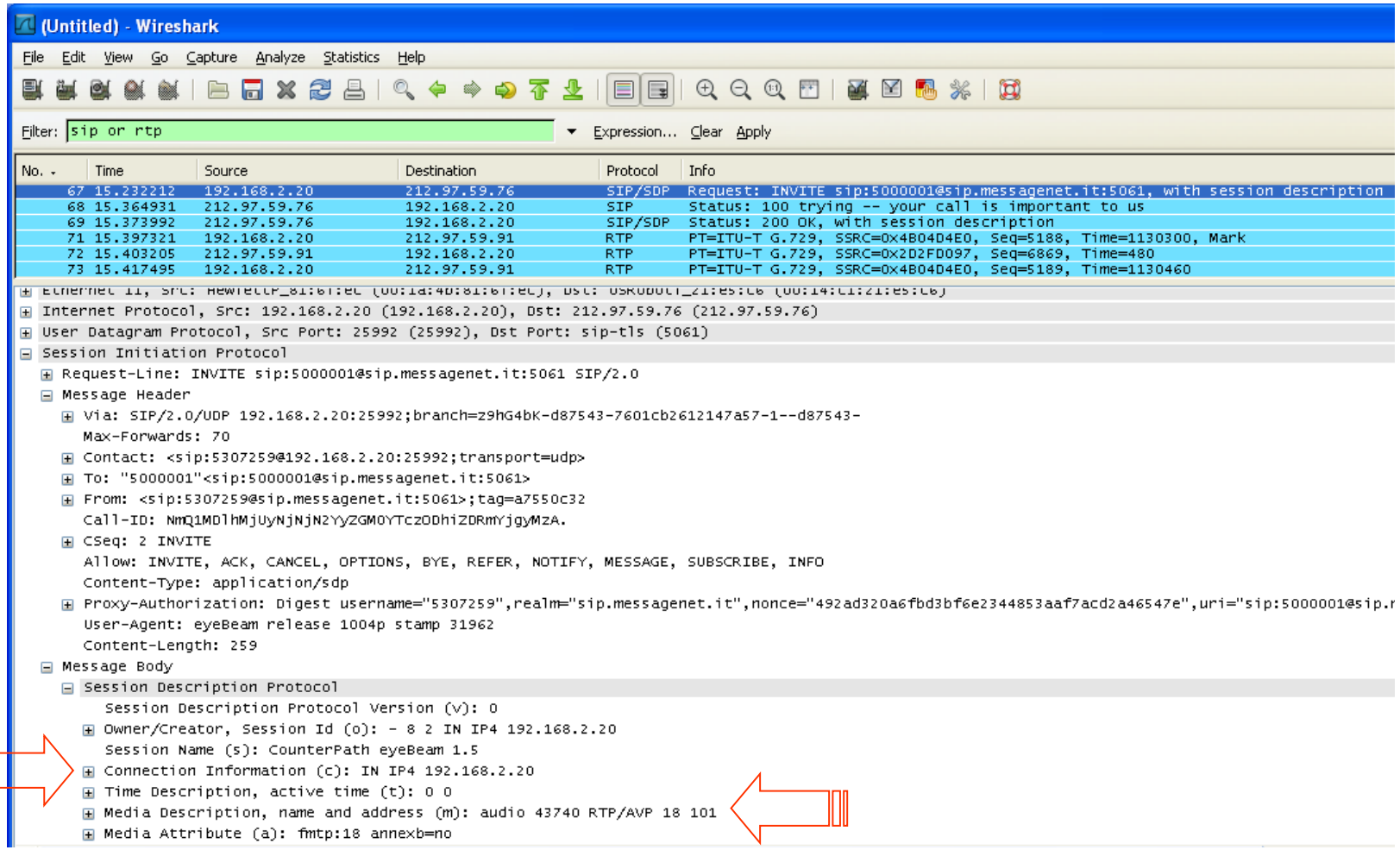
Server side management:



Interaction with other protocols

VoIP / SIP Example 5/13

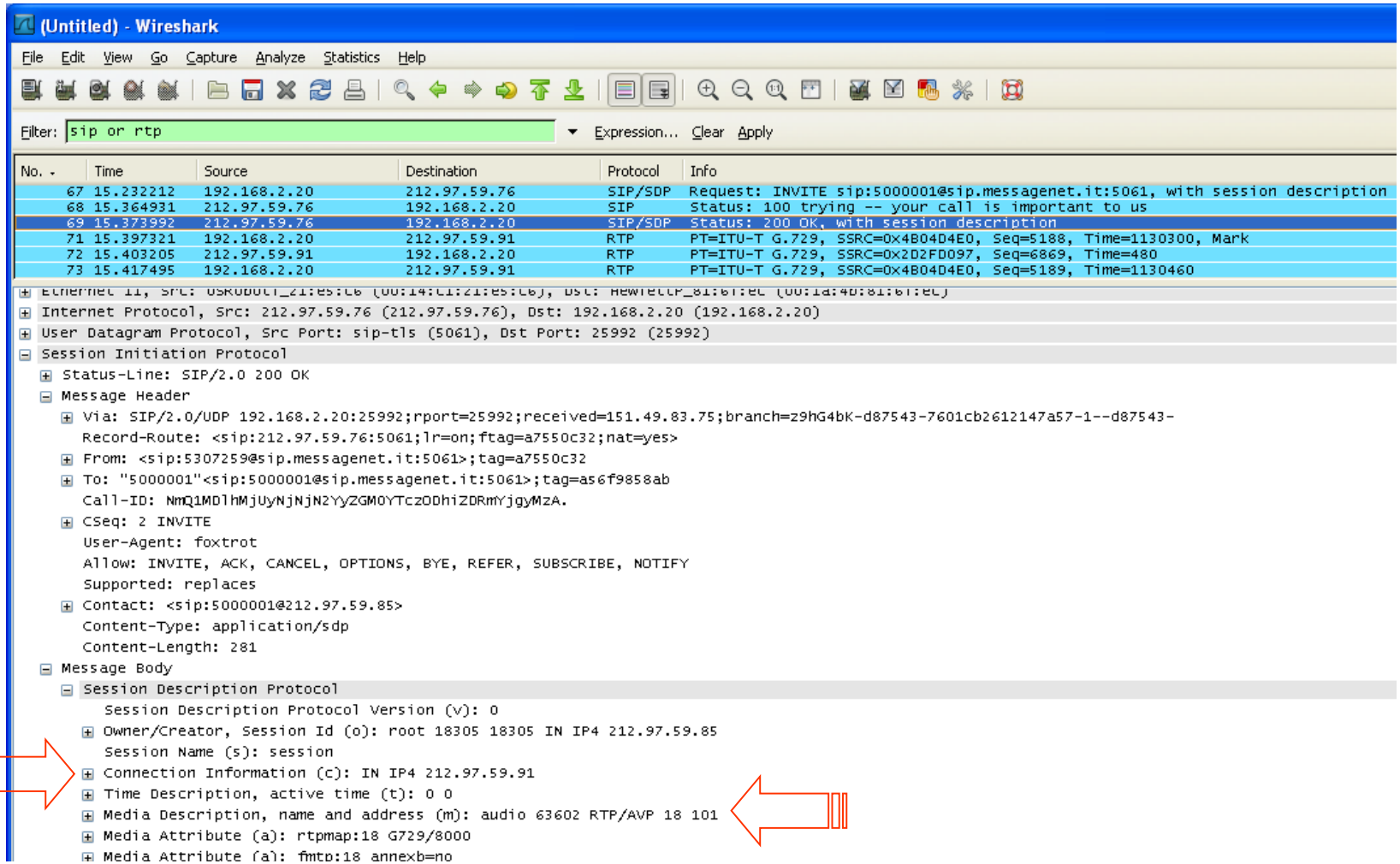
Server side management:



Interaction with other protocols

VoIP / SIP Example 6/13

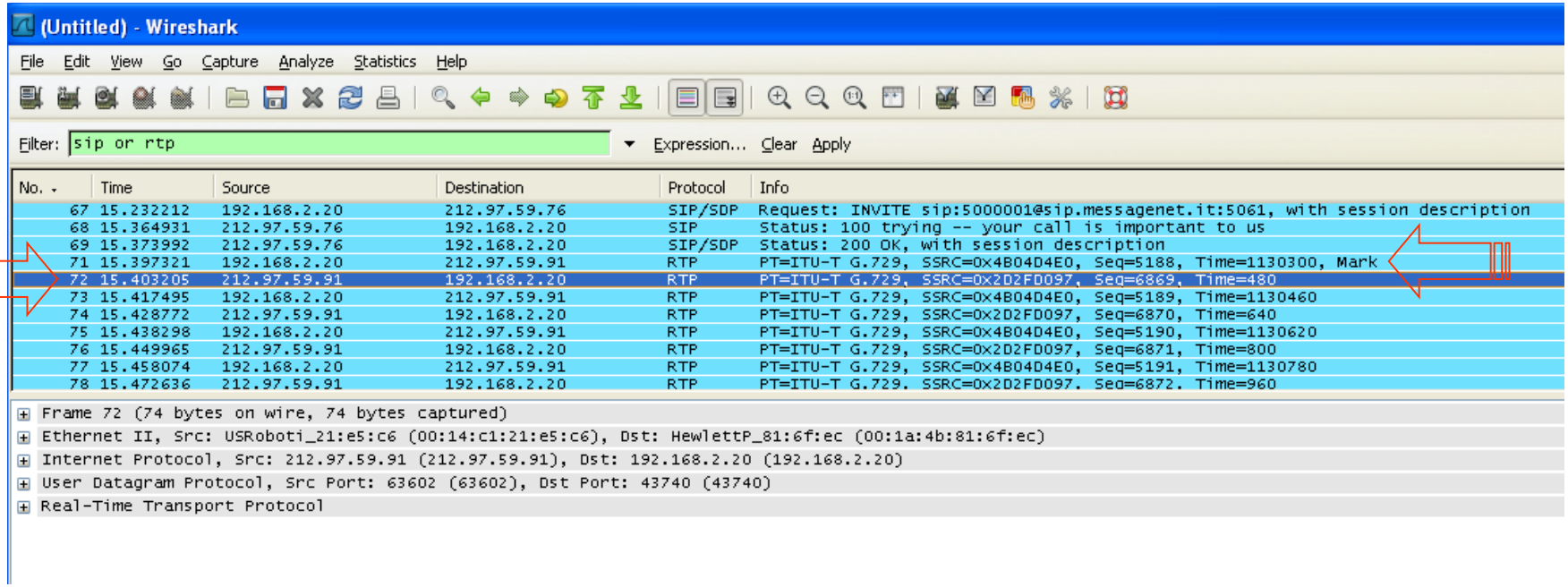
Server side management:



Interaction with other protocols

VoIP / SIP Example 7/13

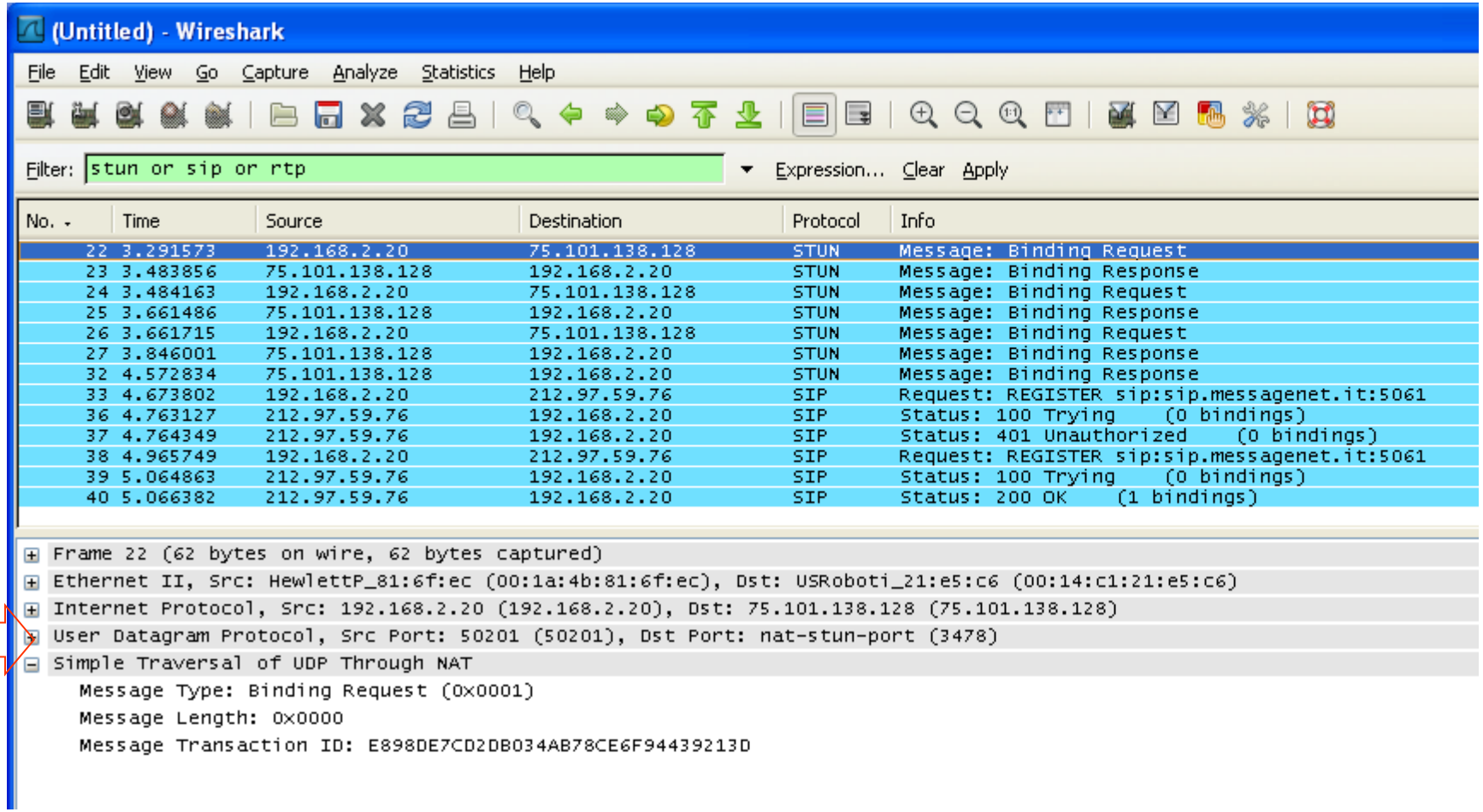
Server side management:



Interaction with other protocols

VoIP / SIP Example 8/13

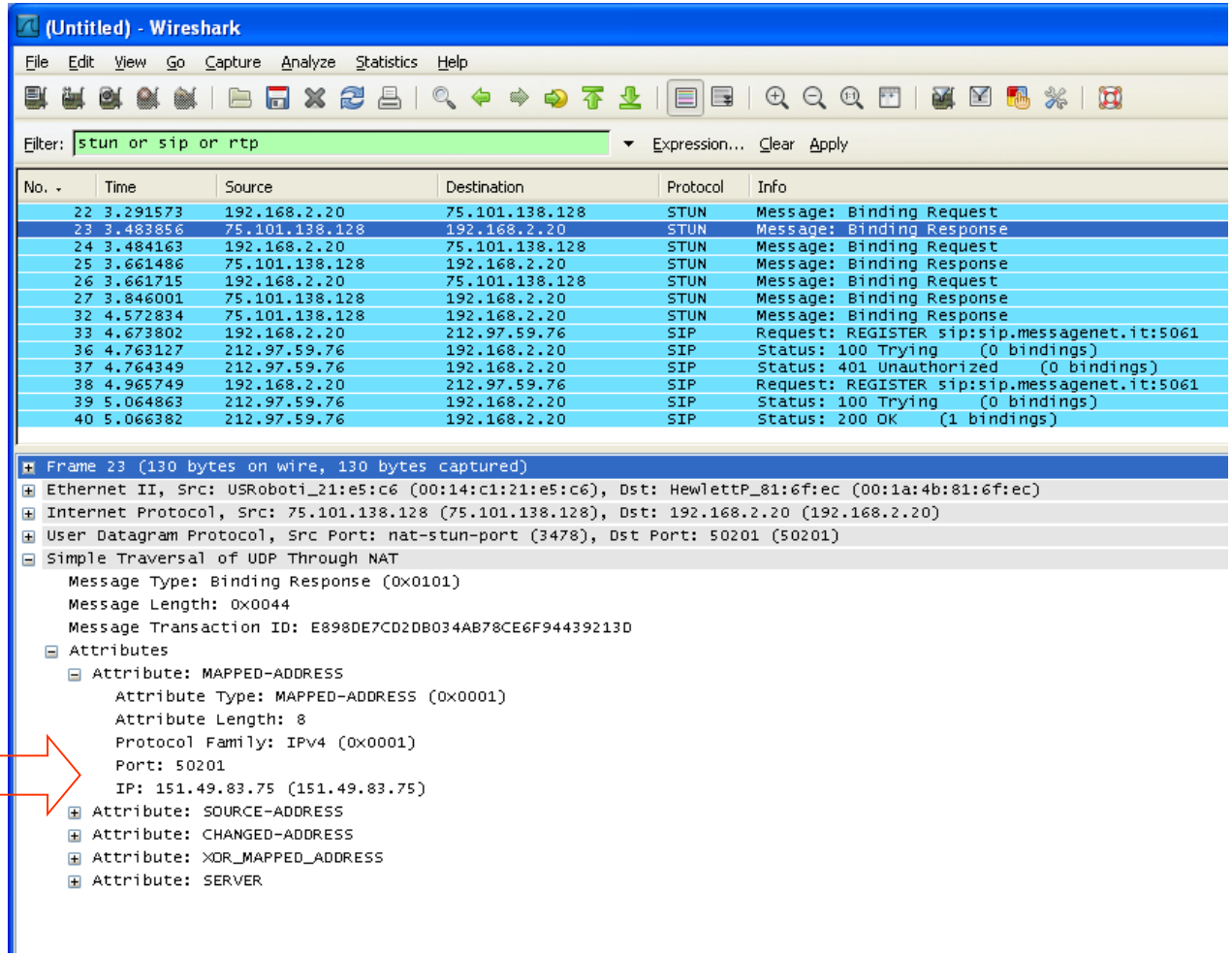
Client side management:



Interaction with other protocols

VoIP / SIP Example 9/13

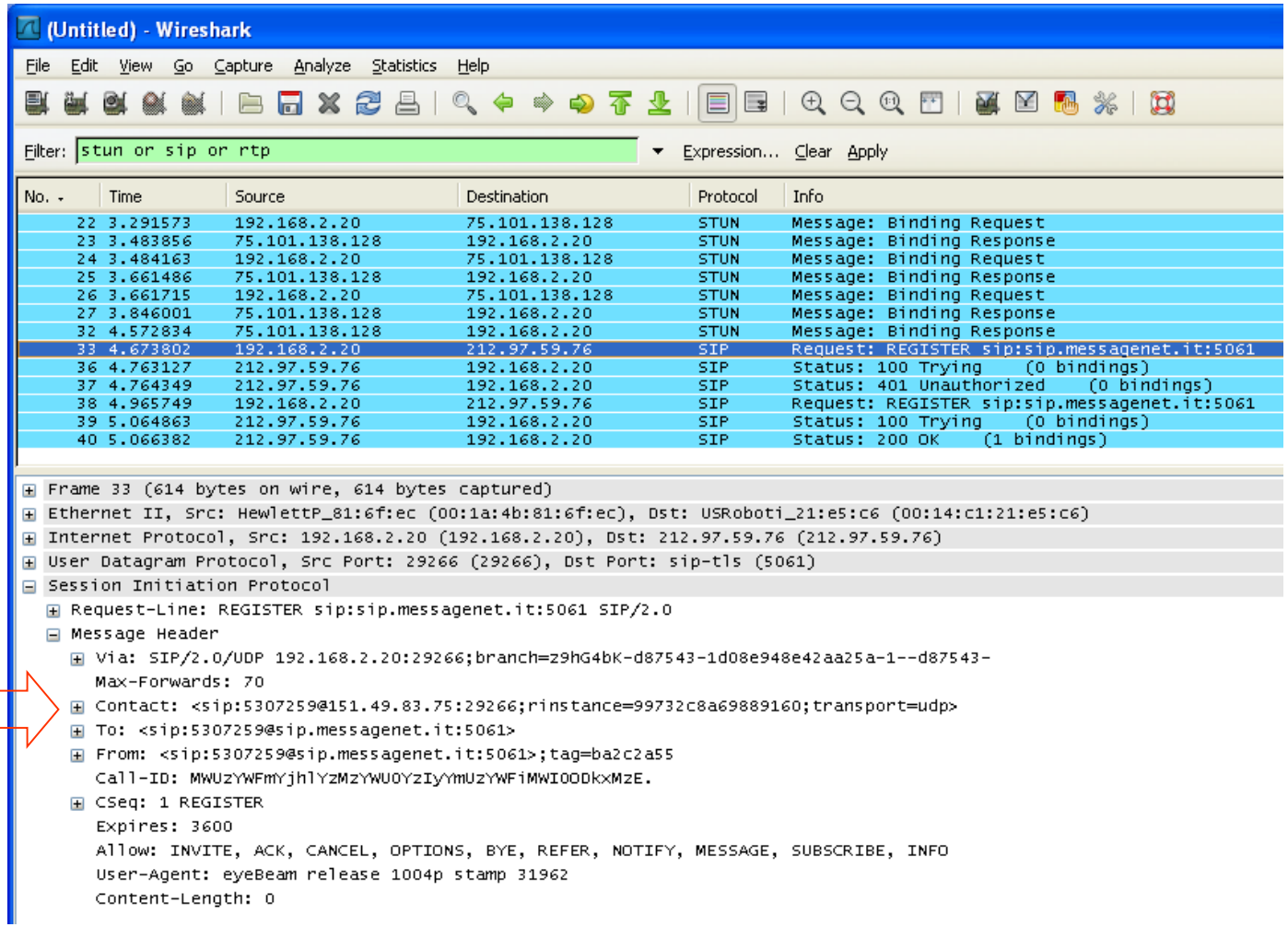
Client side management:



Interaction with other protocols

VoIP / SIP Example 10/13

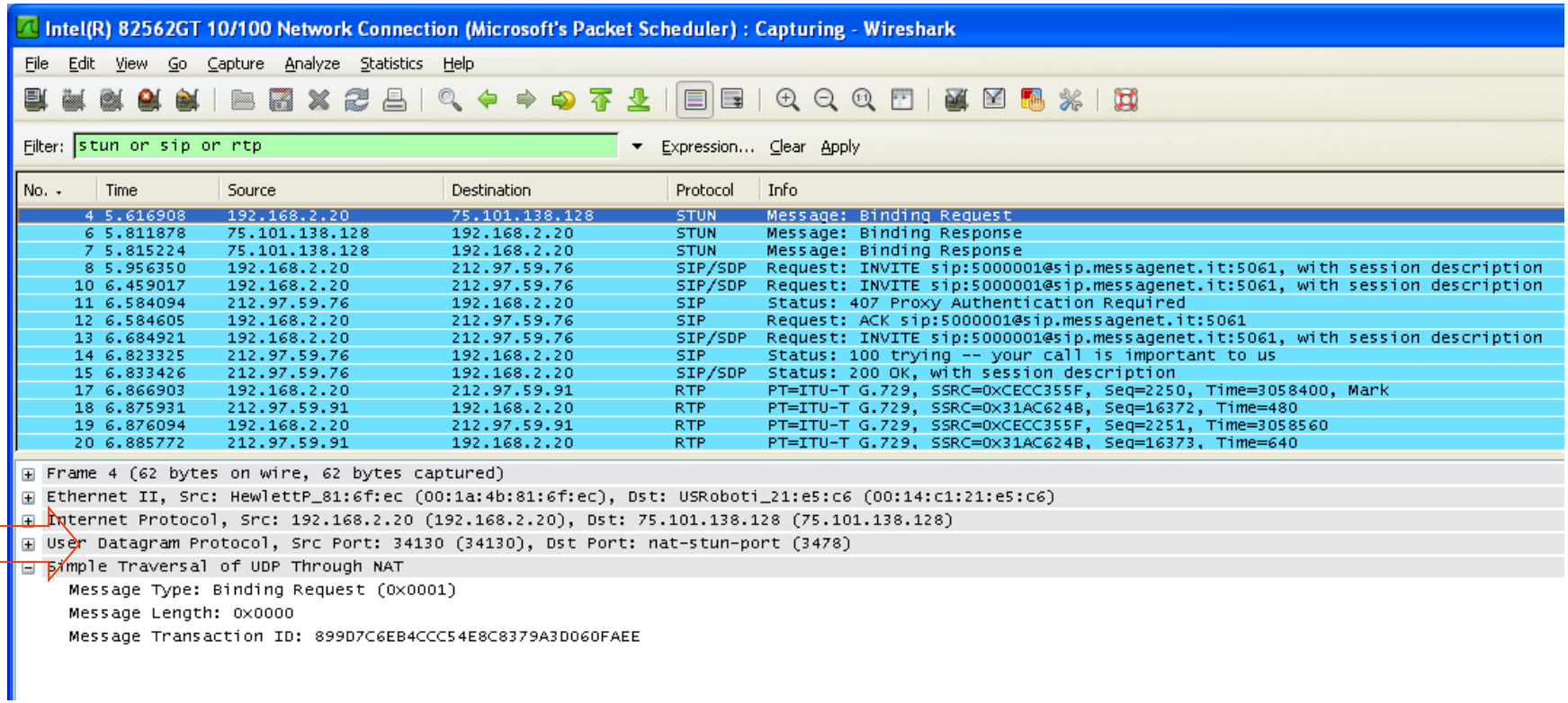
Client side management:



Interaction with other protocols

VoIP / SIP Example 11/13

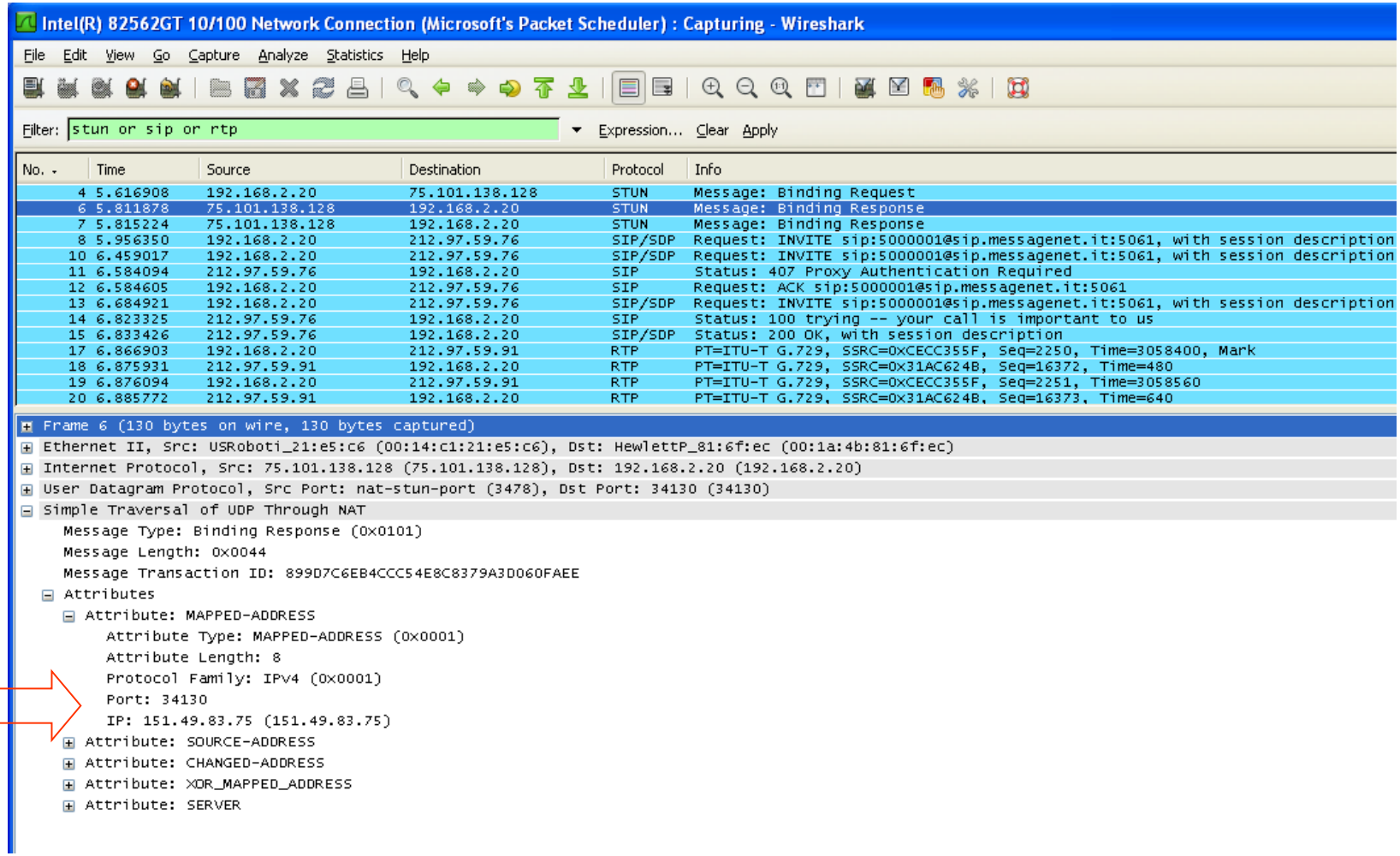
Client side management:



Interaction with other protocols

VoIP / SIP Example 12/13

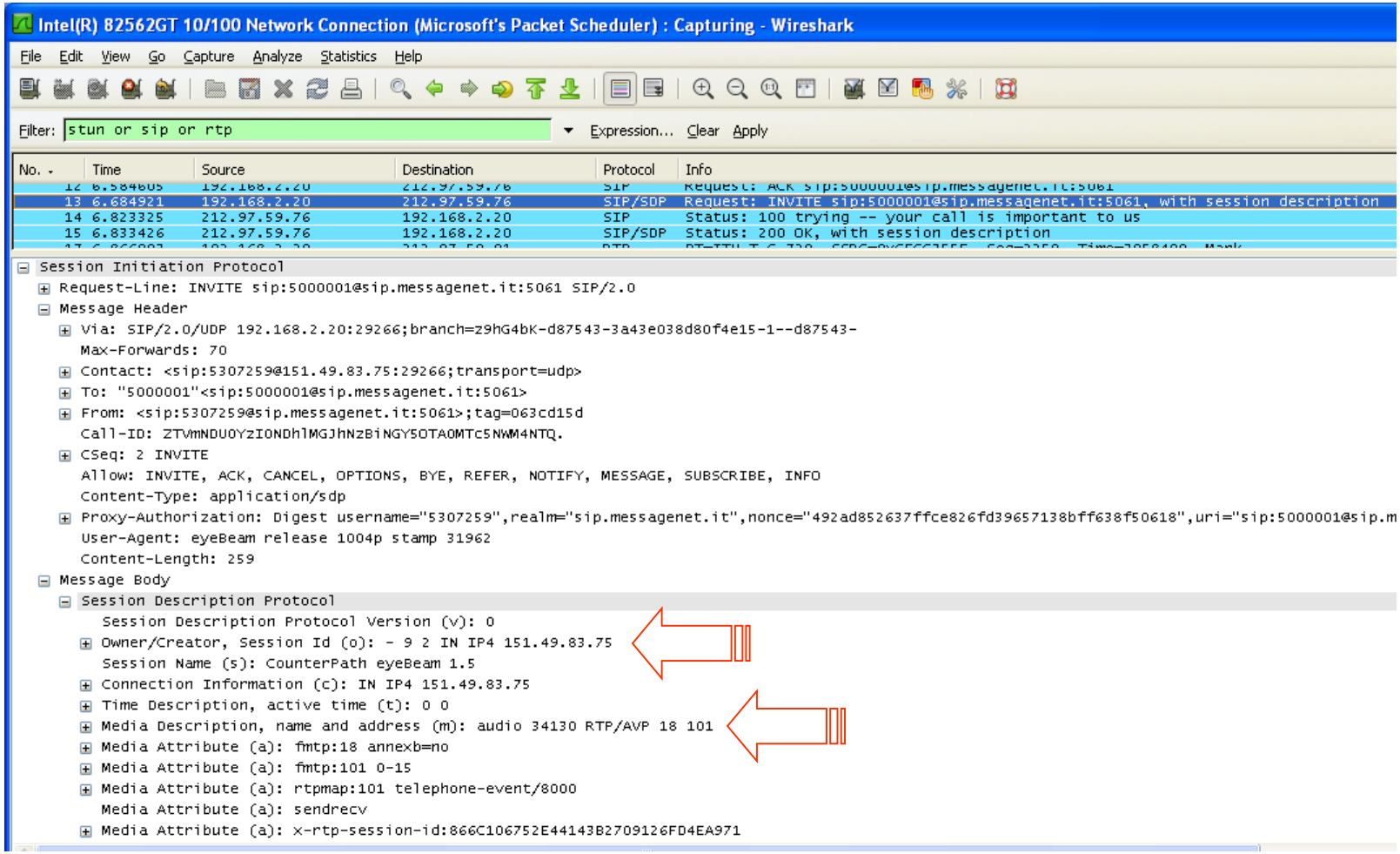
Client side management:



Interaction with other protocols

VoIP / SIP Example 13/13

Client side management:



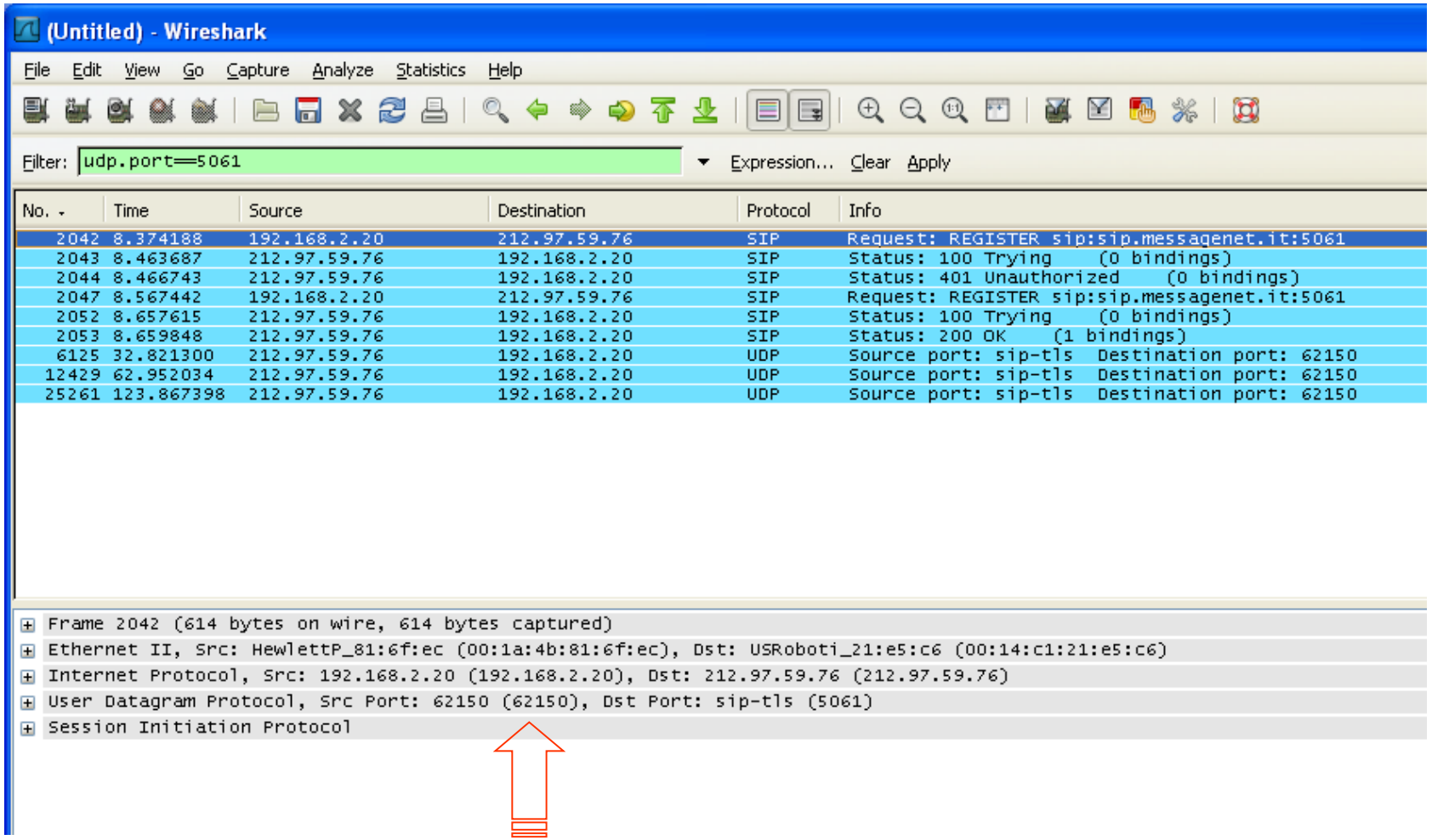
Interaction with other protocols:

binding refresh techniques

- Some server side applications requires client to be reachable for a certain time interval (even hours) over the <IP address, port> pair that it communicated
- What happens with NAT?
  - NAT binding typically has limited duration
  - In the absence of traffic, binding is removed as timer expires and <internal_address, internal_port> no longer reachable by sending packets to <external_address, external_port>
- approach: send keepalive packets to maintain binding
  - Packets can be sent by server or client
  - Typical interval between consecutive keepalive packets: 30secs.

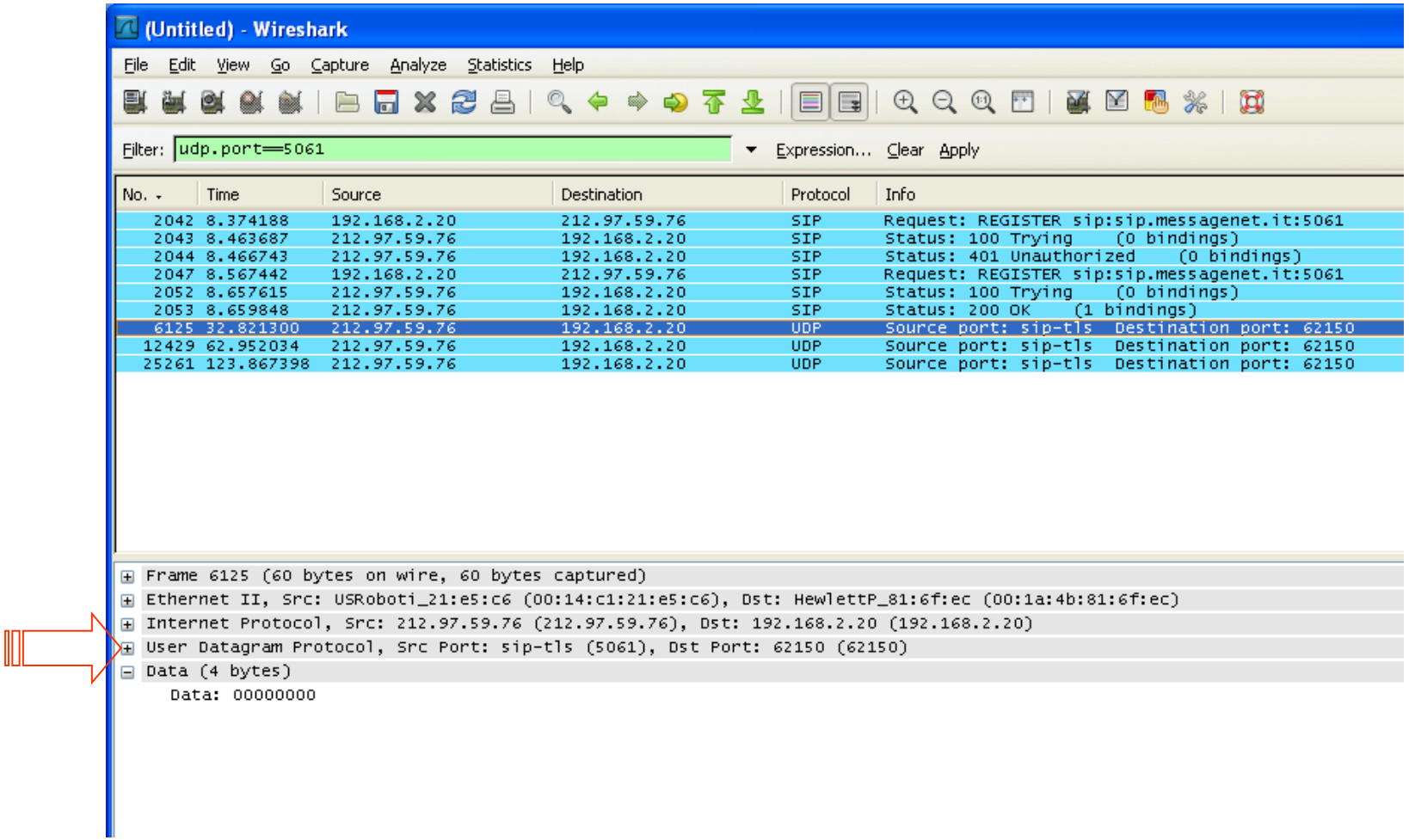
Interaction with other protocols:

Binding refresh techniques: Example 1/2



Interaction with other protocols:

Binding refresh techniques: Example 2/2



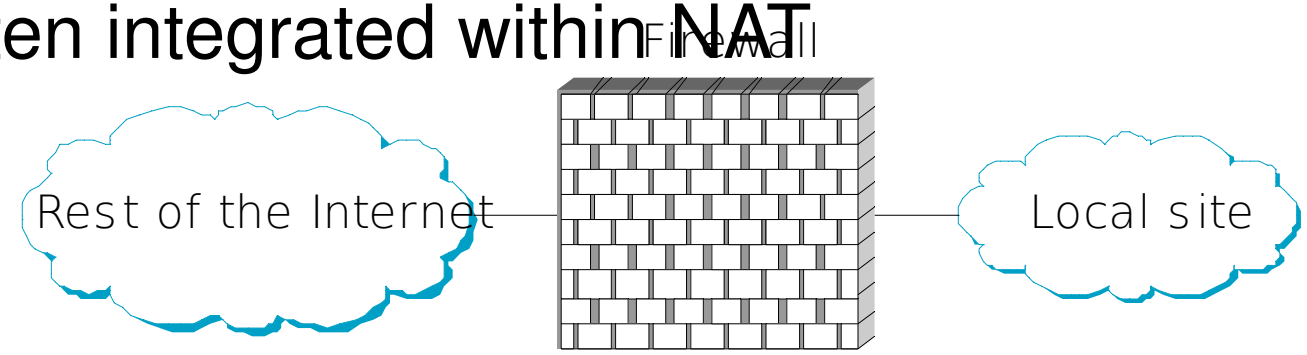
Further aspects

- Binding
  - NAT binding behaviour
  - Port binding behaviour
  - Binding timer refresh
  - Filtering
- Interaction with other protocols
  - STUN
- Basic reference:
  - http://www.cisco.com/web/about/ac123/ac147/archived_issues/ipj_7-3/anatomy.html

Firewalls

Firewall

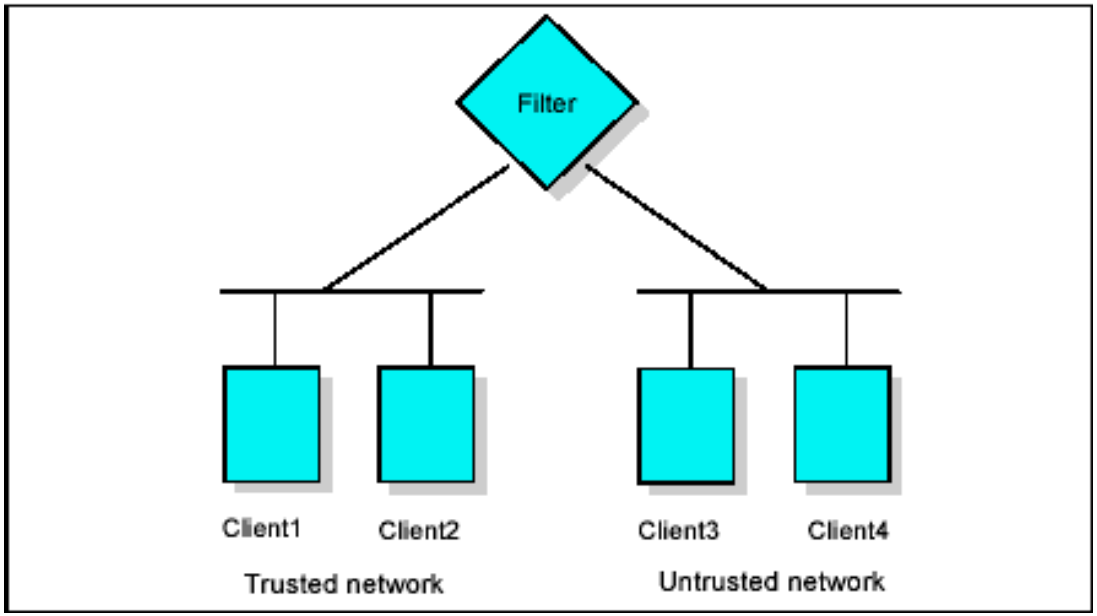
- Router with augmented functionalities, at the border between a private network and the Internet
- Forwards/filters traversing packets
- Allows implementation of centralized security policies
- Often integrated within NAT



Firewalls - components

- Packet-filtering router
- Application level gateway (proxy)
- Circuit level gateway
  - Often, but not necessarily, all present at the same time

Packet filtering router

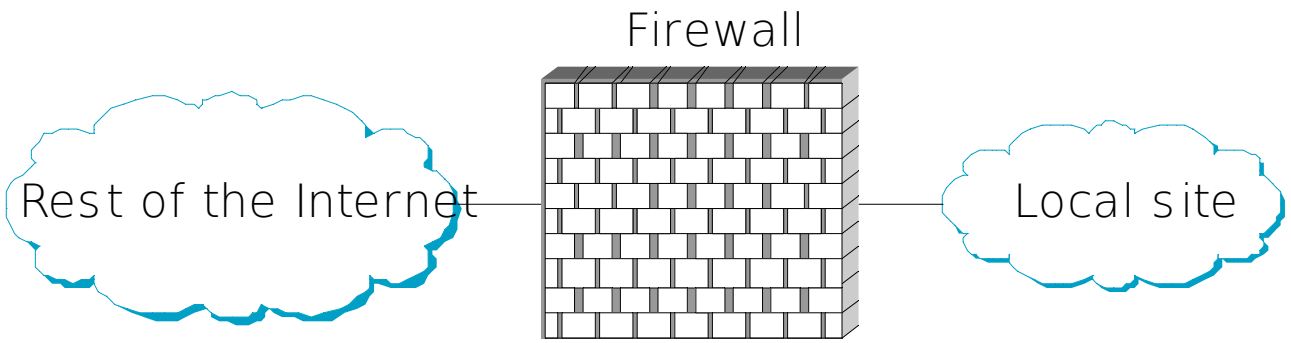


Picture from IBM© Redbook on TCP/IP

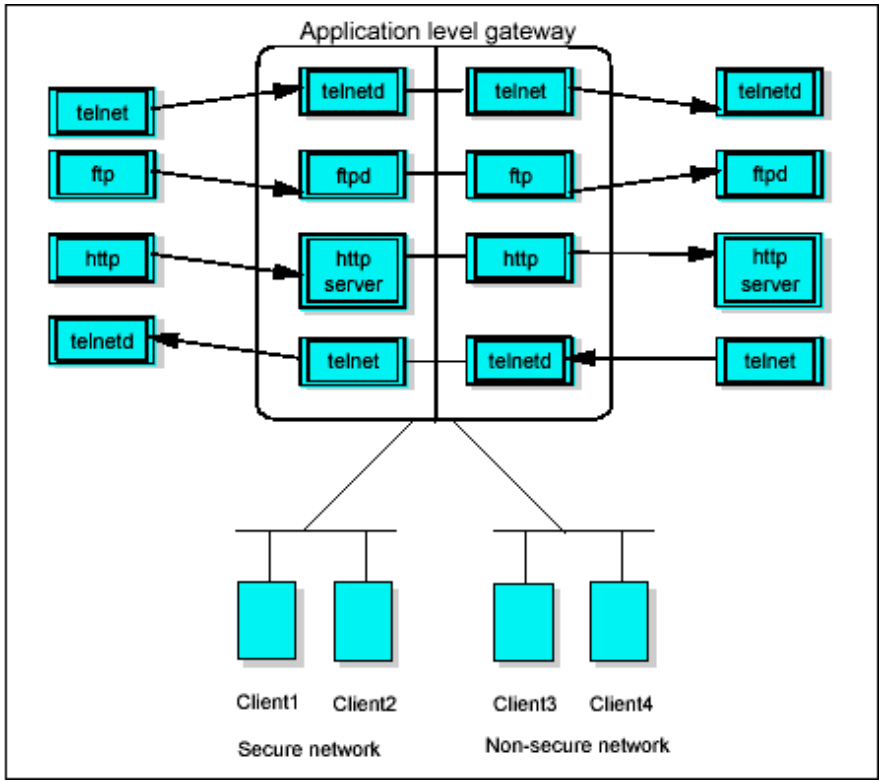
- Filtering based on IP datagrams' headers
  - IP source/destination
  - UDP/TCP Source/destination port
  - ICMP message type
  - Payload (UDP, TCP, IP tunnel…)

Example: IP addresses and port numbers

- Filter-Based Solution
  - Firewall has filtering table
  - Typically, every entry is a 4-tuple (SourceIP, SourcePort, DestIP, DestPort)
    - (192.12.13/24, *, 128.7.6.5, 80)
    - (*,*, 128.7.6.5, 80)
  - default: forward or drop



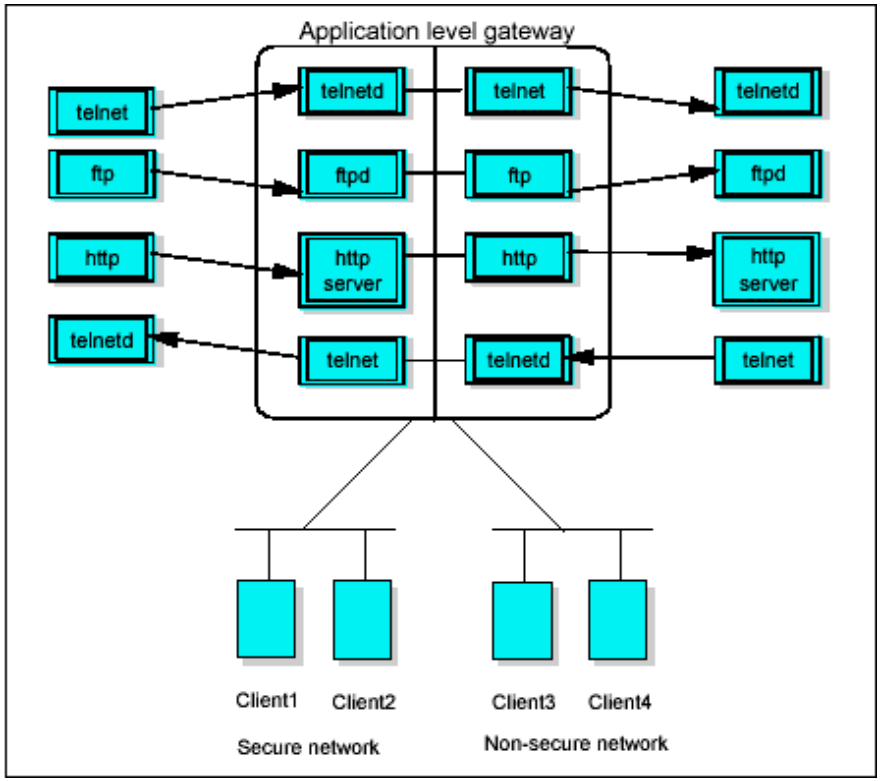
Application Gateway



Picture from IBM© Redbook on TCP/IP

- One proxy for every application
- Proxy behaves as a server to *local* client and as client towards remote server

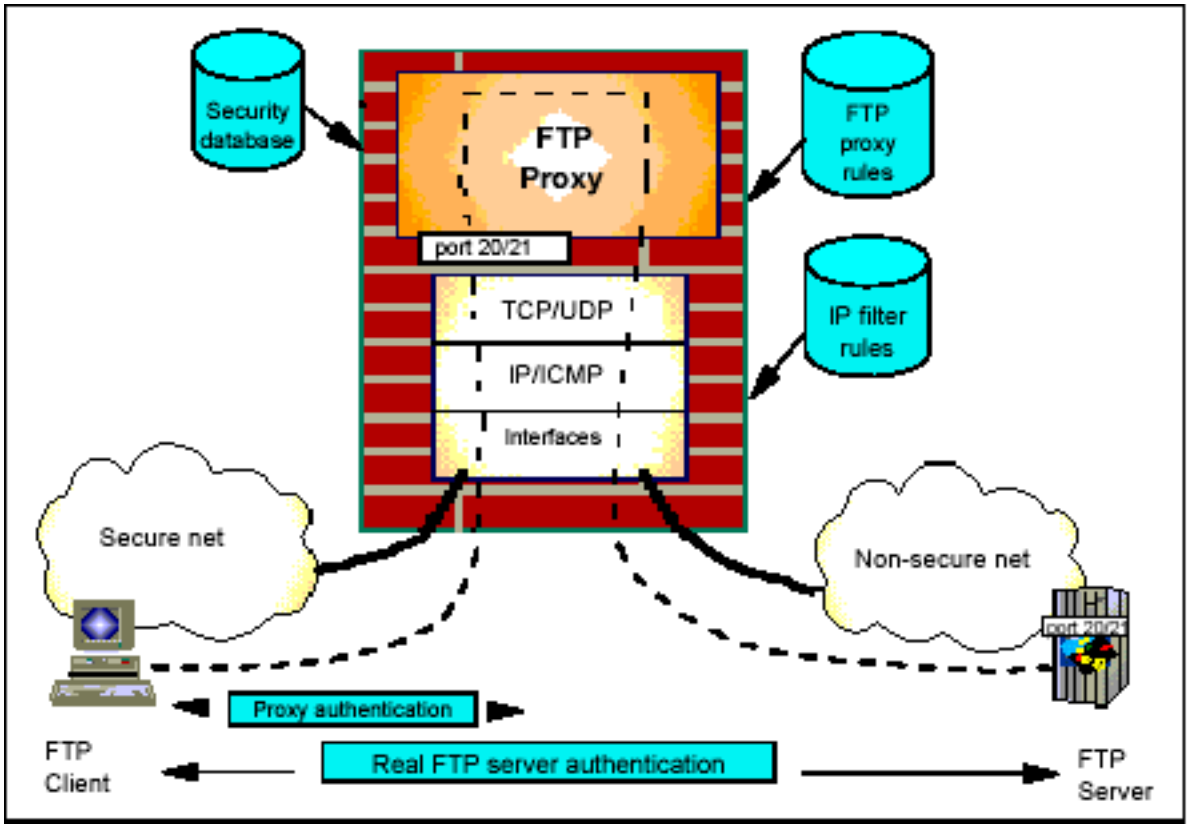
Application Gateway/cont.



Picture from IBM© Redbook on TCP/IP

- Possible to disable IP traffic between internal network and the Internet but …
- … only applications implemented at proxy can interact with external nodes

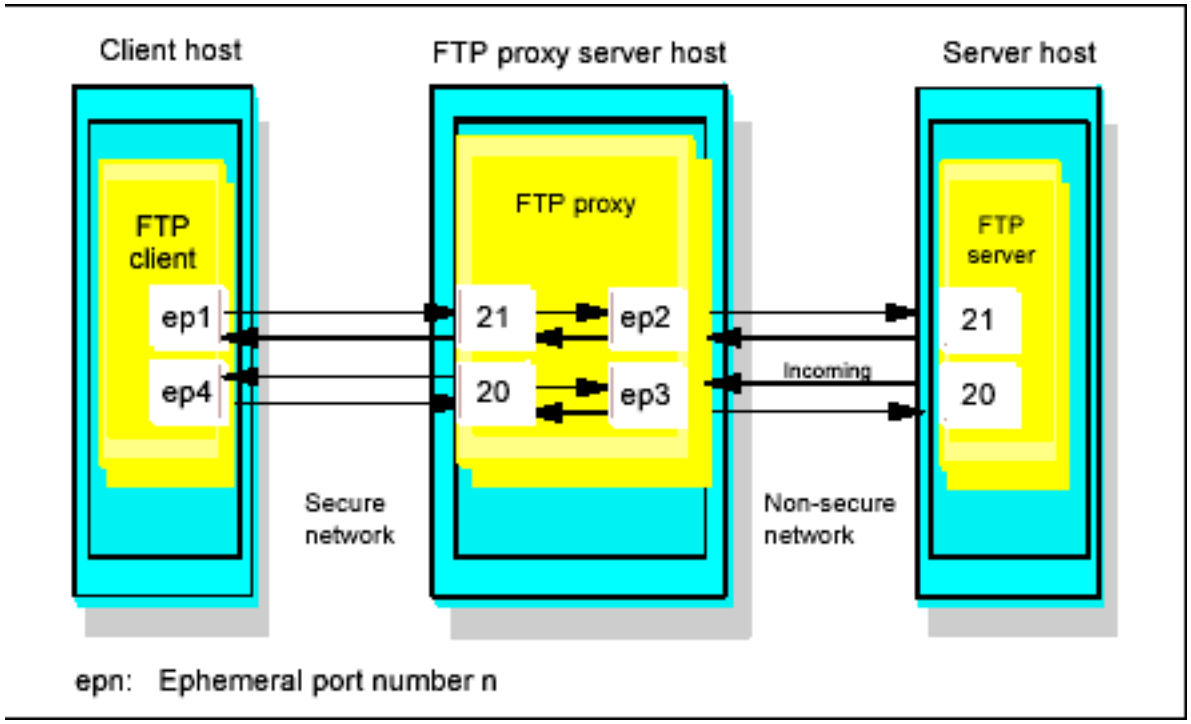
Example - FTP



Picture from IBM© Redbook on TCP/IP

- Necessary to modify FTP Sw
  - Necessary to implement proxy traversal

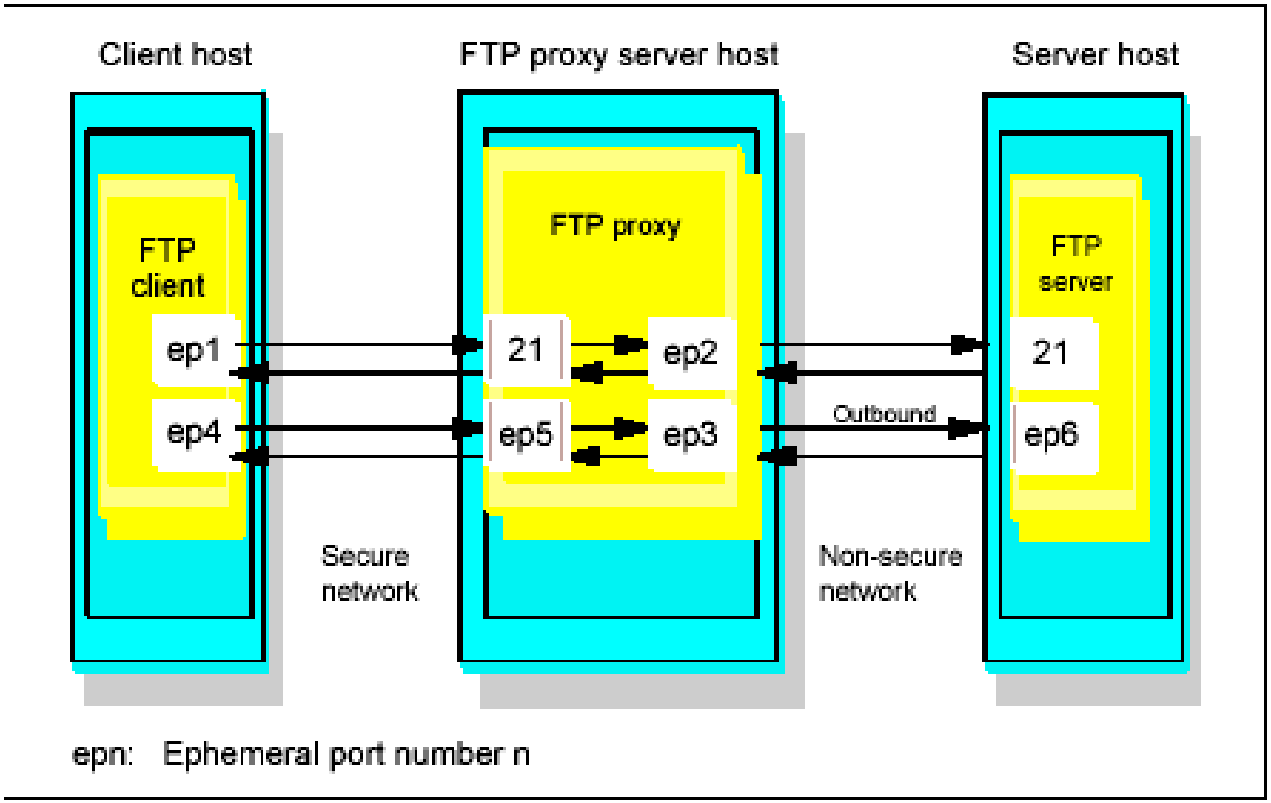
FTP/cont.



Picture from IBM© Redbook on TCP/IP

- Active mode operation
  - Possible attacks - which ones?

A possible fix…



Picture from IBM© Redbook on TCP/IP

- Passive mode or Firewall-friendly FTP

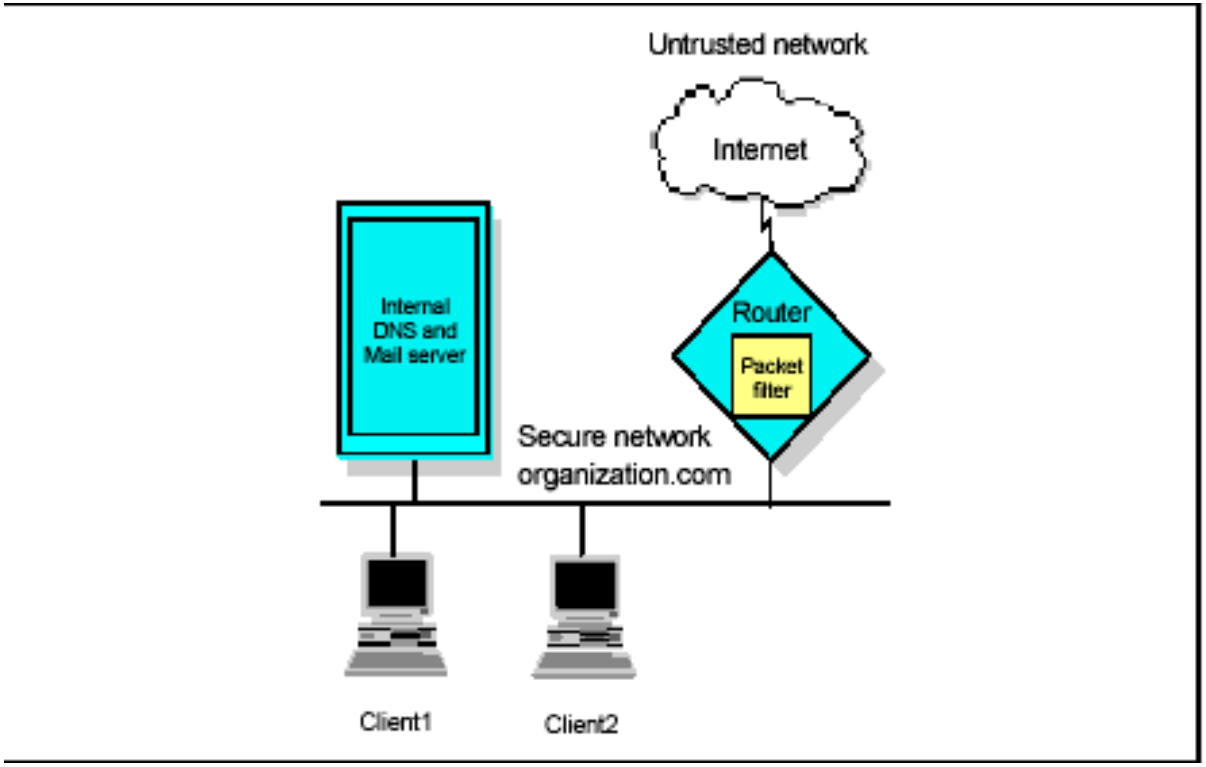
Circuit Level Gateway

- In practice: a transport-layer proxy (UDP or TCP)
- Allows all applications that satisfy desired requirements at the transport layer
  - Example: allow all and only TCP traffic; for UDP allow only DNS traffic
  - All TCP-based applications work without changes, as well as DNS
- No packet filtering

Firewalls

Firewall architectures

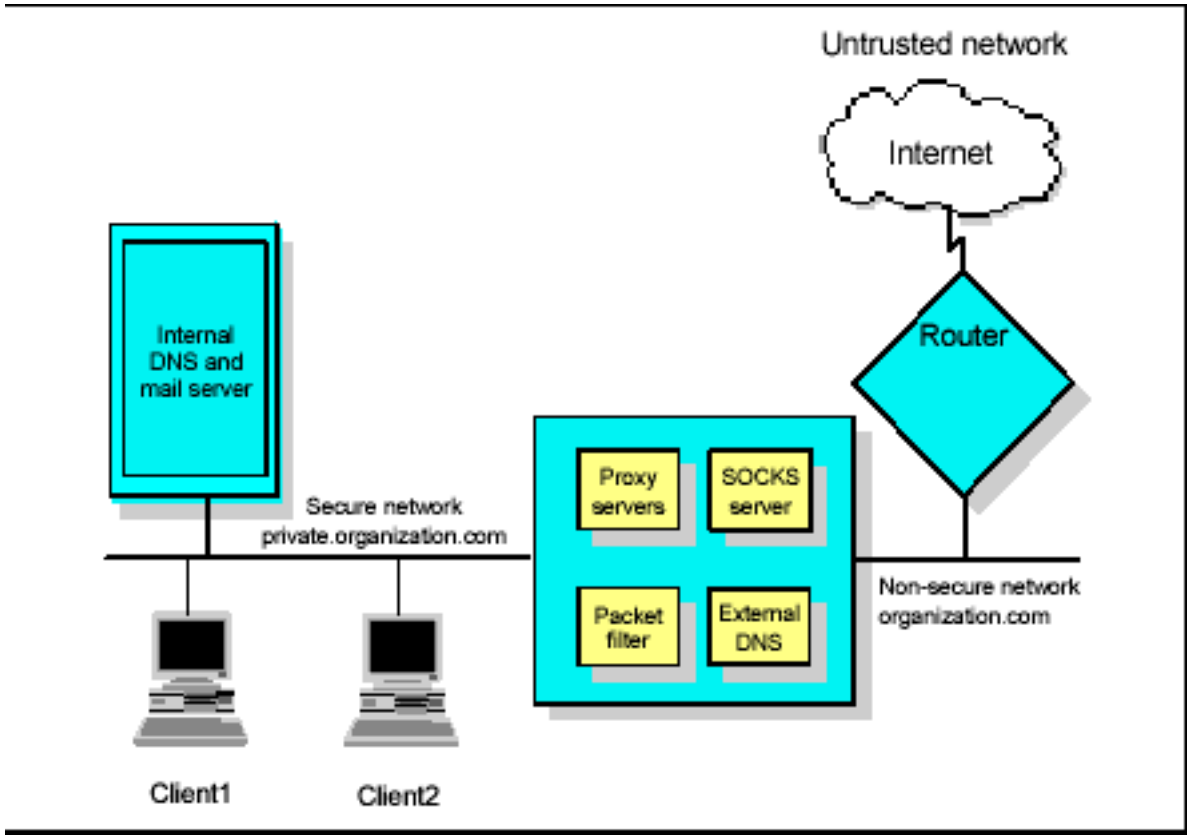
Packet filtering



Picture from IBM© Redbook on TCP/IP

- Cheap
  - Simply a router performing packet filtering
- **Rules: usually, what is not explicitely allowed is forbidden**

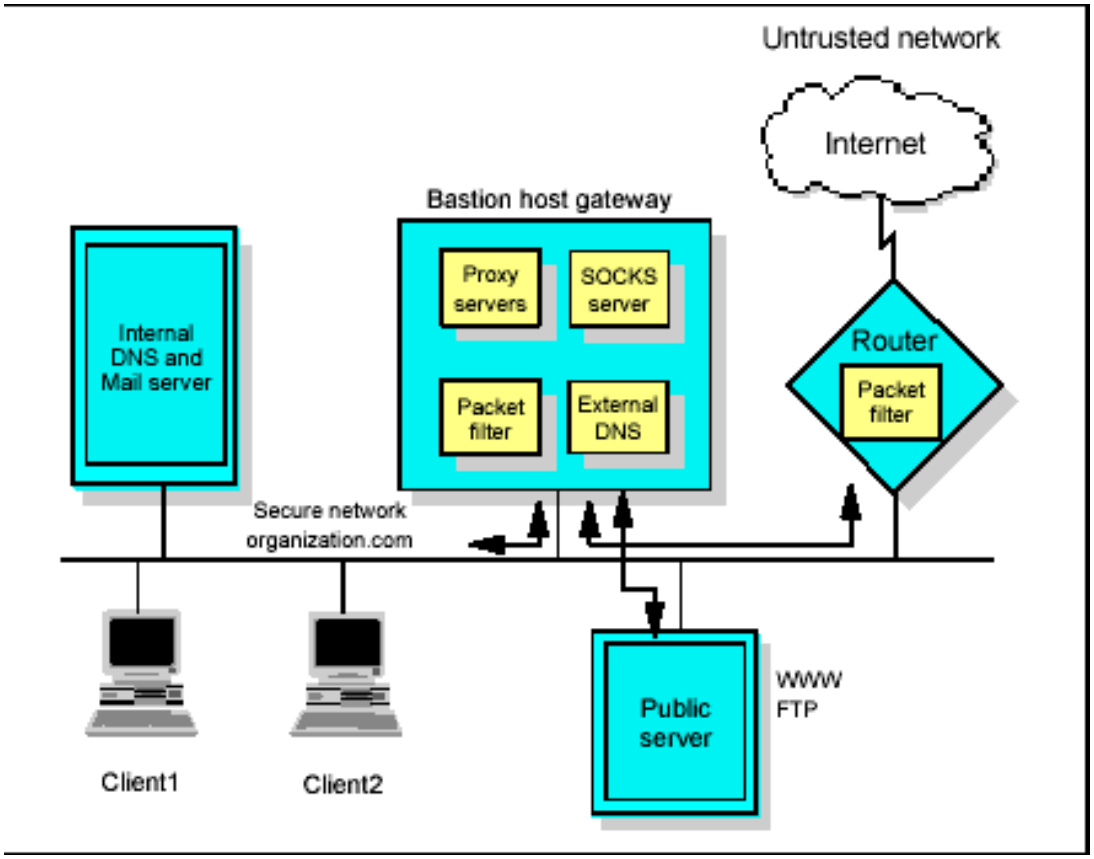
Dual-homed



Picture from IBM© Redbook on TCP/IP

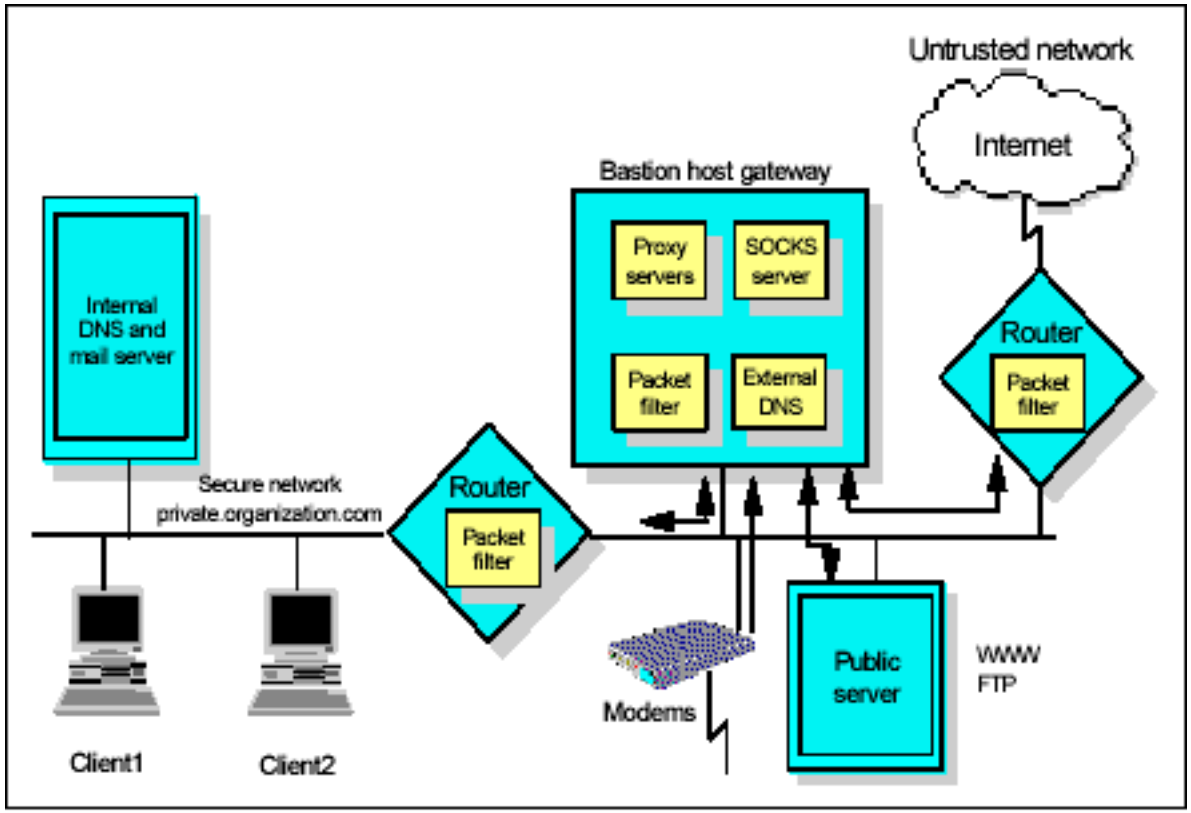
- At least 2 network interfaces (and 2 IP addresses)
- **Everything which is not allowed is forbidden**
- No direct IP traffic between local network and the Internet possible
- E.g.: proxy firewall

Screened host



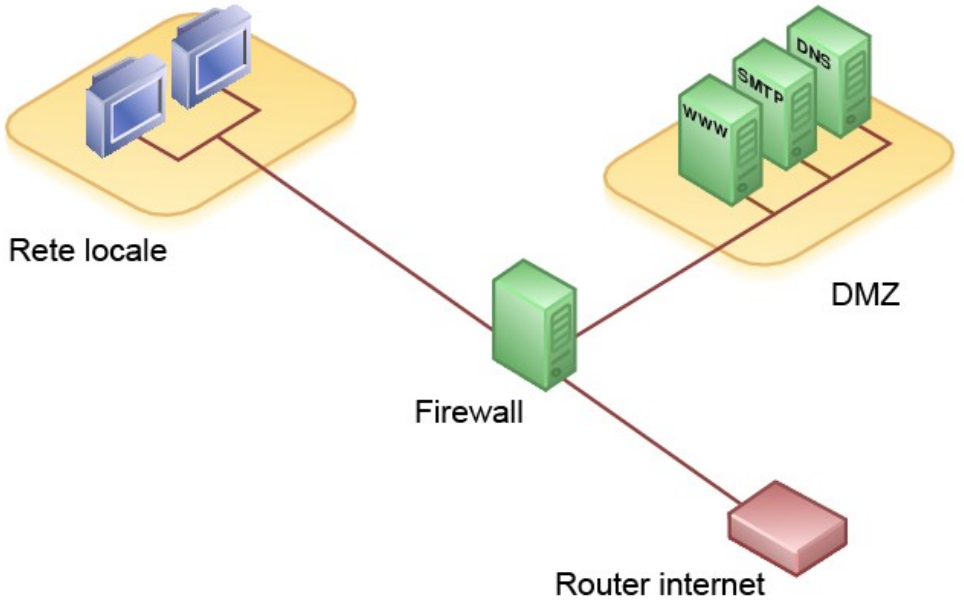
- Application GW (bastion) and packet filter
- Server on same subnet as bastion GW
- Access to server from outside *only* through bastion GW
  - **How do you implement this policy**?
  - **How do you screen server from internal traffic**?

DeMilitarized Zone



- Both internal and external traffic through Bastion GW
- Maximum level of security

DeM**ilitarized** **Z**one



- DMZ can be created by defining distinct policies on one or more firewalls

References

- Tunneling
  - GRE RFCs 2784 and 2890: http://tools.ietf.org/html/rfc2784 (rfc2890)
  - Overview of Virtual Private Networks: http://www.vpnc.org/vpn-technologies.html
- NAT
  - http://www.tcpipguide.com/free/t_IPNetworkAddressTranslationNATProtocol.htm
  - NAT and STUN
    - http://www.cisco.com/web/about/ac123/ac147/archived_issues/ipj_7-3/anatomy.html
- Firewalls: IBM's TCP/IP Tutorial and Technical Overview. Available on-line
  - http://www.redbooks.ibm.com/redbooks/pdfs/gg243376.pdf