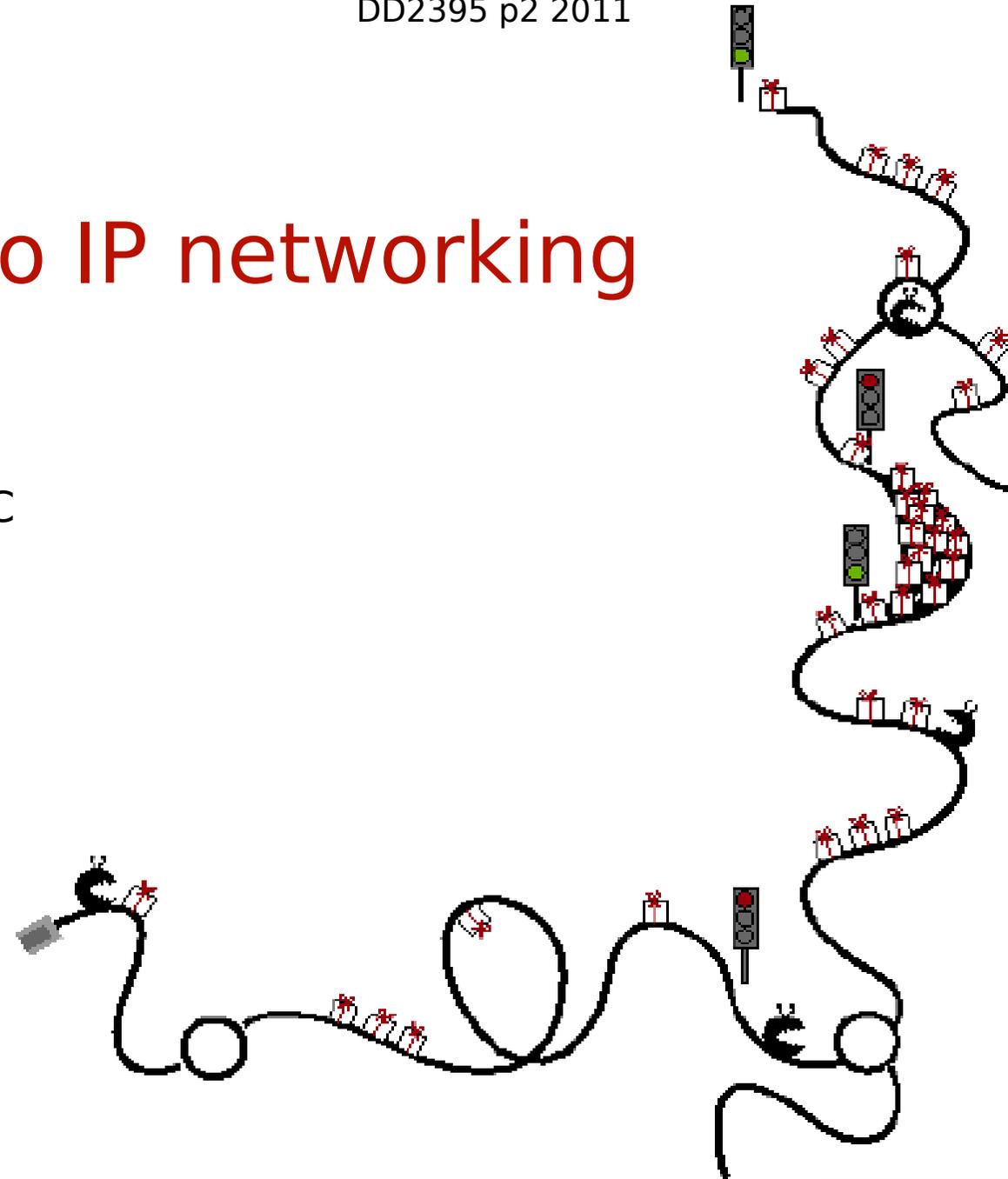


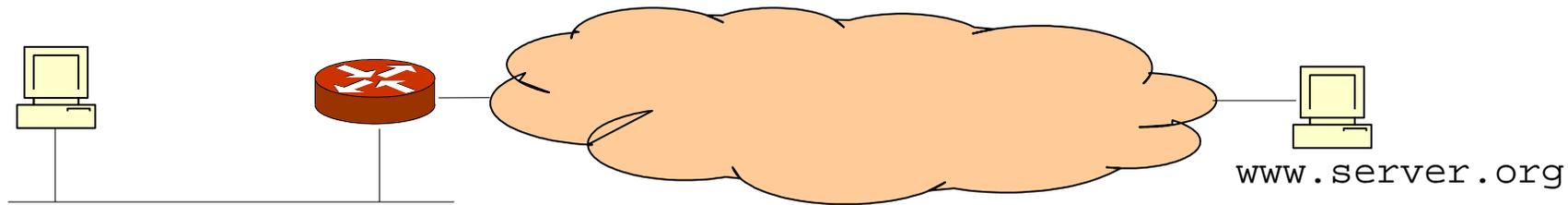
Introduction to IP networking



Olof Hagsand KTH CSC



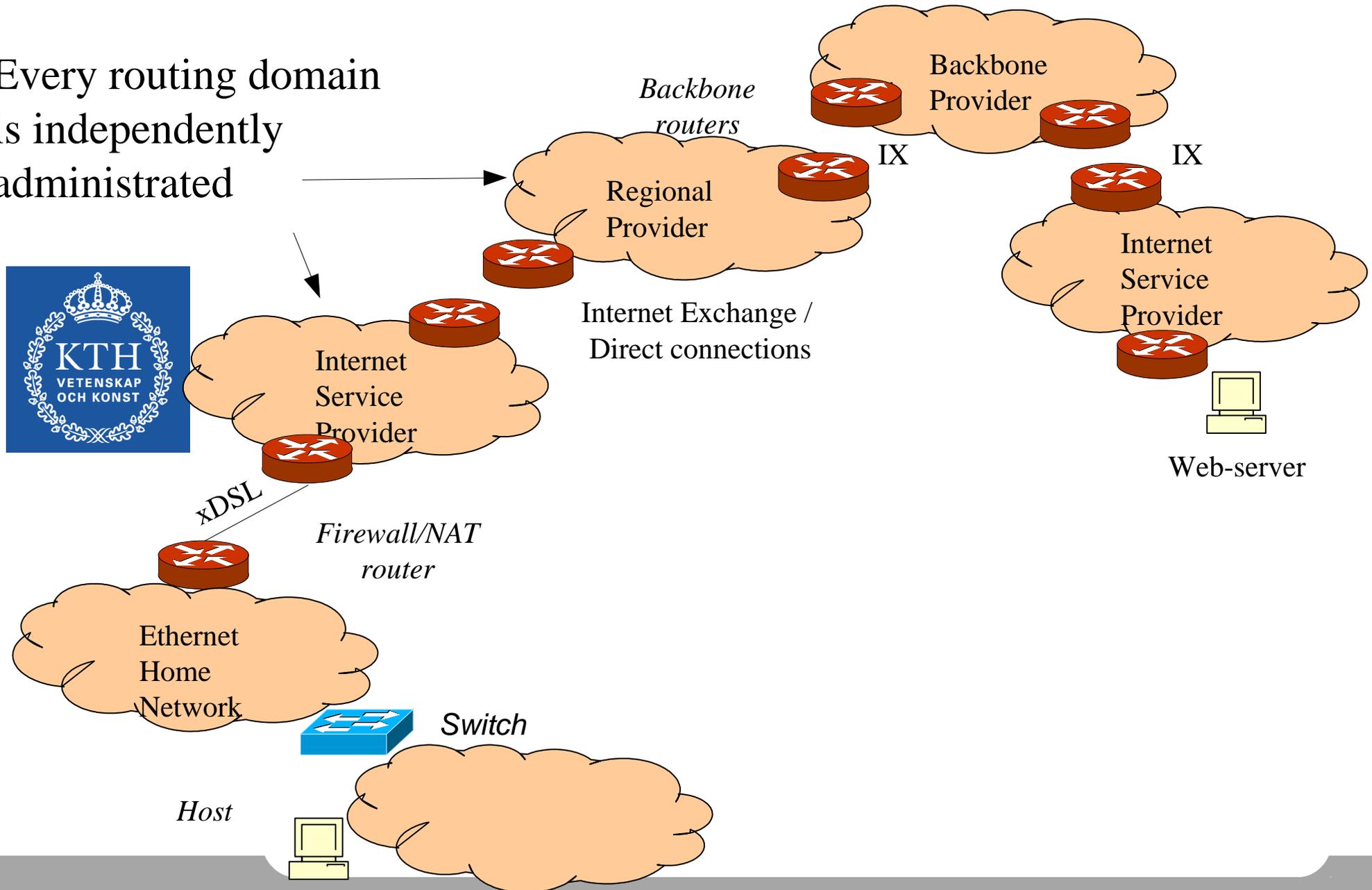
Example: Packet transfer



- An end host requests a web-page from a server via a local-area network
- The aim of this lecture is to give an overview of how this works in practice
- So that you can configure packet filters in the ip-tables lab
- Some protocols involved:
 - Ethernet, ARP, IP, TCP, DNS, HTTP

Internet architecture: Example

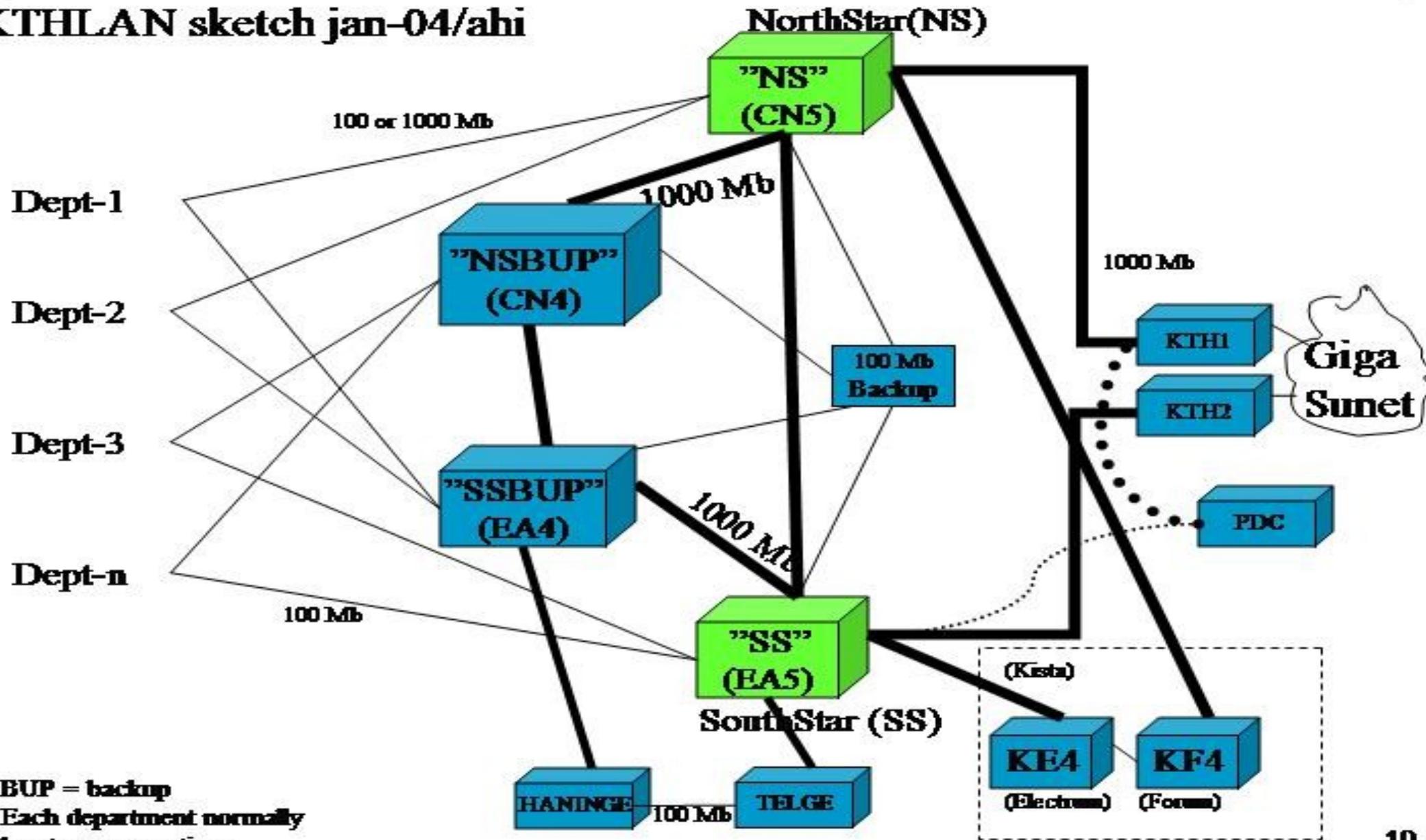
Every routing domain is independently administrated



Network example: KTH Intranet

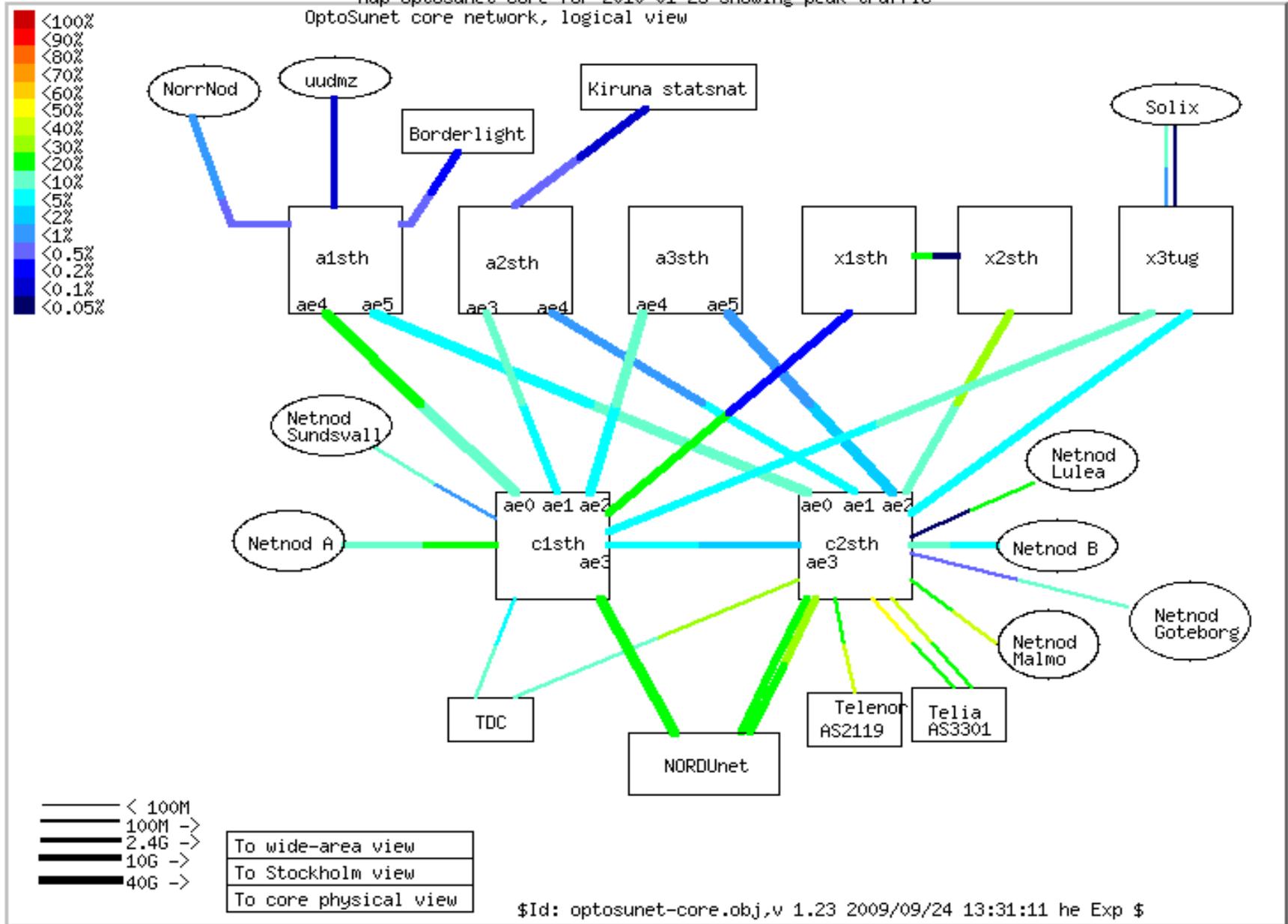
IT-enheten KTH

KTHLAN sketch jan-04/ahi

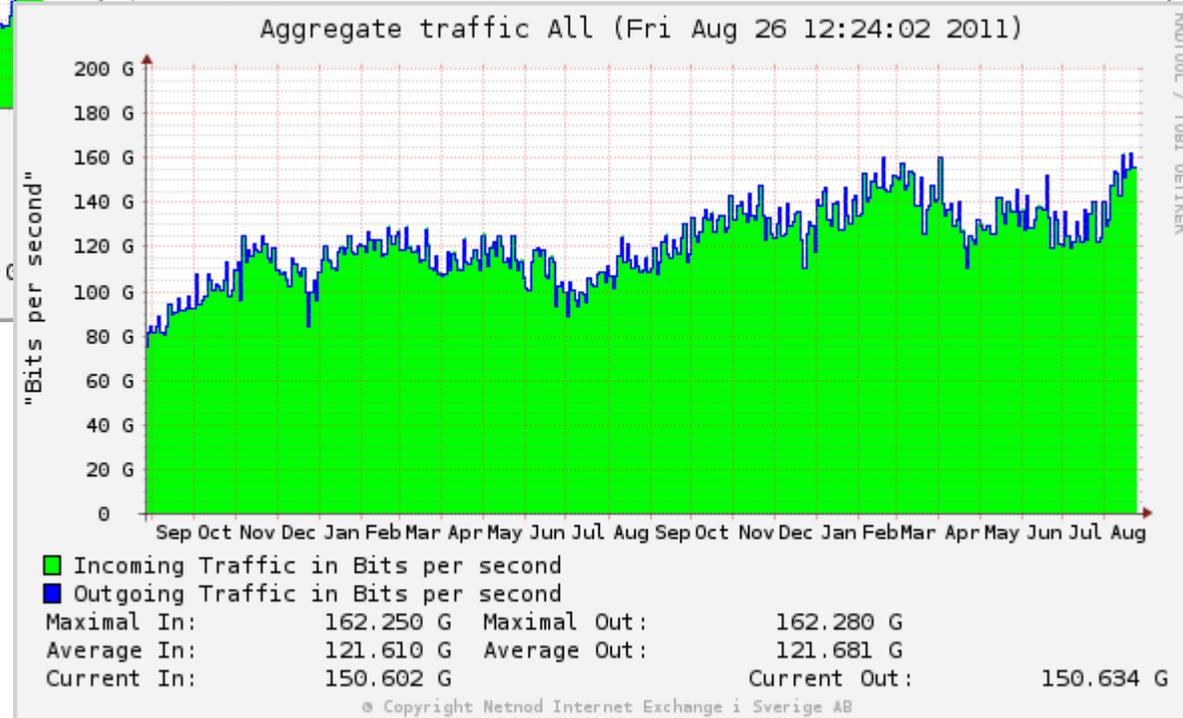
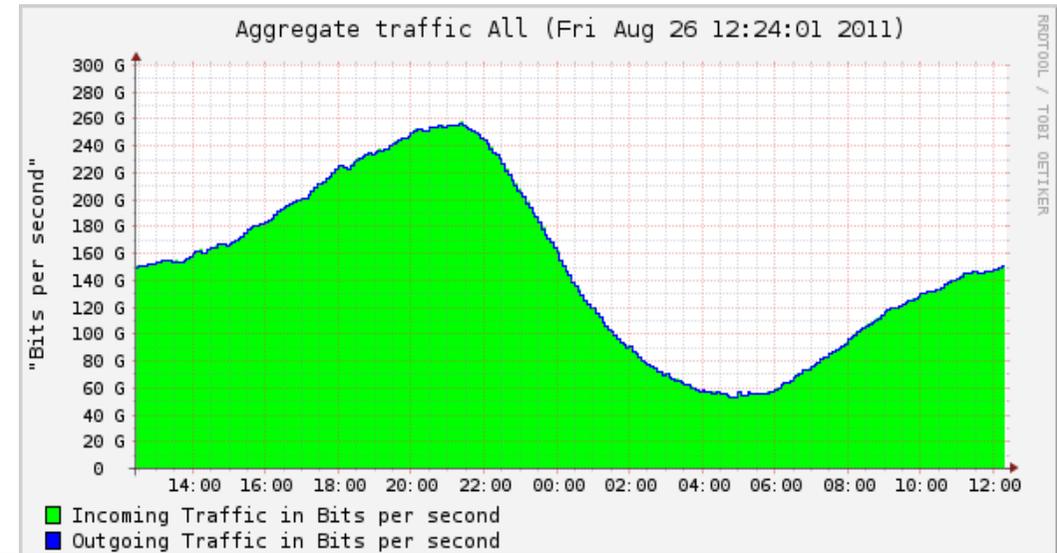
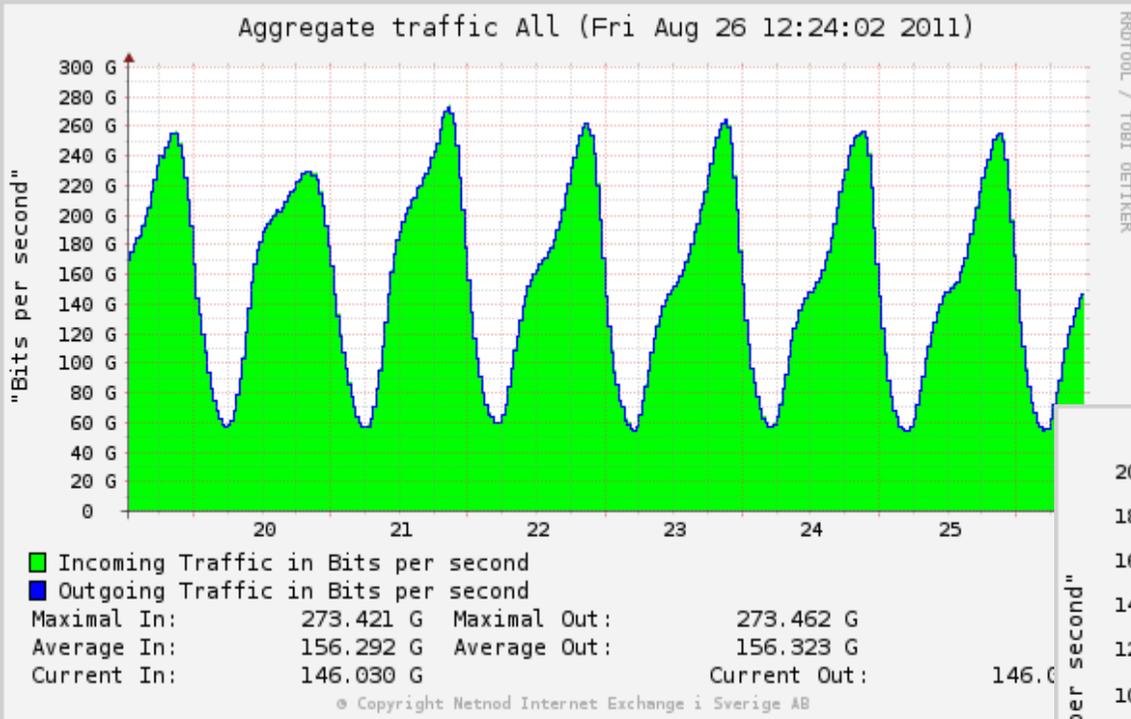


Network example: OptoSUNET core

Map optosunet-core for 2010-01-25 showing peak traffic
OptoSUNET core network, logical view



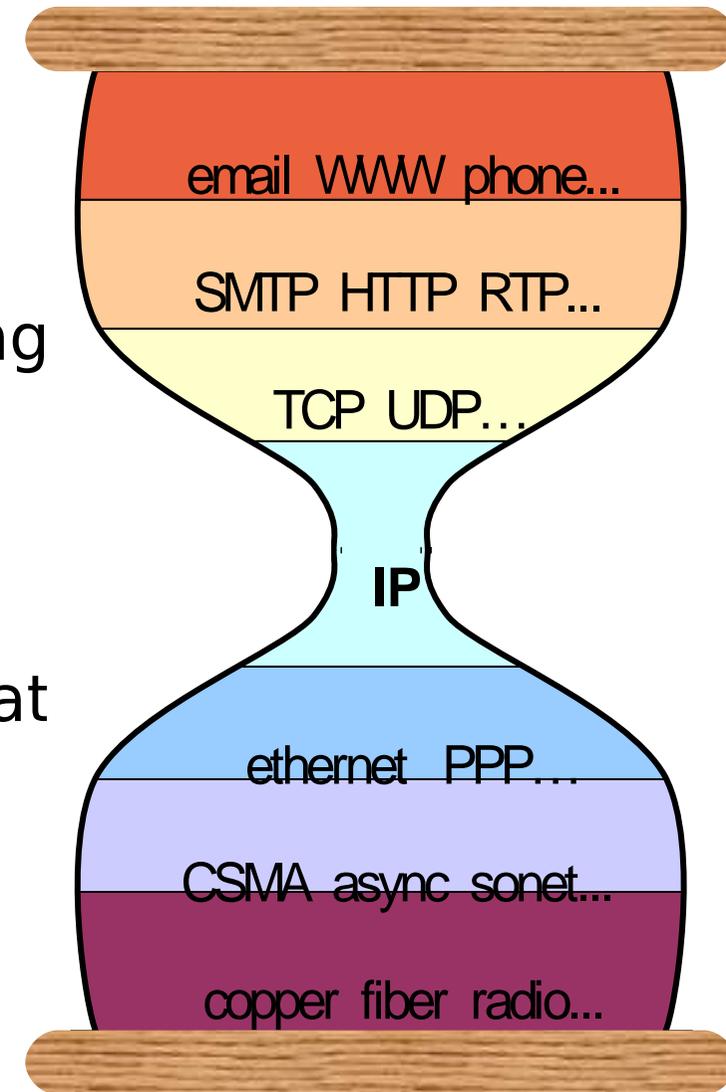
Netnod aggregated traffic



The Hourglass Model

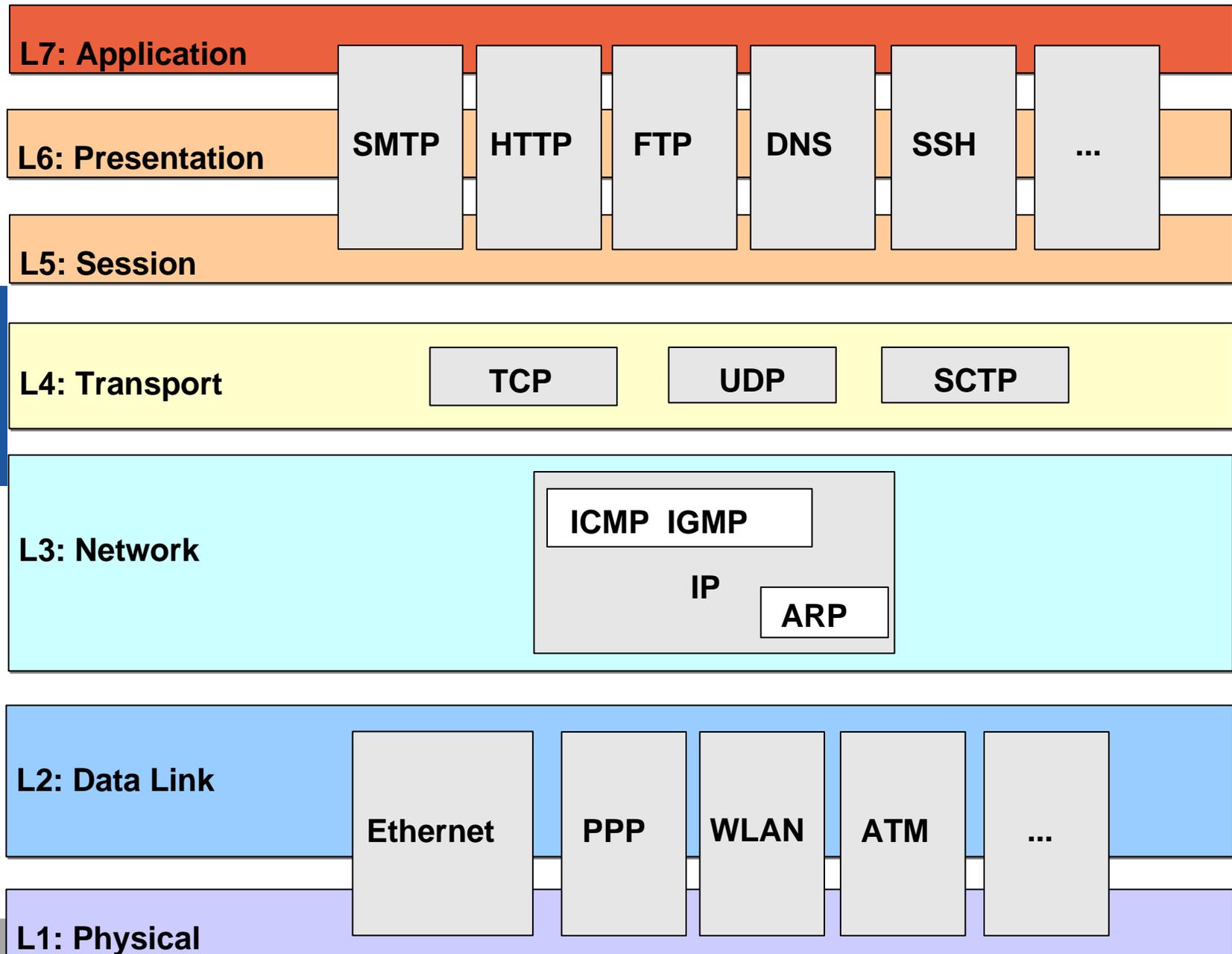


- Anything over IP – IP over anything
- All applications depend on IP
- IP runs over all networks
- IP is at the heart of all communication



From Steve Deering, 2000

The TCP/IP stack and OSI ref model



Encapsulation

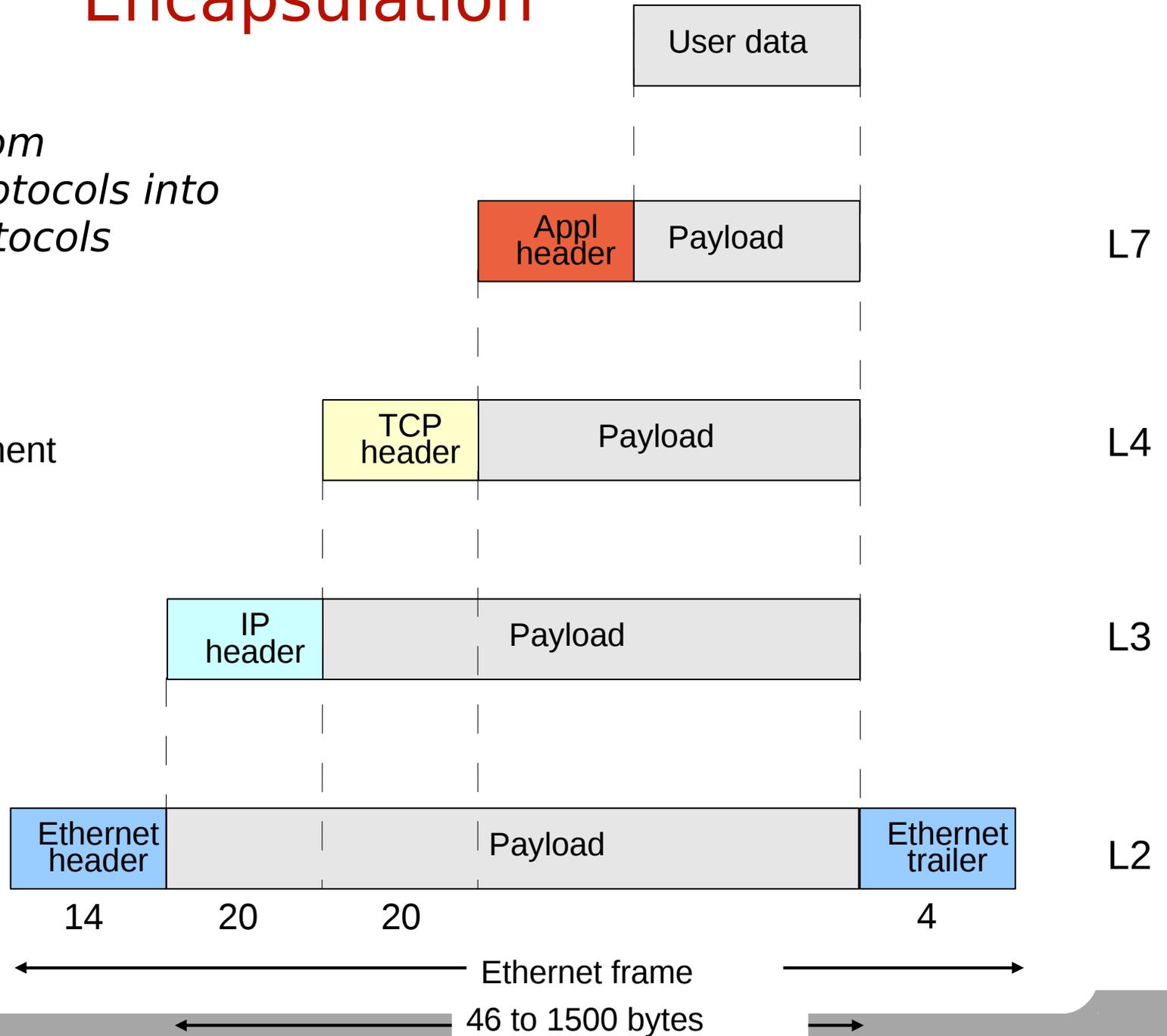
Include data from higher layer protocols into lower layer protocols using headers



Segment

Datagram

Frame



Link-level example: Ethernet

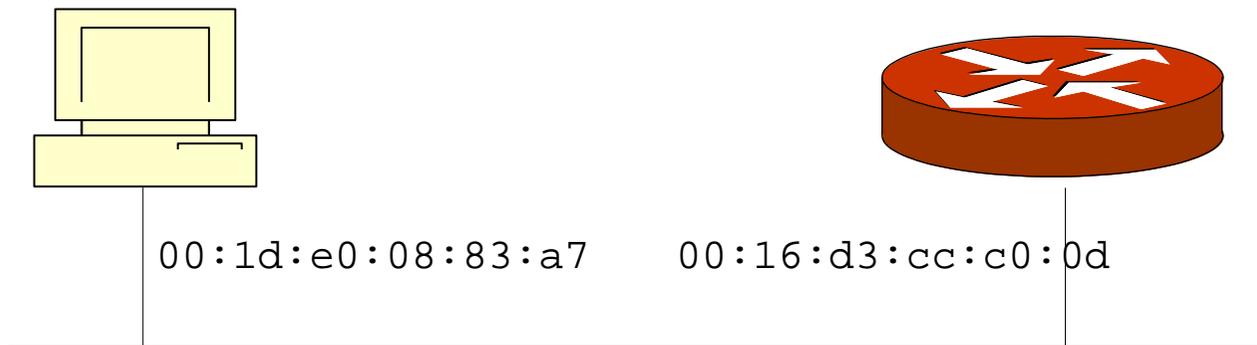
Ethernet frame:



- Ethernet is an example of a link-level protocol, that uses copper or fiber. WLAN is similar to Ethernet but runs on 'air'.
- One important task of the link-layer is *framing*
 - 'create packets from the underlying physics'
- Another is detecting bit errors (cyclic redundancy check) and addressing ('MAC' addresses)
- A MAC (IEEE 802) address has a “flat” structure – they cannot be aggregated into more abstract addresses: networks
- Typically only requested on a local-area network on a *directly connected* network
 - But a “local-area” network can nowadays be very large

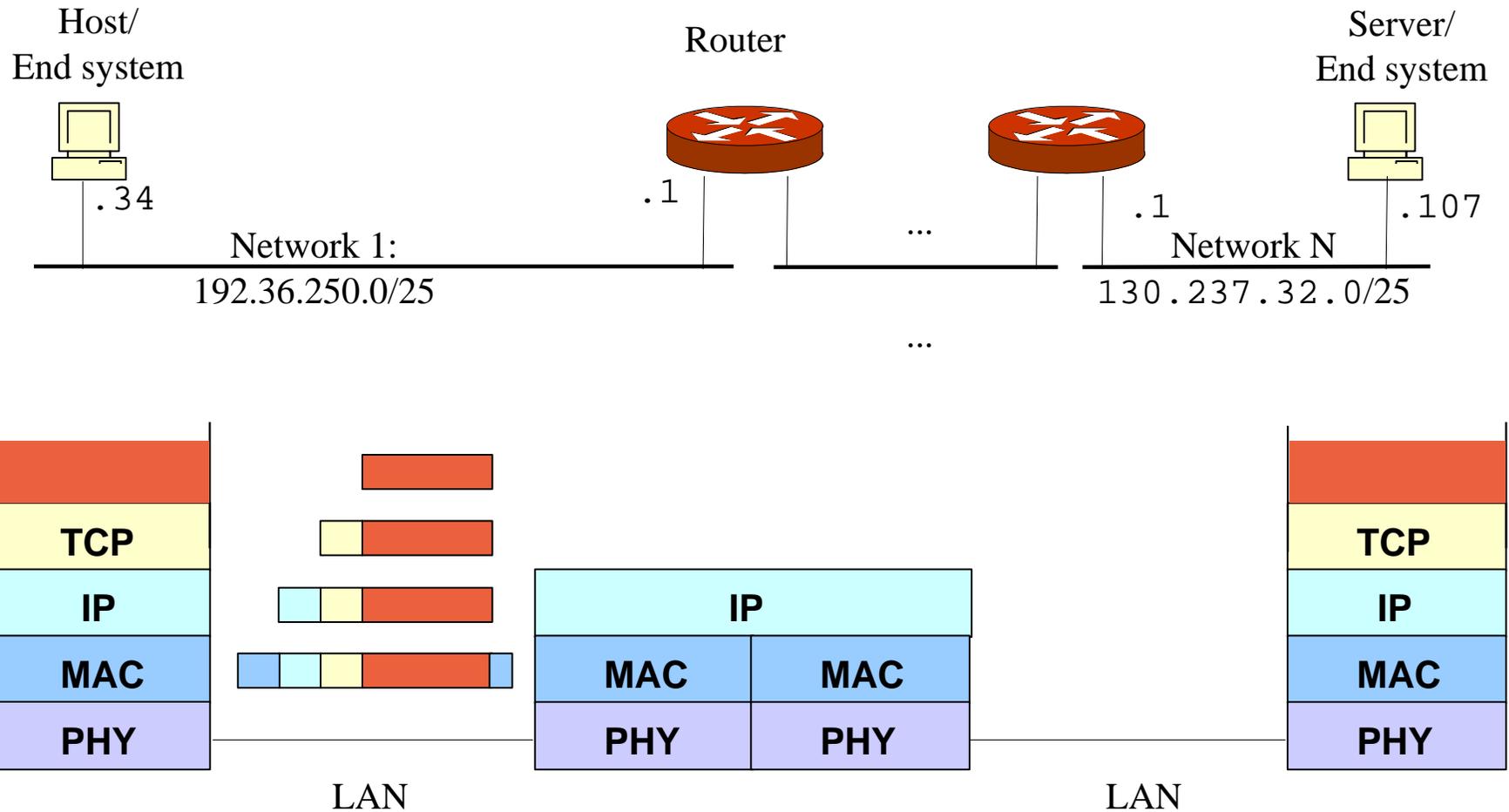
Ethernet /WLAN communication

- Using a link-level protocol, you can now communicate directly over a link



- But what about communicating over several hops?

Network layer and IP



- The network layer (IP) primarily adds the ability to cross several networks using 'routing'

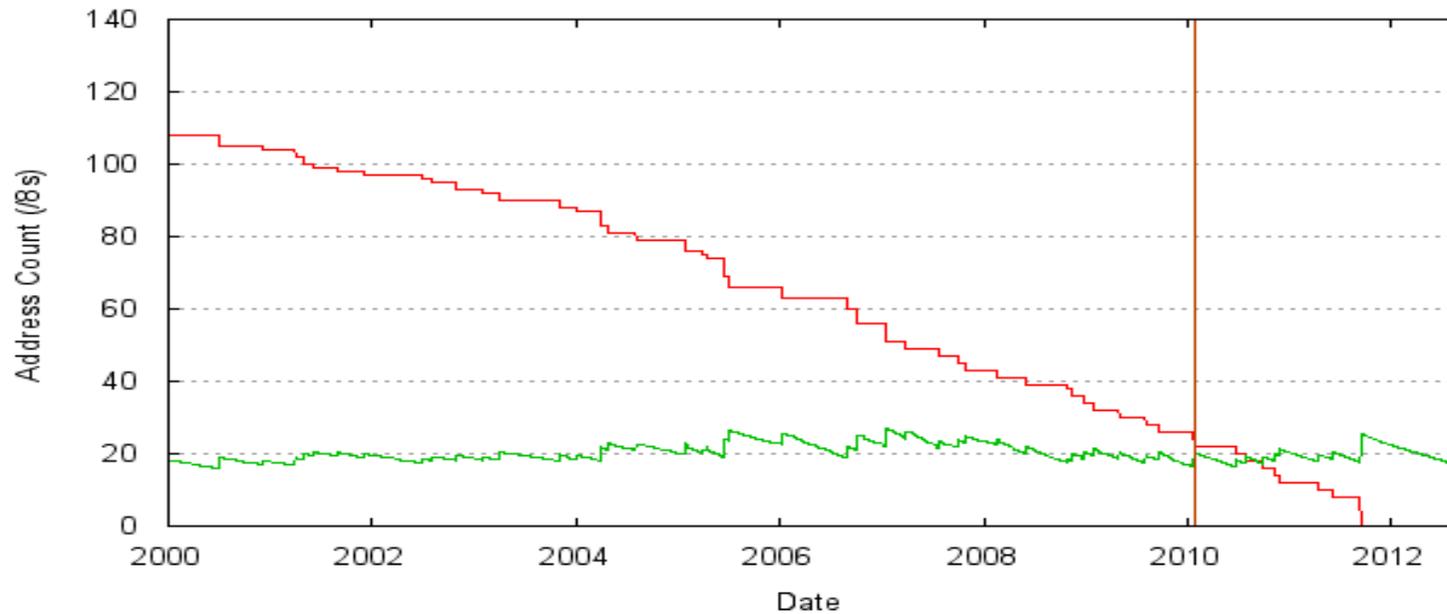
IPv4 addresses

- Each *interface* in an IPv4 Internet is assigned a unique 32-bit internet address
 - Not node addresses!
- Address types
 - Unicast - one-to-one
 - Anycast - one-to-any
 - Multicast - one-to-many
 - Broadcast - one-to-all
- Address Space
 - $2^{32} = 4\,294\,967\,296$
- Notation
 - Dotted-decimal: 192.36.125.18
- An address has two purposes
 - Identifier: Uniquely identify a host
 - Locator: Give location of the host
- It therefore has two parts
 - Netid (prefix) identifies a network
 - Hostid identifies a node on that network
- Slash notation: <netid>/<netidlen>
 - Example: 192.36.120.0/21

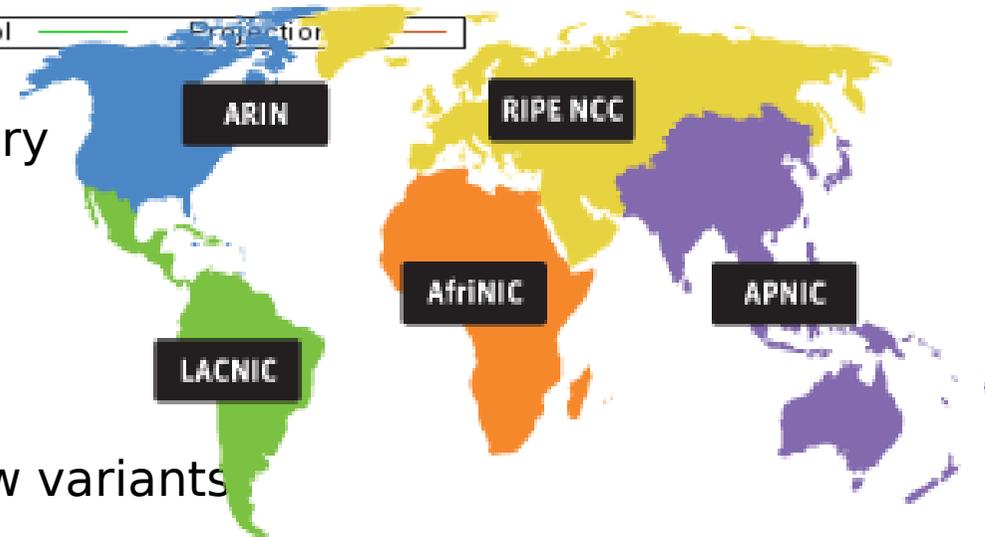


192.36.125.18

IPv4 address exhaustion



IANA Pool — RIR Pool — Projection



- IANA ran out of IPv4 address blocks in February
- APNIC (asia) is already empty
- RIPE (europe) end of 2011
- ARIN (america) in 2012
- What to do then?
- Address translation (NAT) in different and new variants
- IPv6!

ARP - Address Resolution Protocol

- Problem: A source wants to send a packet to an interface on a *directly attached* broadcast network - we know the IP-address of the destination but not the MAC address.
- Idea: Broadcast a request - “On which MAC address can IP-address X be reached?”.

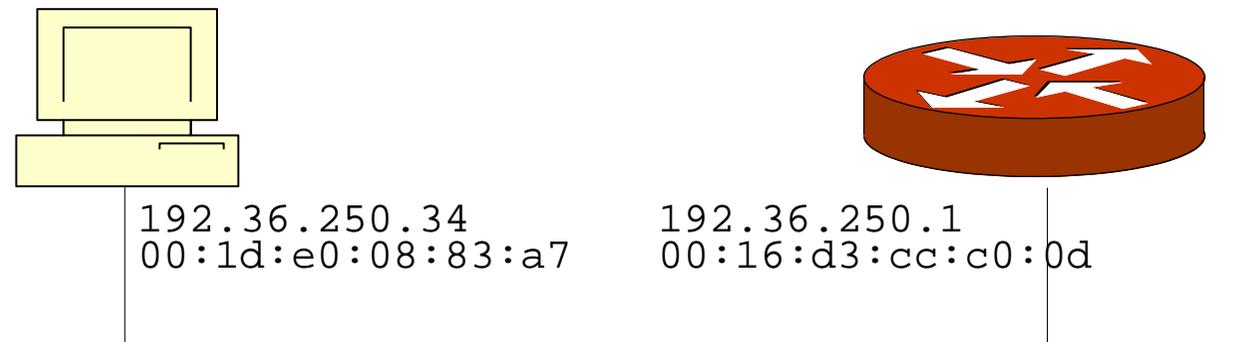
-ARP request

- The host/router with the destination replies with its MAC address

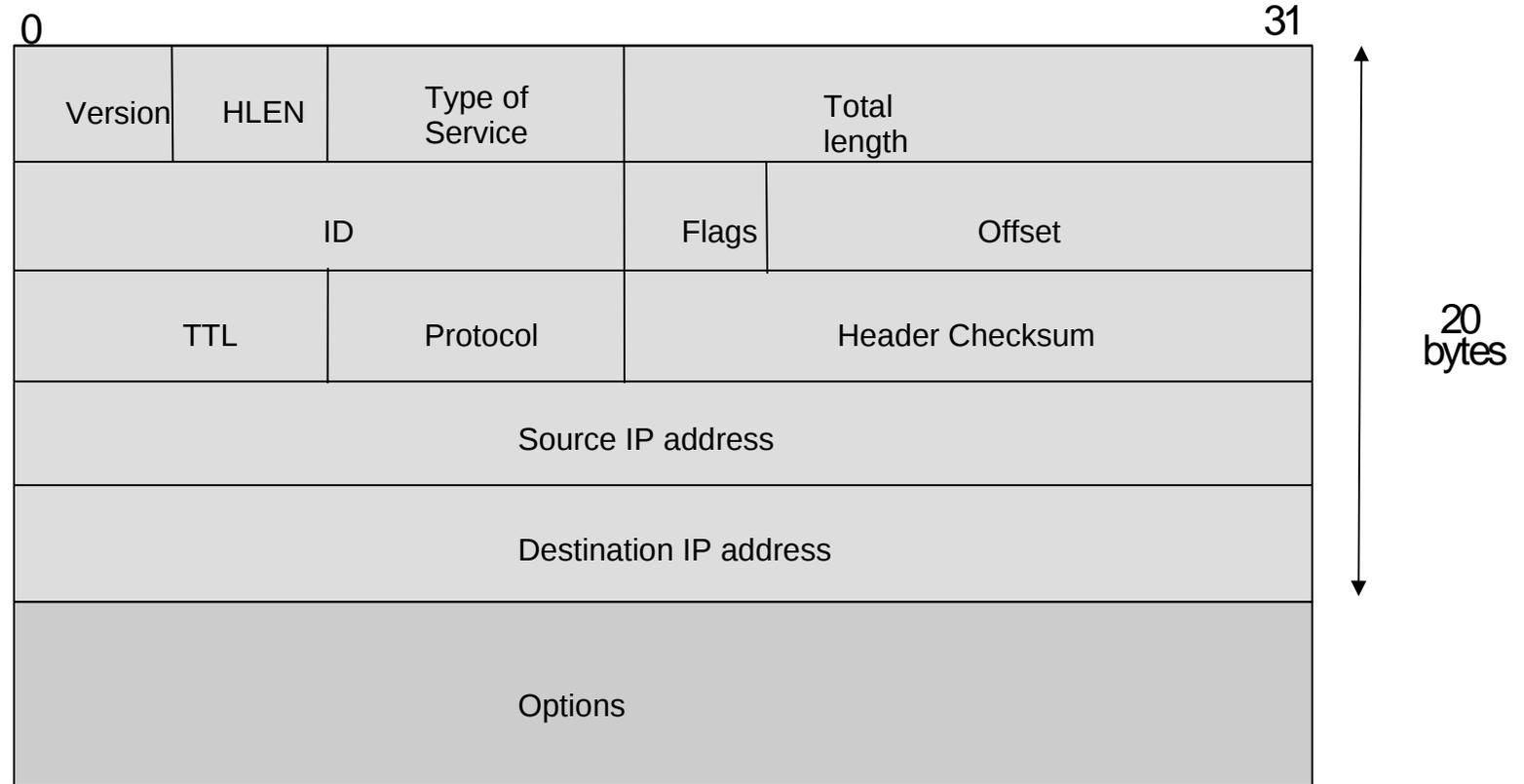
-ARP reply

- The source saves the reply in a cache

-So you dont need to ask next time



IPv4 Header



- Version
- HLEN - Header Length
- Type of Service
- Total Length
 - Header + Payload
- Fragmentation
 - ID, Flags, Offset
- TTL - Time To Live
 - Limits lifetime
- Protocol
 - Higher level protocol
- Header checksum
- IP Addresses
 - Source, Destination
- Options
 - up to 40 bytes

ICMP

ICMP is a limited signalling protocol for IPv4.

- Report IP problems back to sender
- Control and Management
- Considered a part of IP, but uses IP for transfers.



Type	Message
3	Destination unreachable
4	Source quench
11	Time exceeded
12	Parameter problem
5	Redirection

Type	Message
8/0	Echo request/reply
13/14	Timestamp request/reply
17/18	Address mask request/reply
10/9	Router solicitation/advertisement

Transport layer

Provides service to end-applications: ports

TCP

- Connection-oriented
- Reliable
- Full-duplex
- Data as byte-stream

- Mostly used
- Network-friendly

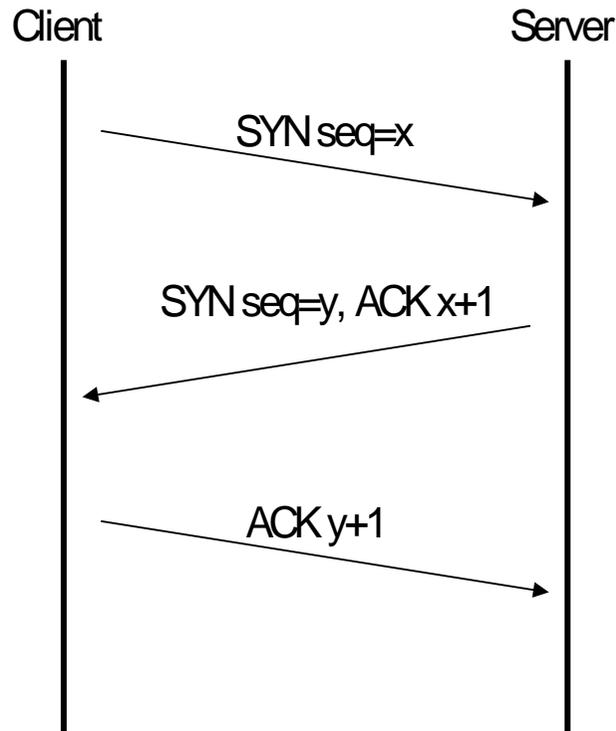
UDP

- Packet-oriented
- Unreliable
- Full-duplex
- Data in packets

- Real-time traffic
- Reliability in application



TCP Connection Establishment



Normally, client initiates the connection

3-way handshake:

- Guarantees both sides ready to transfer data
- Allows both sides to agree on initial sequence numbers

Initial sequence number (ISN) must be chosen so that each incarnation of a specific TCP connection between two end-points has a different ISN.

Note: two well-known TCP attacks:

- SYN flooding
- Sequence number attacks



TCP end-to-end reliability

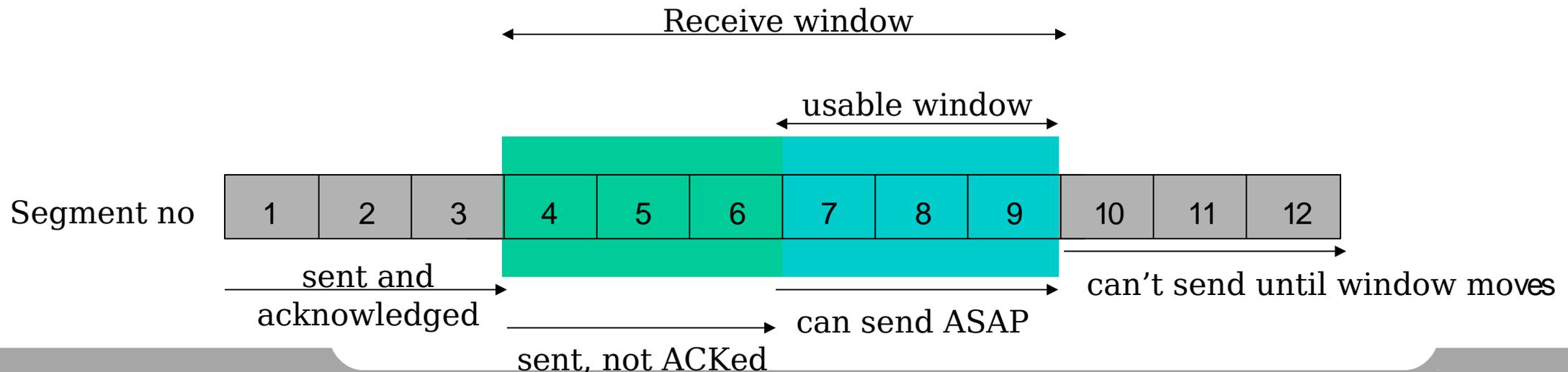


- *Flow* control : sender shall not overrun receiver
- Sliding window
- Receiver announces how much data it can receive using ACKs
- Avoid
 - Stop and go
 - silly windows
- *Congestion* control: Back-off when packets are lost in network
- Slow-start to quickly reach network capacity
- Congestion avoidance to slowly reach limit
- Fast retransmit
 - Dont drop to zero at loss

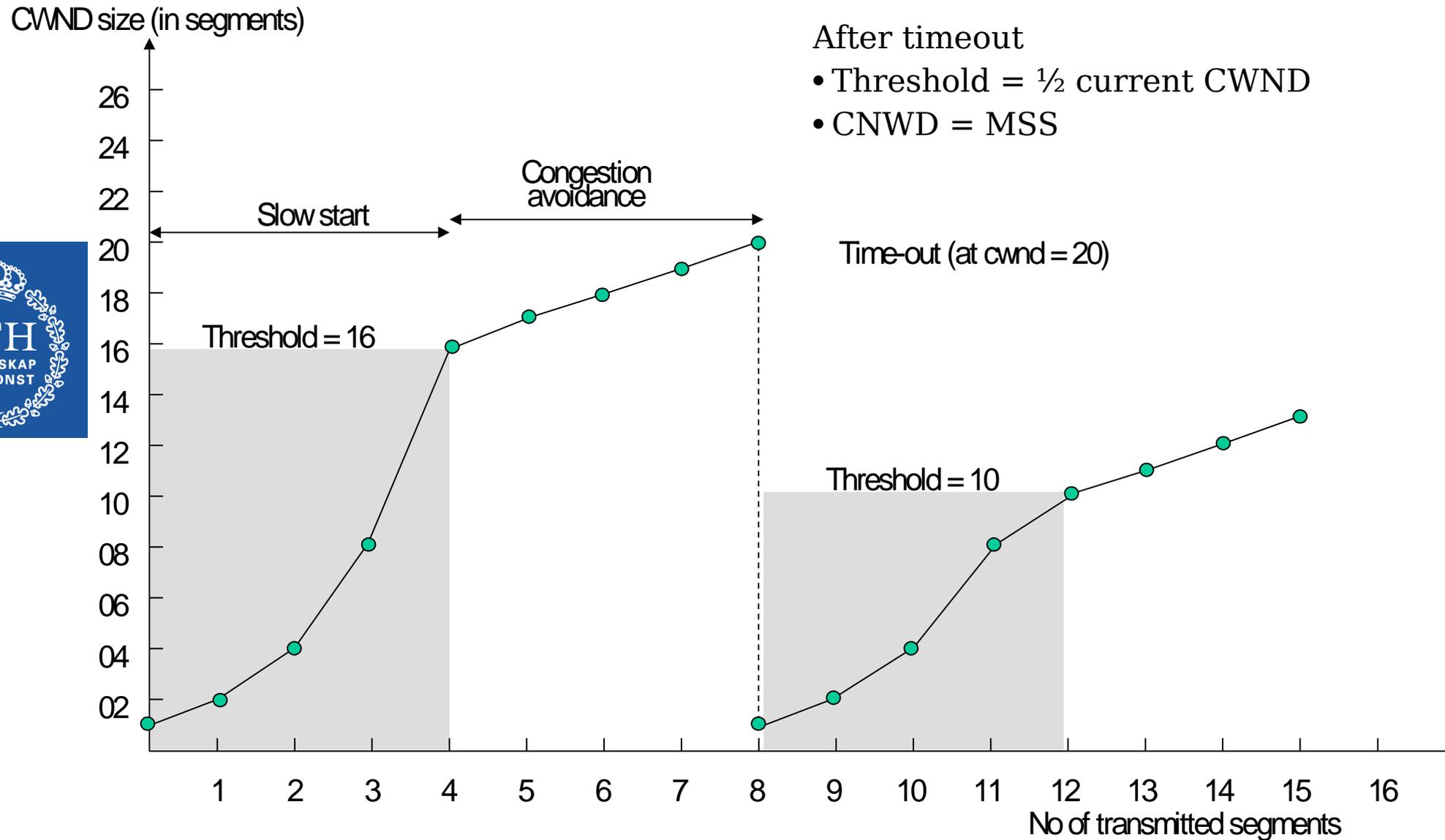
Actual window size: $\min(\text{receive window}, \text{congestion window})$

Sliding window

- Receiver: receiver window – acknowledges data sent and what it is prepared to receive
- Sender window size opened and/or closed
 - Receiving an ACK shifts the window by a constant value.
- Stop and go:
 - Send data, wait for ACKs
- Silly window:
 - Send one byte



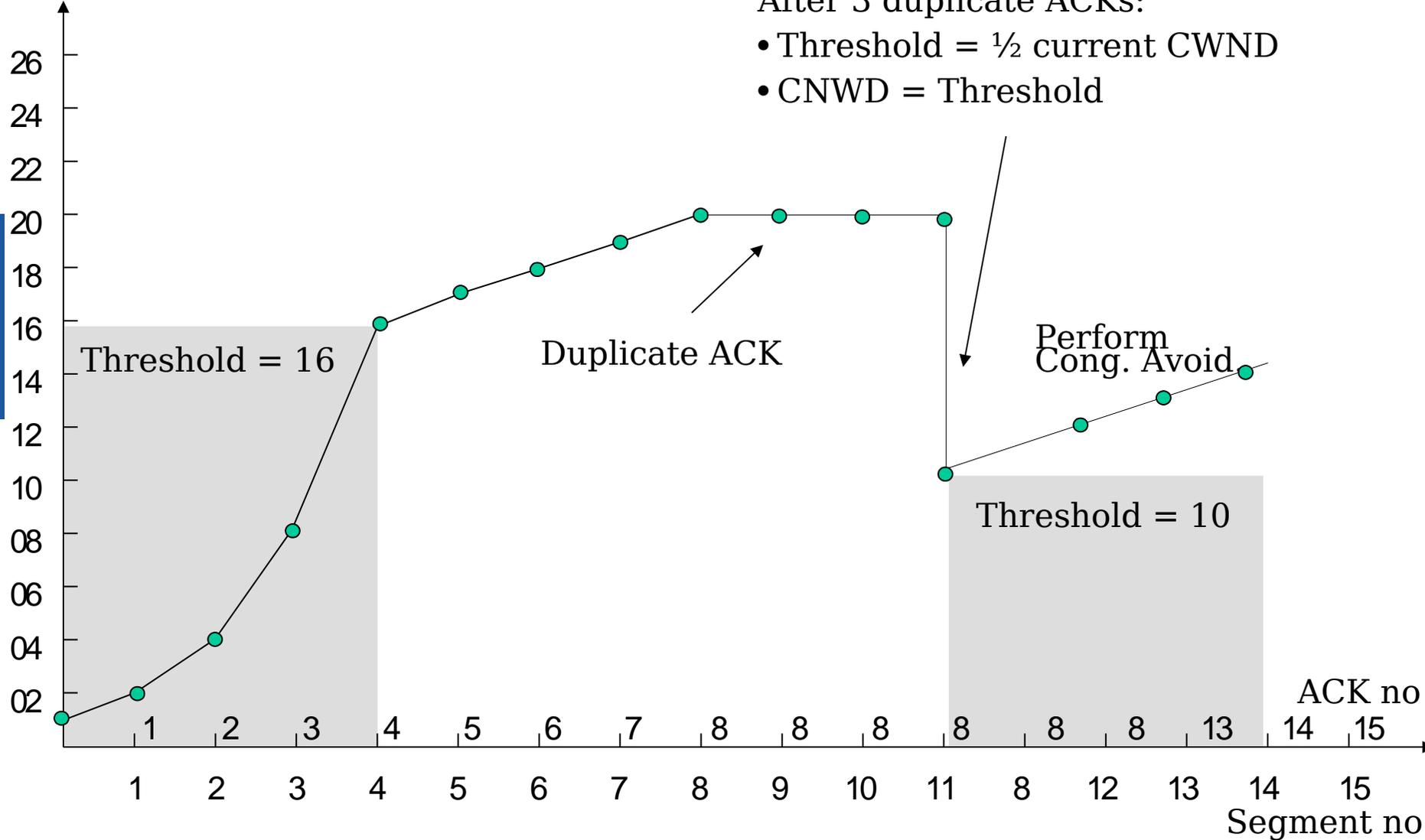
Congestion avoidance



Fast Retransmit and Recovery



CWND size (in segments)



DNS: Why do we need names?

- In the underlying network and transport layers it is all about addresses.

Interfaces, TCP, routing, etc.

- In IP, names are translated directly to addresses

And then we deal with addresses only

No names in the network

- Why don't we just stick with addresses?

- Names are better for humans

fe80::216:d3ff:fecc:c00d

- Names add another layer of indirection

One name can map to several logical addresses

One logical address can map to several names

- Names can be used for other things than just addressing
load balancing, mail direction, descriptions, finding services,

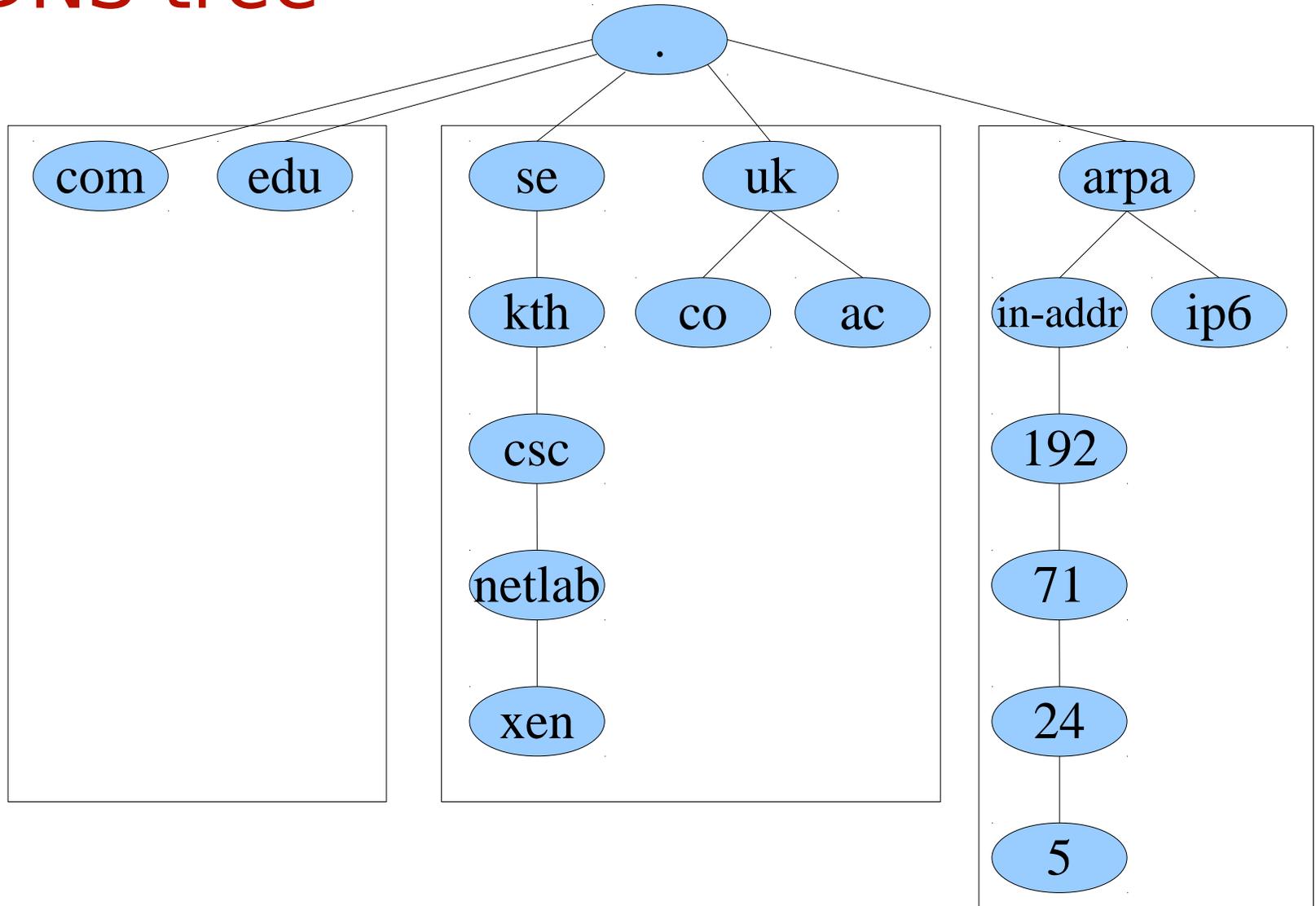


DNS architecture

- Names are structured hierarchically - in a tree form
- The DNS architecture is client-server
 - Client is called resolver
 - Server is called name server
- The resolver queries the nameservers hierarchically
 - Ultimately, you ask one of 13 root name-servers
 - Replies are cached at several places in the system

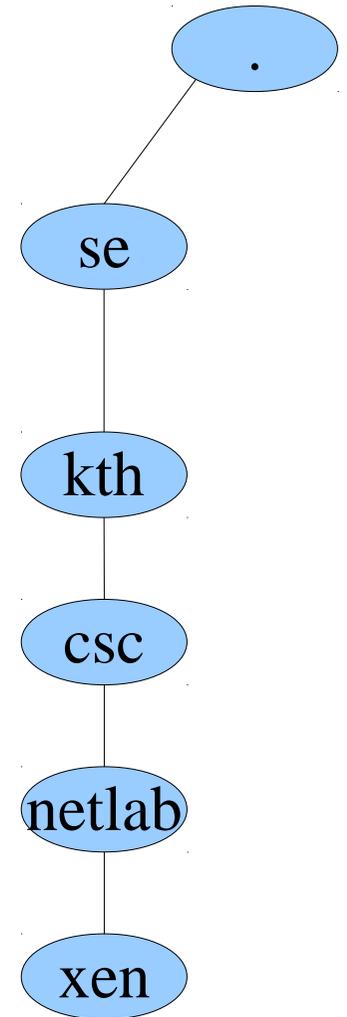
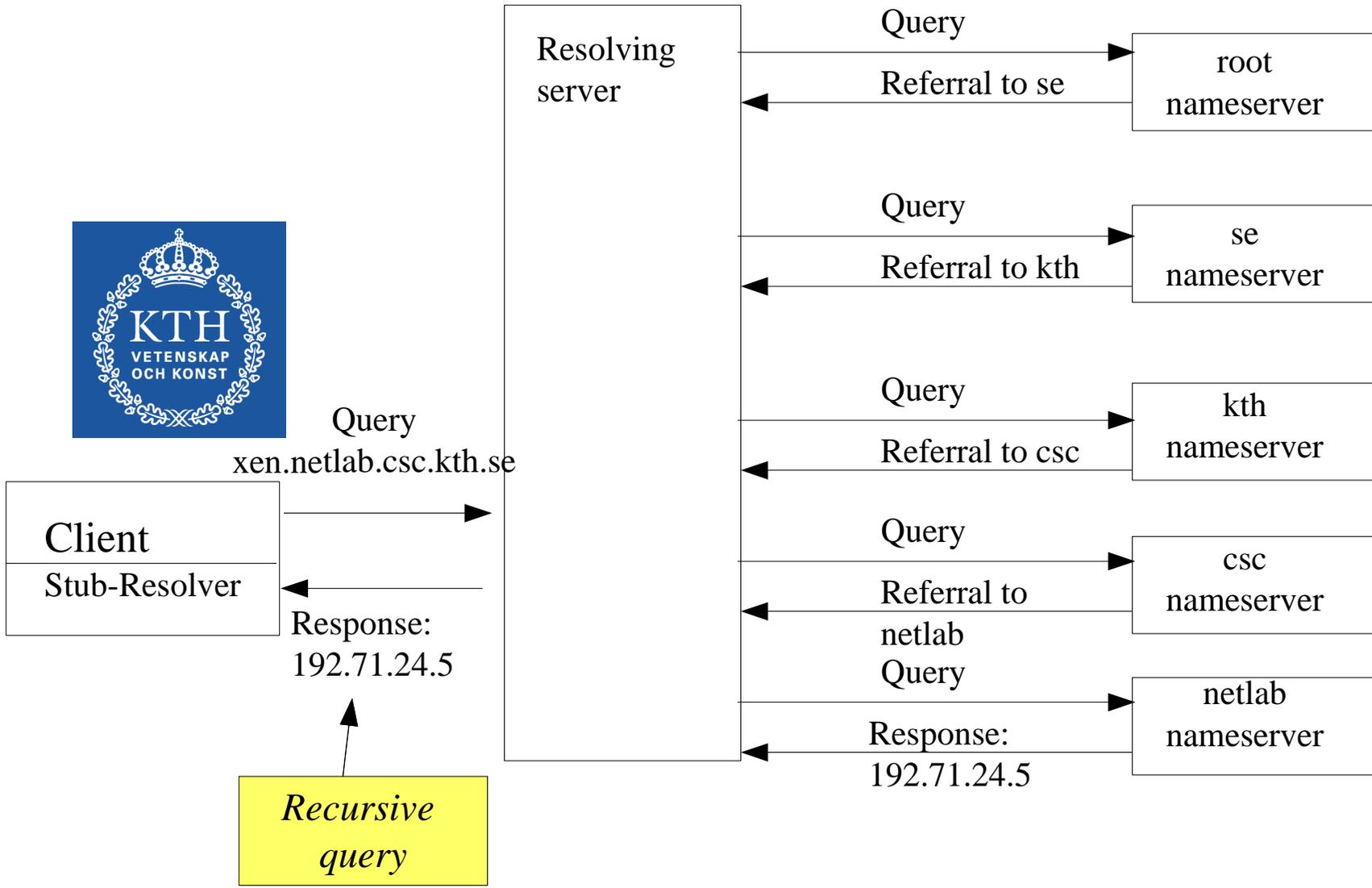


DNS tree

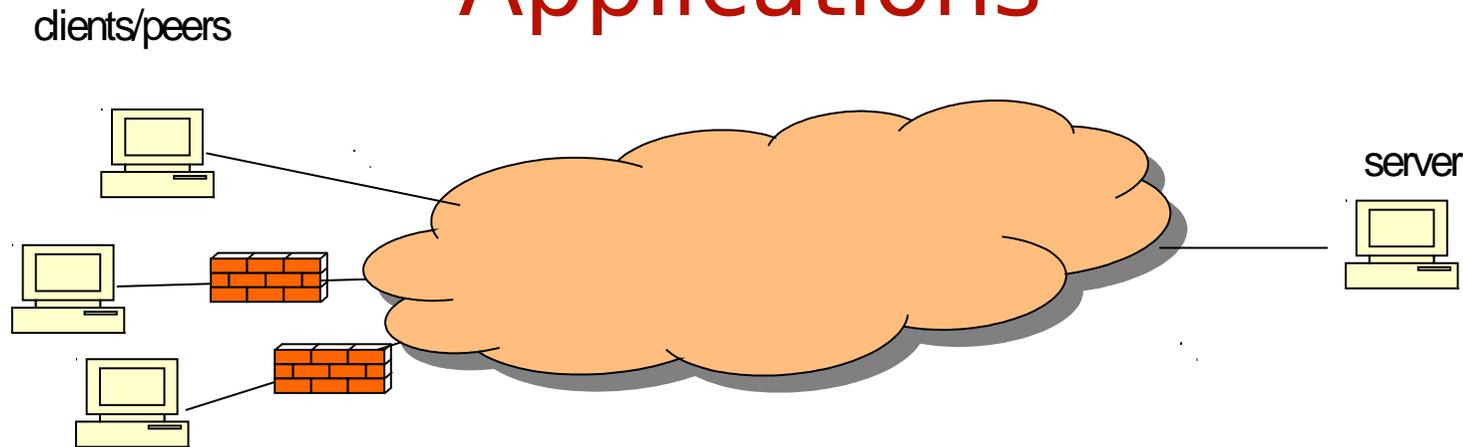


Resolving process

Iterative query

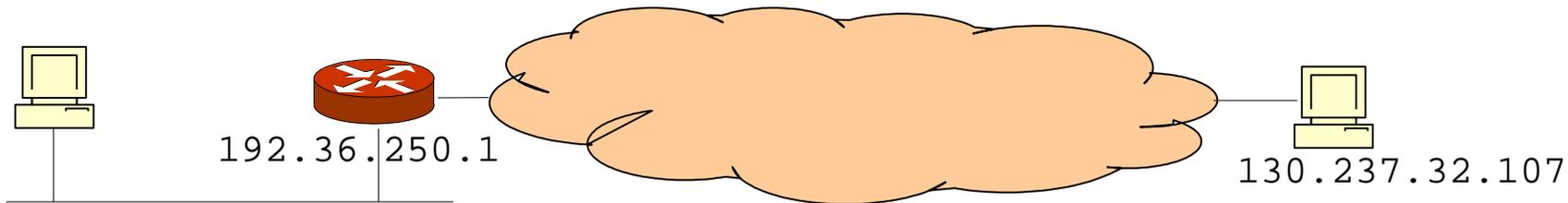


Applications



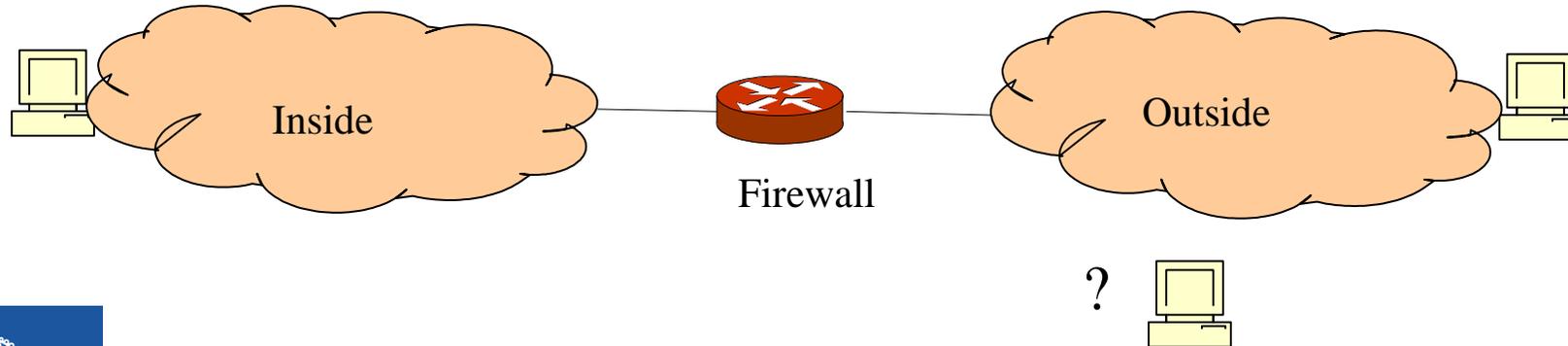
- Most applications use TCP sockets to connect between clients and servers or directly between peers - p2p
- Applications are typically end-to-end
- Anybody can create an application
 - Anybody can also attack
- Firewalls and NAT increase the security but also hinder the freedom of applications
 - Especially for peer-to-peer applications
- Firewalls are in general application-aware

Example: packet transfer



- An end host communicating with a server via a wireless LAN
- DNS: Resolve the name to an IP address
 - www.kth.se -> 130.237.32.107
- Make a routing lookup to get next-hop
 - Nexthop(130.237.32.107) -> 192.36.250.1
- ARP: Translate the IP address of nexthop to MAC address
 - 192.36.250.1-> 00:e0:35:64:e9:e7
- Send TCP/IP SYN packet to 130.237.32.107 with port 80 to next hop using wireless LAN MAC address 00:e0:35:64:e9:e7
- TCP/IP SYN+ACK packet returns from remote peer
- Send HTTP Request on the TCP connection
- HTTP Response comes back with content

DD2395 lab – packet filter firewall



- The lab configuration consists of three virtual hosts running on `xen.netlab.csc.kth.se`
- There is an outside host, an inside host, and a packet-filter firewall that you have control over.
- There is also an unknown server.
- Your first task is to scan the server to detect its address & capabilities
 - You use 'nmap' for this
- Your second task is to program packet-filters on the firewall
 - Protect internal servers and clients from access and attacks from the outside while allowing access of the outside from the inside
- You use the Linux '*ip-tables*' to achieve this.
- Read the lab instructions thoroughly,
 - read about ip-tables and nmap!