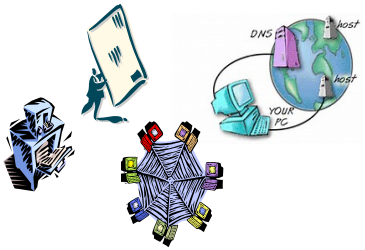
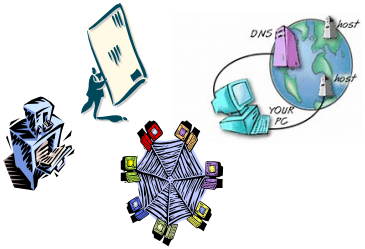
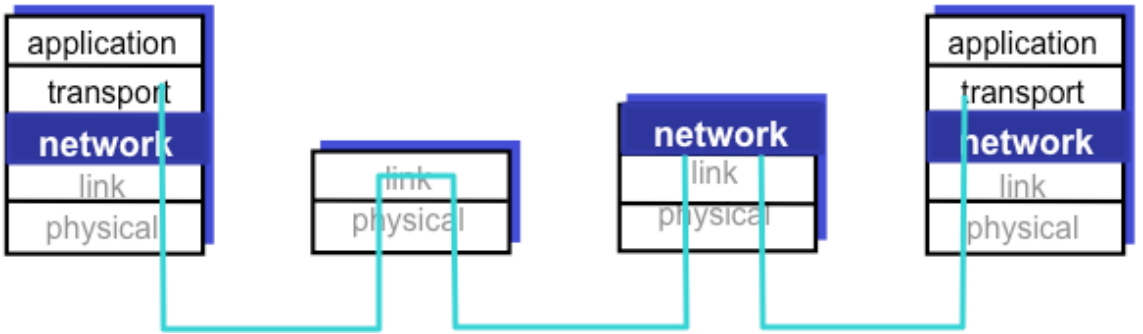
Network layer
Chapter 4

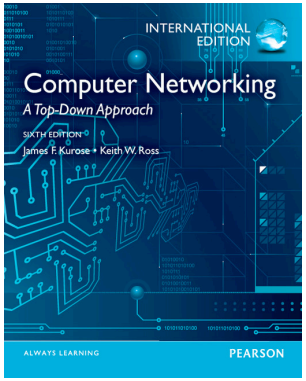
Kjersti Moldeklev, Prof II
Information Security and Communication Technology

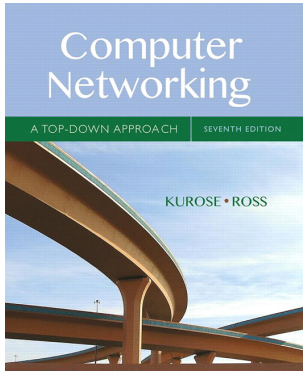
kjersti.moldeklev@ntnu.no











The *content* of some of these slides are based on slides available from the web-site of the book by J.F Kurose and K.W. Ross.

Network layer – Feb 3 + Feb 9/10

4	Friday 09:15 – 11:00	Transport Layer (cont)	R1	Kjersti	Chapter 3
5	Thursday 12:15 – 14:00	Transport Layer (cont)	R1	Kjersti	Chapter 3 Chapter 8.6
5	Friday 09:15 – 11:00	Network Layer	R1	Kjersti	Chapter 4
6	Wednesday 18:15 – 20:00	Python Crash Course	F1	Magnus/Bank?	Install Python before the crash course.
6	Thursday 12:15 – 14:00 Thursday 14:15 – 15:00	Network Layer (cont) Theory Assignment 3: *Transport Layer* Wireshark Lab 2: *TCP (optional but highly recommended!)*	R1 R1	Kjersti Assistants/ Ida/Norvald	Chapter 4 One must deliver and pass at least 5 of the 8 theory assignments.
6	Friday 09:15 – 11:00	Network Layer (cont)	R1	Kjersti	Chapter 4 Chapter 8.7 and 8.9, 8.9.1

Network layer – in each and every node and end system

4.1 Introduction

~~4.2 Virtual circuit and datagram~~

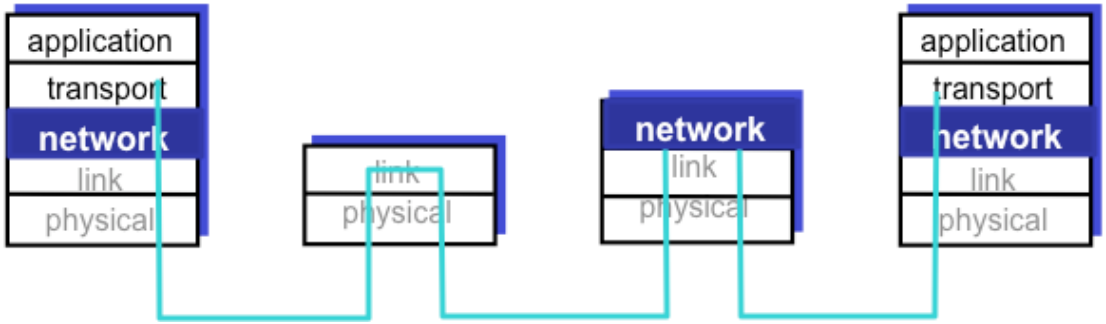
4.3 What's inside a router
- Input processing
- Switching
- Output processing
- Queuing

4.4 IP: Internet Protocol
- Datagram format
- IPv4 addressing
- ICMP
- IPv6

4.5 Routing algorithms
- ~~Link state vs Distance vector~~
- ~~Hierarchical routing~~

8.7 Network layer security

8.9.1 Firewalls



Network layer - goals

- Understand principles behind network layer services:
  - network layer **service models**
  - **forwarding** versus routing
  - how a **router** works
  - **routing** (path selection)
  - dealing with **scale**

Instantiation and implementation in the Internet:

- **IP – Internet Protocol, v4 and v6**
- **ICMP – Internet Control Message Protocol**
- **Routing hierarchy**
- Adding **network layer security**

NTNU

Network layer functions

1. **Addressing**
   - Global addresses crossing different network technologies

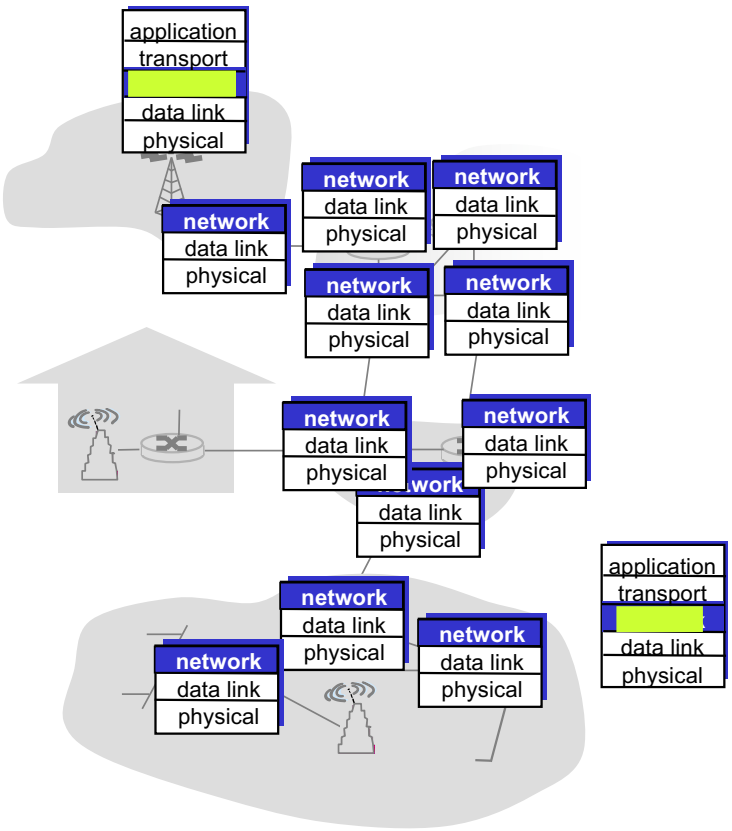
2. **Fragmentation** and **reassembly**
   - Underlying network technologies have different properties, e.g. MTU (Maximum Transfer Unit)

3. **Routing** and **forwarding**
   - Find the route from source to destination
   - Forward packets from input to output port

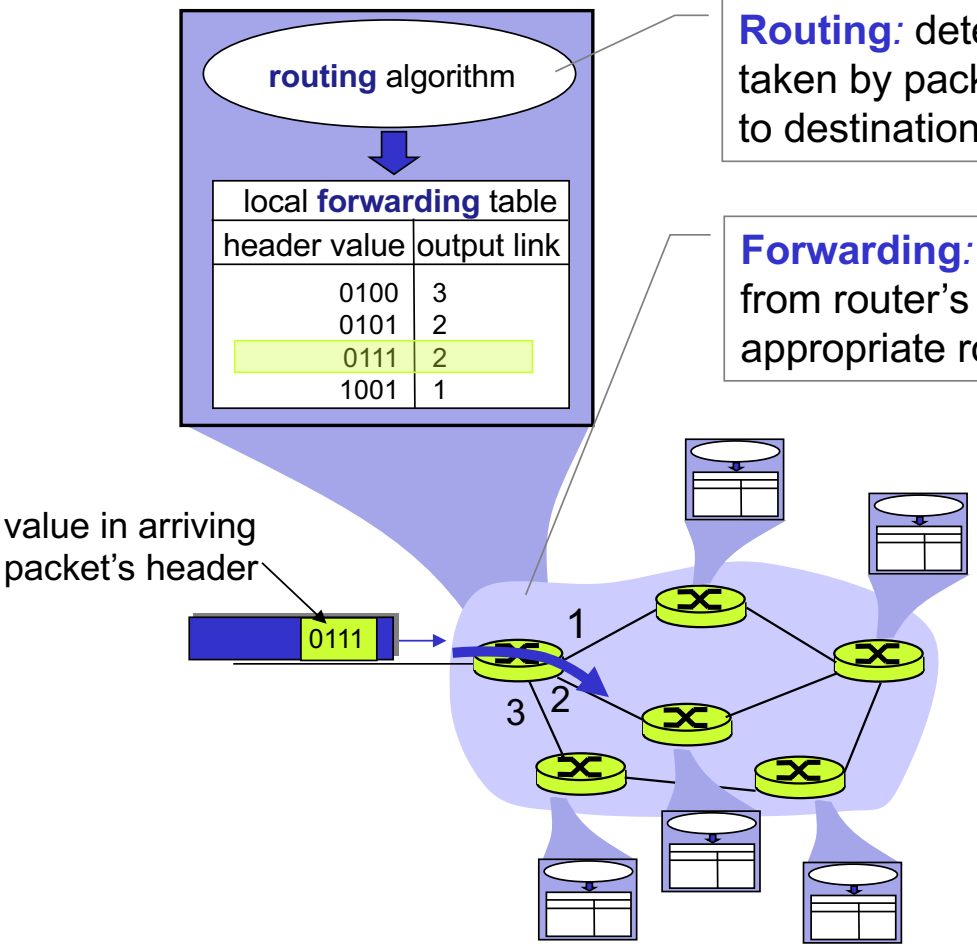
4. **Network interconnect**
   - Different underlying network technologies

Network layer transports segment from sending to receiving host

- Network layer protocols **in every node [host, router]**
- Routers **examine** header fields in all **IP datagrams** passing through
- Sending side **encapsulates** segments into datagrams
- Receiving side, **delivers segments** to transport layer



Routing and forwarding



Routing: determine route taken by packets from source to destination

Forwarding: move packets from router's input to appropriate router output

Strategies for packet switching

Packet switching

- Store-and-forward
- Handles packets
- Statistical multiplexing

Virtual circuit: connection oriented

- Connection set-up establishes state information in network nodes
- Routes **connections**
- Packets follow the same path

Datagram: connectionless

IP

- No set-up and establishment of state information
- Routes **packets**
- Packets may take different ways to the destination

Datagram packet switching

Virtual circuit

- Buffers reserved at connection time
- Can guarantee packets in sequence
- Shorter headers
- Each packet VC (virtual circuit) number
- Delayed duplicates are avoided
- Router table space per connection
- Problems if subnetwork is based on datagram
- RTT (round trip time) for connection set-up
- Router failures: VC terminated

Datagram

- Robust against network failure
- Congestion avoidance
- No requirement on underlying network
- Longer headers - higher overhead than for connection oriented networks
- Each packet full address
- Routers do not hold state information per connection
- Potential for congestion
- No establishment delay
- Router failures: limited

Network layer

4.1 Introduction

~~4.2 Virtual circuit and datagram~~

4.3 What's inside a router

- **Input processing**
- **Switching**
- **Output processing**
- **Queuing**

4.4 IP: Internet Protocol

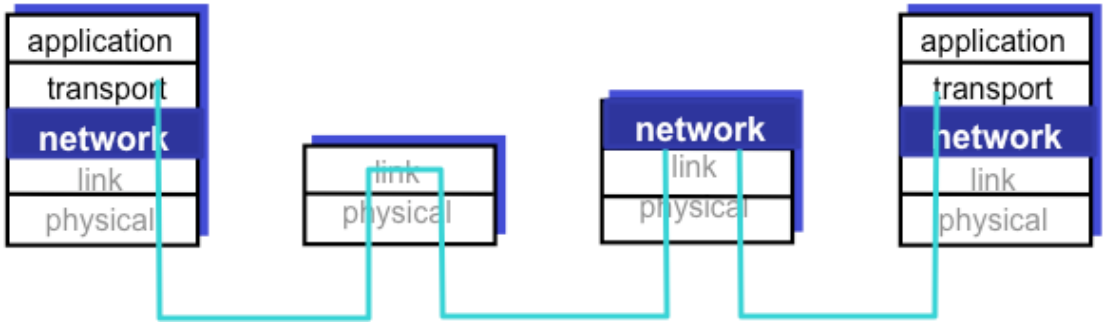
- Datagram format
- IPv4 addressing
- ICMP
- IPv6

4.5 Routing algorithms

- ~~Link state vs Distance vector~~
- ~~Hierarchical routing~~

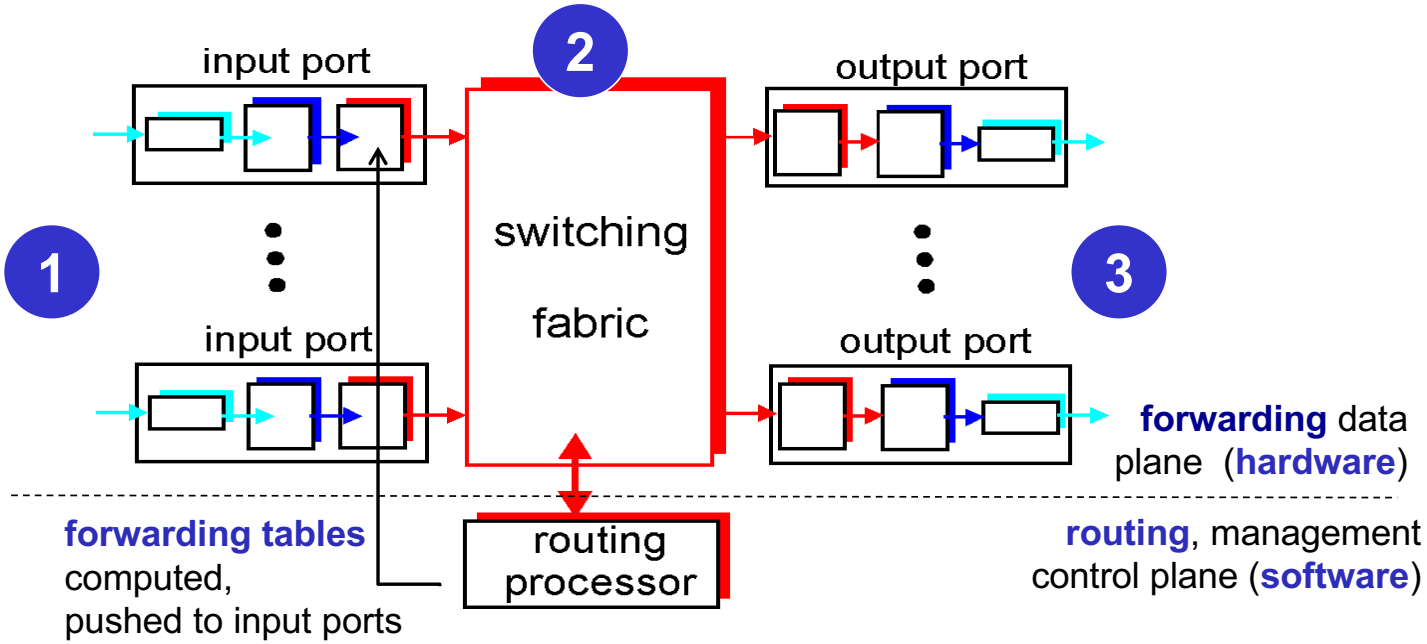
8.7 Network layer security

8.9.1 Firewalls



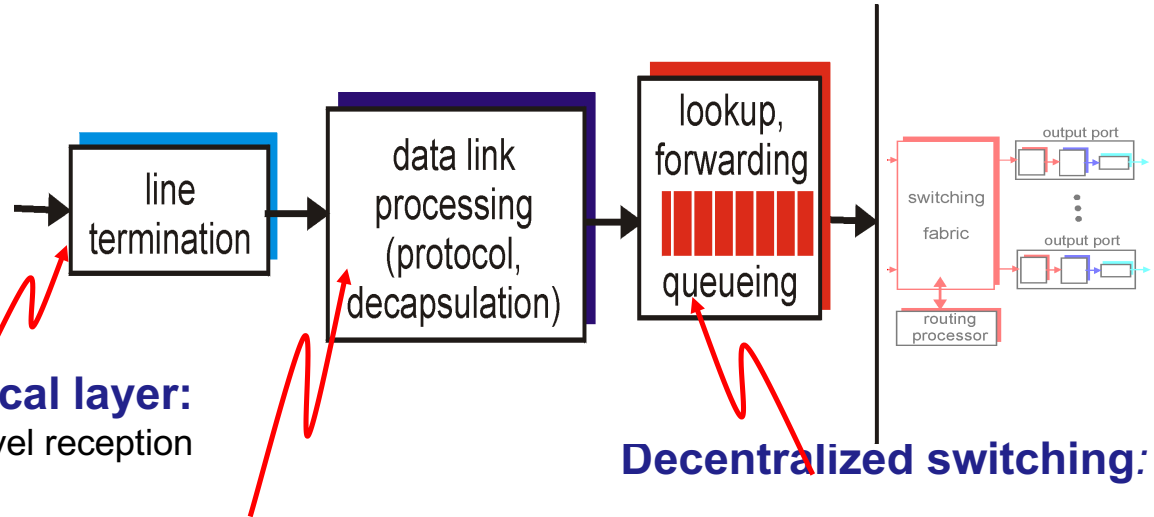
Router architecture – two key functions, three key modules

- Run **routing algorithms/protocols**
- **Forward datagrams** from incoming to outgoing port



NTNU

1 Queuing on input port if datagrams arrive faster than forwarding rate into switch fabric



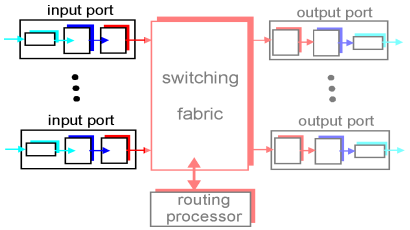
Physical layer:
bit-level reception

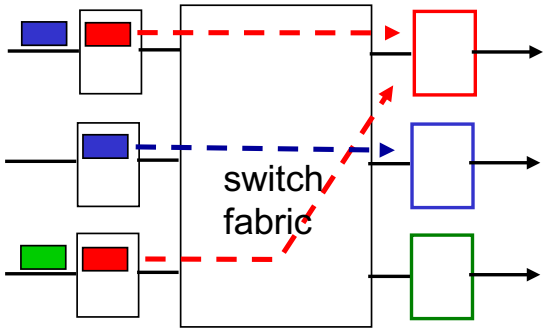
Data link layer:
e.g. Ethernet, IEEE802.3

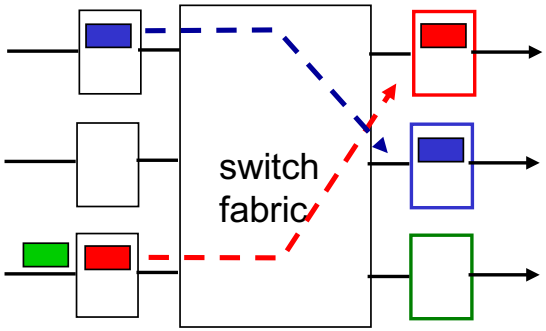
Decentralized switching:

- Given datagram destination, lookup output port using forwarding table in input port memory
- Goal: complete input port processing at 'line speed'

1 Input port queuing challenges



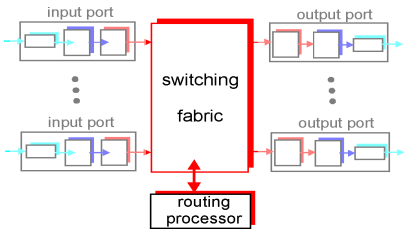




- **Output port contention:**
  switch slower than input ports combined -> queueing may occur at input queues
  - only one **red** packet at a time

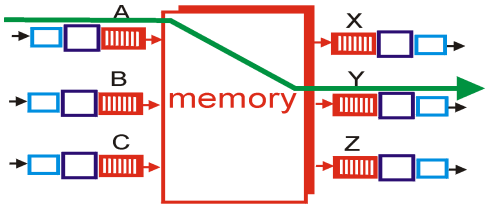
- **Head-of-the-Line (HOL) blocking**:
  queued datagram at front of queue prevents others in queue from moving forward
  - **green** packet is delayed

2 Three types of switching fabrics

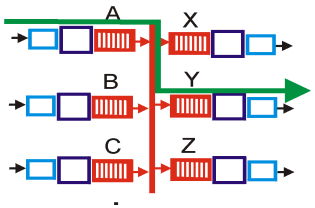


- Transfer packet from input buffer to appropriate output buffer
- **Switching rate**: rate at which packets can be transferred from inputs to outputs
  - often measured as multiple of input/output line rate
  - N inputs: switching rate N times line rate desirable

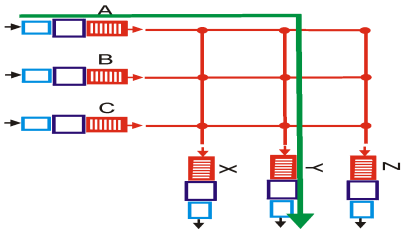
I. Memory



II. Bus



III. Crossbar

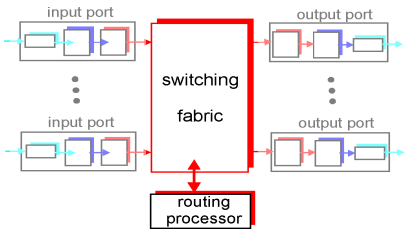


NTNU

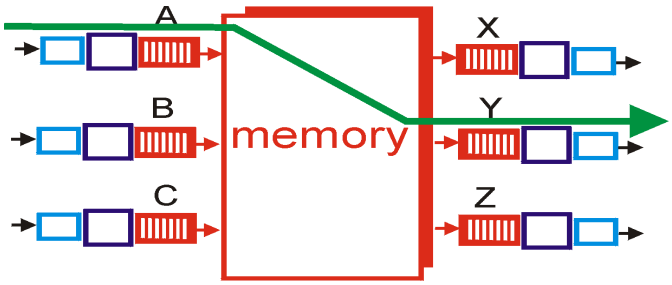
2

Three types of switching fabrics

I. Memory



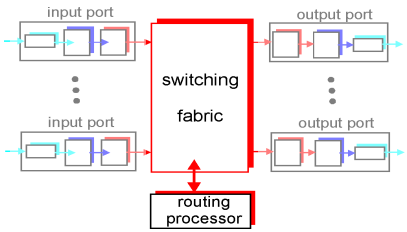
- Packet **copied to** system's memory
- One packet at a time
- Speed limited by **memory bandwidth** (2 bus crossings per datagram)
- First generation routers: traditional computers with switching under control of CPU



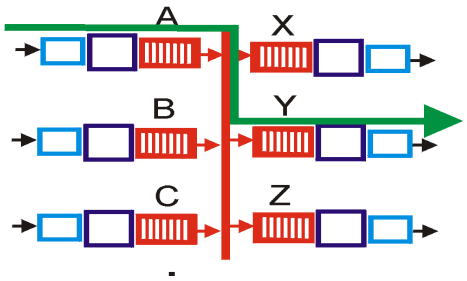
2

Three types of switching fabrics

II. Bus

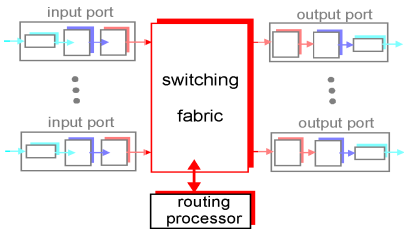


- Datagram from input port memory to output port memory **via shared bus**
- One packet at a time
- **Bus contention**: switching speed limited by bus bandwidth

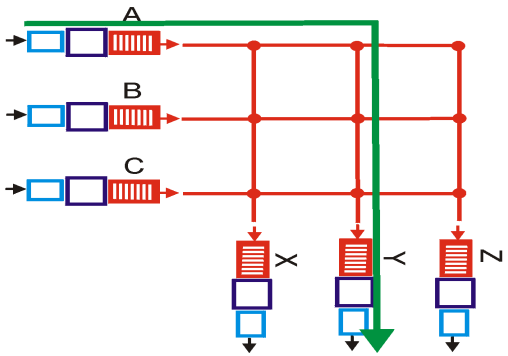


2 Three types of switching fabrics

III. Interconnection network



- Overcome bus bandwidth limitations
- Forwards multiple packets in parallel

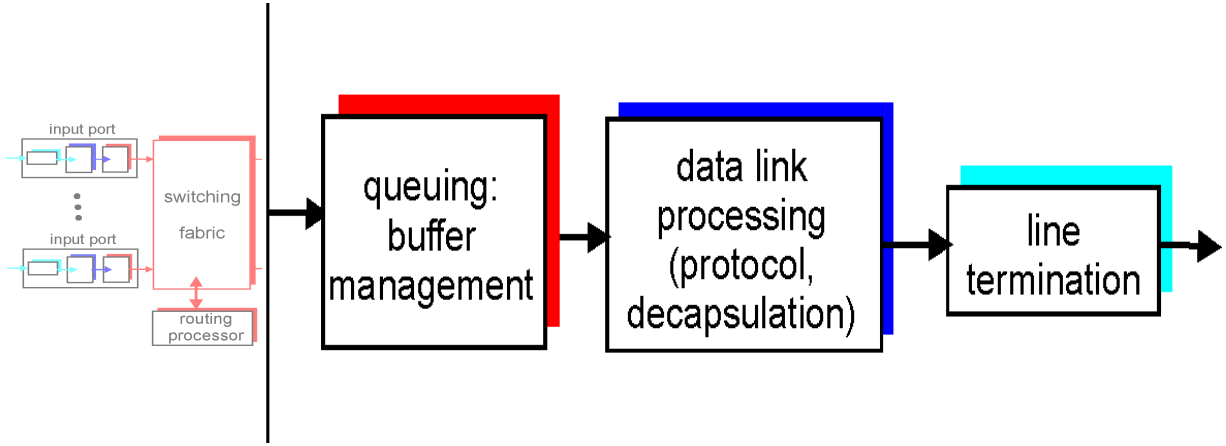


- Switching via **interconnection network**
  - Banyan networks, crossbar, other interconnection nets initially developed to connect processors in multiprocessor
  - When packet from port A needs to forwarded to port Y, controller closes cross point at intersection of two buses

- Advanced design: fragmenting datagram into fixed length cells, switch cells through the fabric

NTNU

3 Output port – queuing and loss when packets arrive from fabric faster than output line speed



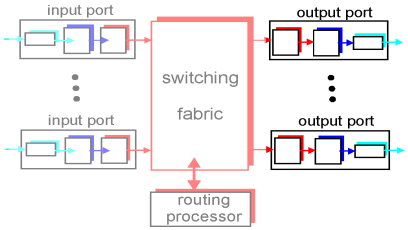
- **Buffers** required to queue packets
  - Buffering rule of thumb: average buffering equal to "typical" **RTT * C_{link}**
    e.g. R= 250 msec, C_{link} = 10 Gps -> 2.5 Gbit buffer

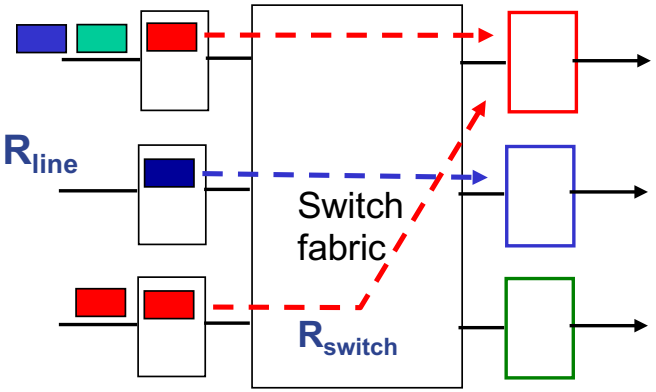
- Recent recommendation: with **N** flows, buffering equal to $\frac{RTT * C_{link}}{\sqrt{N}}$
- **Scheduling discipline** chooses among queued datagrams for transmission

NTNU

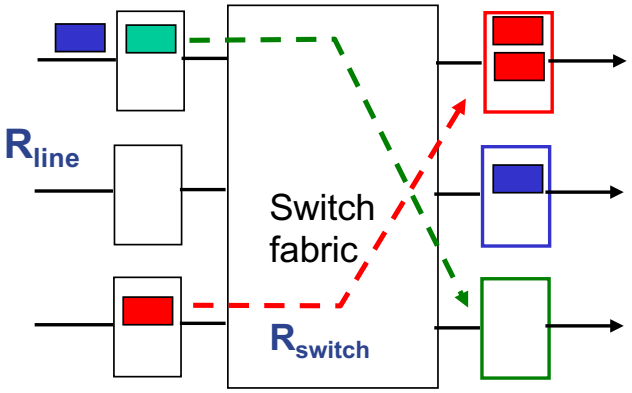
3

Output port queuing





at *t*, packets move from input to output



one packet time later

- Suppose R_{switch} is N times faster than R_{line}
- Still have output buffering when multiple inputs send to same output
- Queuing (delay) and loss due to output port buffer overflow!

Network layer – in each and every node and end system

4.1 Introduction

~~4.2 Virtual circuit and datagram~~

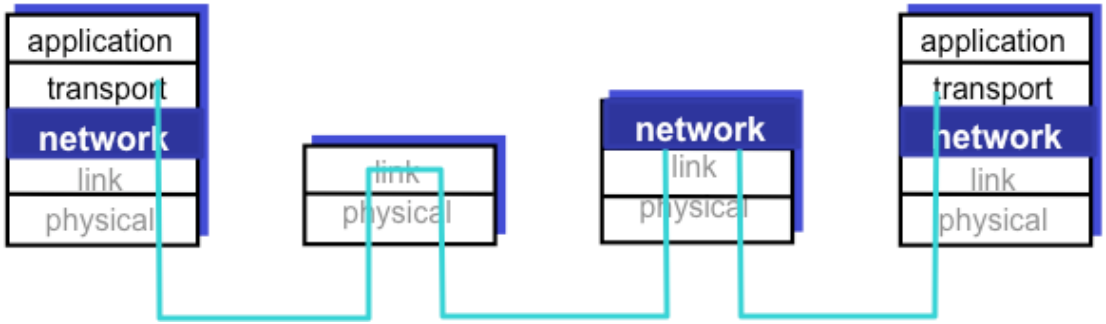
4.3 What's inside a router
- Input processing
- Switching
- Output processing
- Queuing

4.4 IP: Internet Protocol
- **Datagram format**
- **IPv4 addressing**
- **ICMP**
- **IPv6**

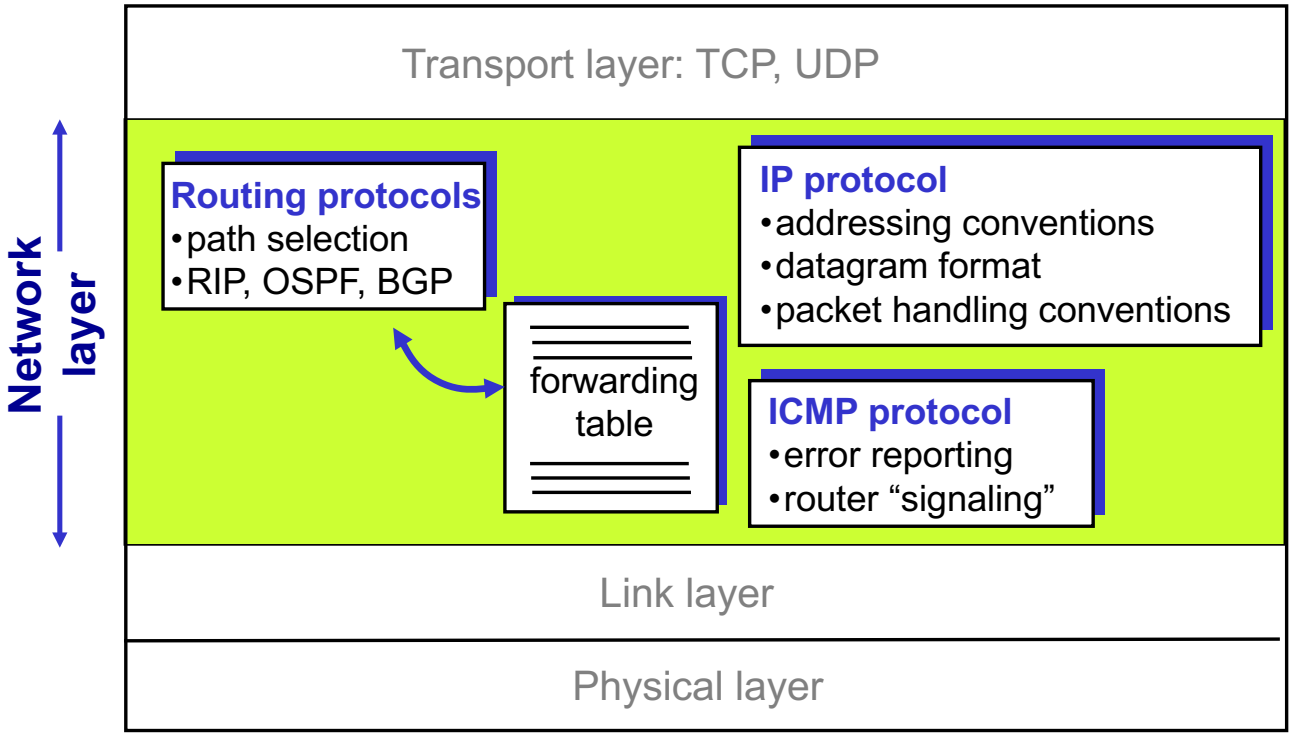
4.5 Routing algorithms
- ~~Link state vs Distance vector~~
- ~~Hierarchical routing~~

8.7 Network layer security

8.9.1 Firewalls



The network layer in routers and hosts

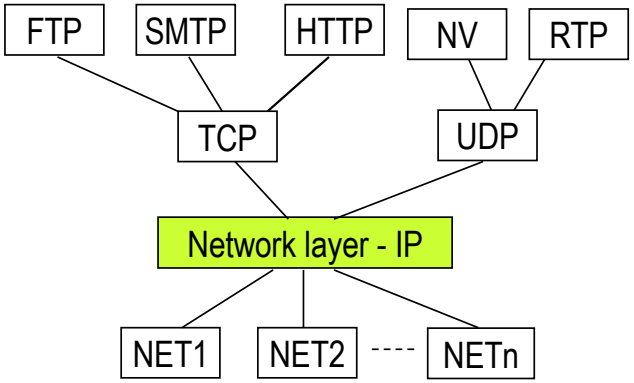


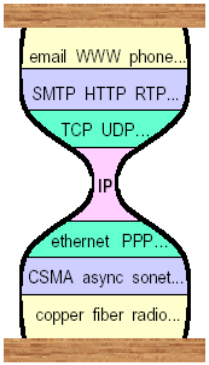
RIP Routing Information protocol
OSPF Open Shortest Path First
BGP Border Gateway Protocol

IP Internet Protocol
ICMP Internet Control Message Protocol

The Internet Protocol (IP) plays a key role in the Internet

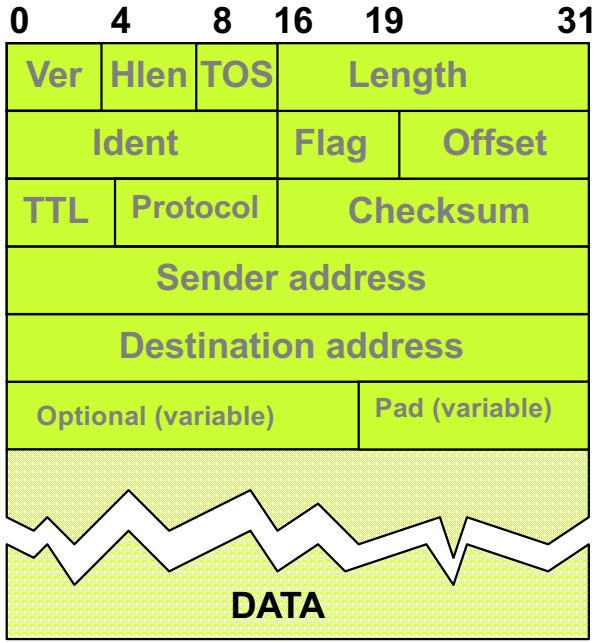
- Packet delivery is based on **datagram** and a "**best effort**" service
- IP runs both in the **end systems** and in the **intermediate network nodes**
- IP makes it possible to consider a collection of networks as a **homogenous internetwork**





- **Everything over IP**: Higher layer transport protocols such as TCP and UDP are applied above IP in the end systems
- **IP everywhere**: No requirements on underlying networks – the protocol can run everywhere

IP datagram format - most fields are unchanged from source to destination



IP version 4

Ver	4	IP version (today version 4)
HLen	4	Number of 32 bit words in header
TOS	8	Service quality
Length	16	Number of bytes in the datagram
Ident	16	Used for fragmentation/reassembly
Flag/Off	16	DF, MF, Offset: in *8 bytes
TTL	8	Time to live max number remaining hops
Protocol	8	Upper layer protocol (TCP=6, UDP=17)
Checksum	16	Checksum for header
Address	32	Source and destination address

Max IP datagram length is 65535 bytes

Network layer functions

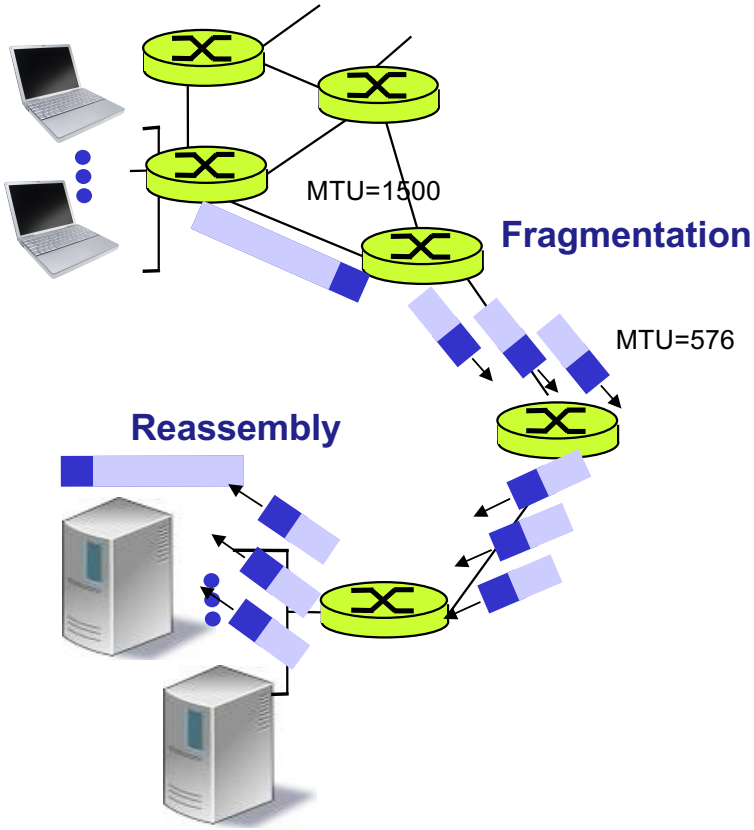
1. **Addressing**
   - Global addresses crossing different network technologies
2. **Fragmentation** and **reassembly**
   - Underlying network technologies have different properties, e.g. MTU (Maximum Transmission Unit)
3. **Routing** and **forwarding**
   - Find the route from source to destination
   - Forward packets from input to output port
4. **Network interconnect**
   - Different underlying network technologies

IP fragmentation and reassembly

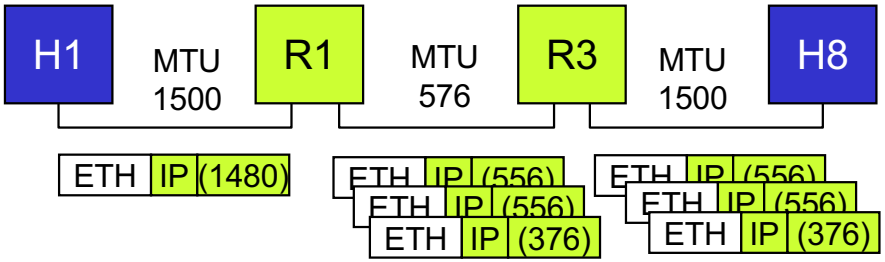
Network links have an **MTU (Maximum Transmission Unit)**

- largest possible link-level frame
- ethernet 1500 bytes

- Large IP datagram divided (“fragmented”) within net
  - one datagram becomes several datagrams
  - “reassembled” only at final destination!!
  - IP header fields used to identify, and order related fragments



Fragmentation and reassembling (cont.)



Start of header, Len = 4000				
Ident=x			0	Offset=0
Rest of header				
1480 byte data				

Unfragmented large packet becomes several smaller packets

Packet fragments

Start of header, Len = 1500				
Ident=x			1	Offset=0
Rest of header				
552 byte data				

Start of header, Len = 1500				
Ident=x			1	Offset=69
Rest of header				
552 byte data				

offset
= 552/8

Start of header				
Ident=x			0	Offset=138
Rest of header				
376 byte data				

offset
= (2*552)/8

Network layer functions

1. **Addressing**
   - Global addresses crossing different network technologies

2. **Fragmentation** and **reassembly**
   - Underlying network technologies have different properties, e.g. MTU (Maximum Transfer Unit)

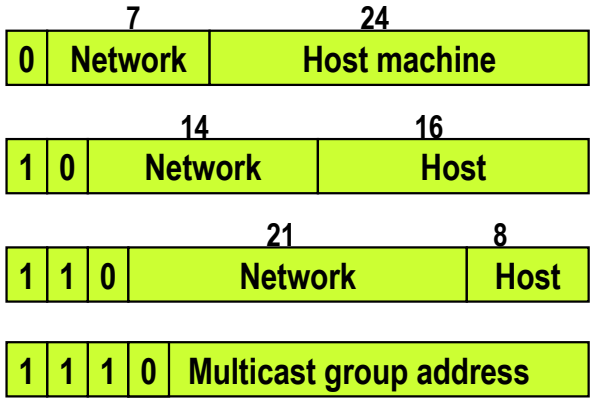
3. **Routing** and **forwarding**
   - Find the route from source to destination
   - Forward packets from input to output port

4. **Network interconnect**
   - Different underlying network technologies

IP addressing: Classful in the beginning

- Globally unique, hierarchical (network + host/router interface)
  - routers typically have multiple interfaces
  - hosts typically have one (active) interface
  - IP address **associated** with **each interface**

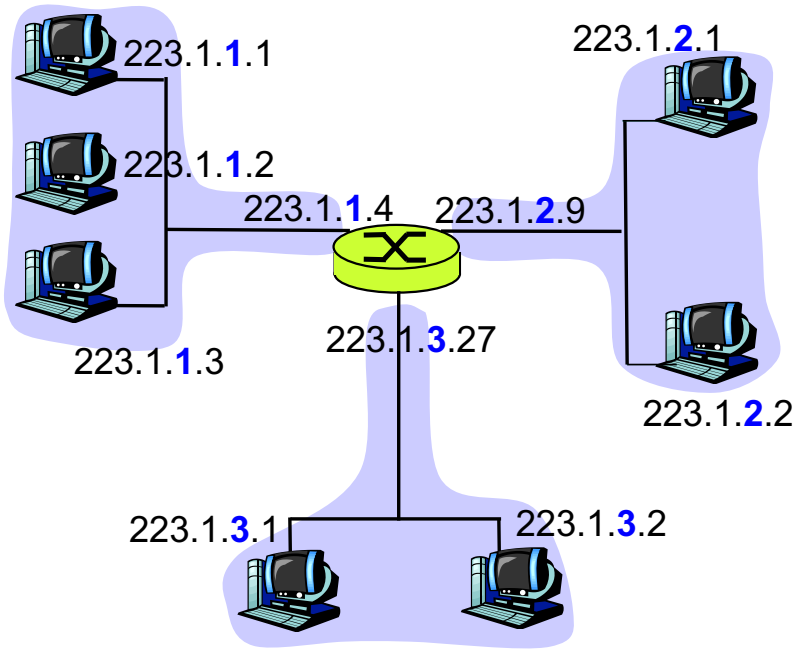
- **32-bit format**
  - Class A: 1 - 126
  - Class B: 128-191
  - Class C: 192-223
  - Class D: 224-239



- Point notation
  - Only "1"-s in the host part indicates broadcast address
  - Only "0"-s in the host part indicates a network

IP address: 32-bit identifier for host and router interfaces

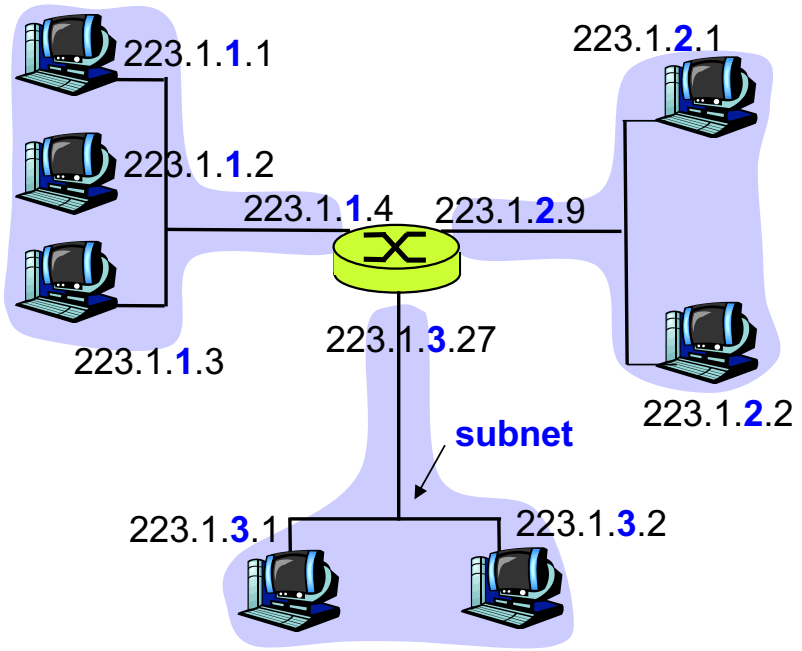
- **IP-address**: hierarchical (network + host/router interface)
- **Interface**: connection between host/router and physical link
  - routers typically have multiple interfaces
  - hosts typically have one (active) interface



One IP address associated with each interface

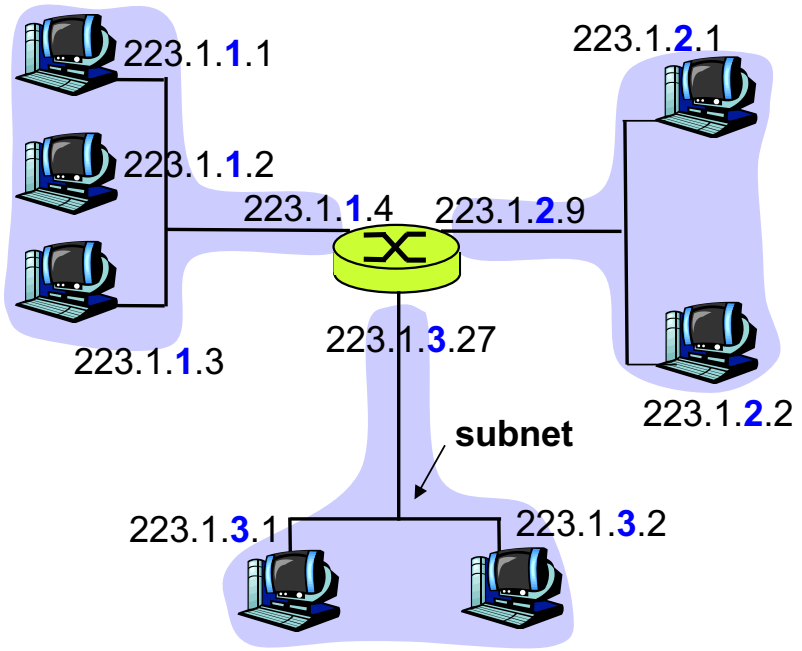
Subnetting to utilize IP address space

- **IP address**
  - subnet part - high order bits
  - host part - low order bits
- **Subnet**
  - can physically reach each other **without intervening router**
  - device interfaces with **same subnet part of IP address**



Subnet is an isolated network

- **Subnet masks** (/n) define the use of the address space
- Indicates the network part of the 32-bit IP address



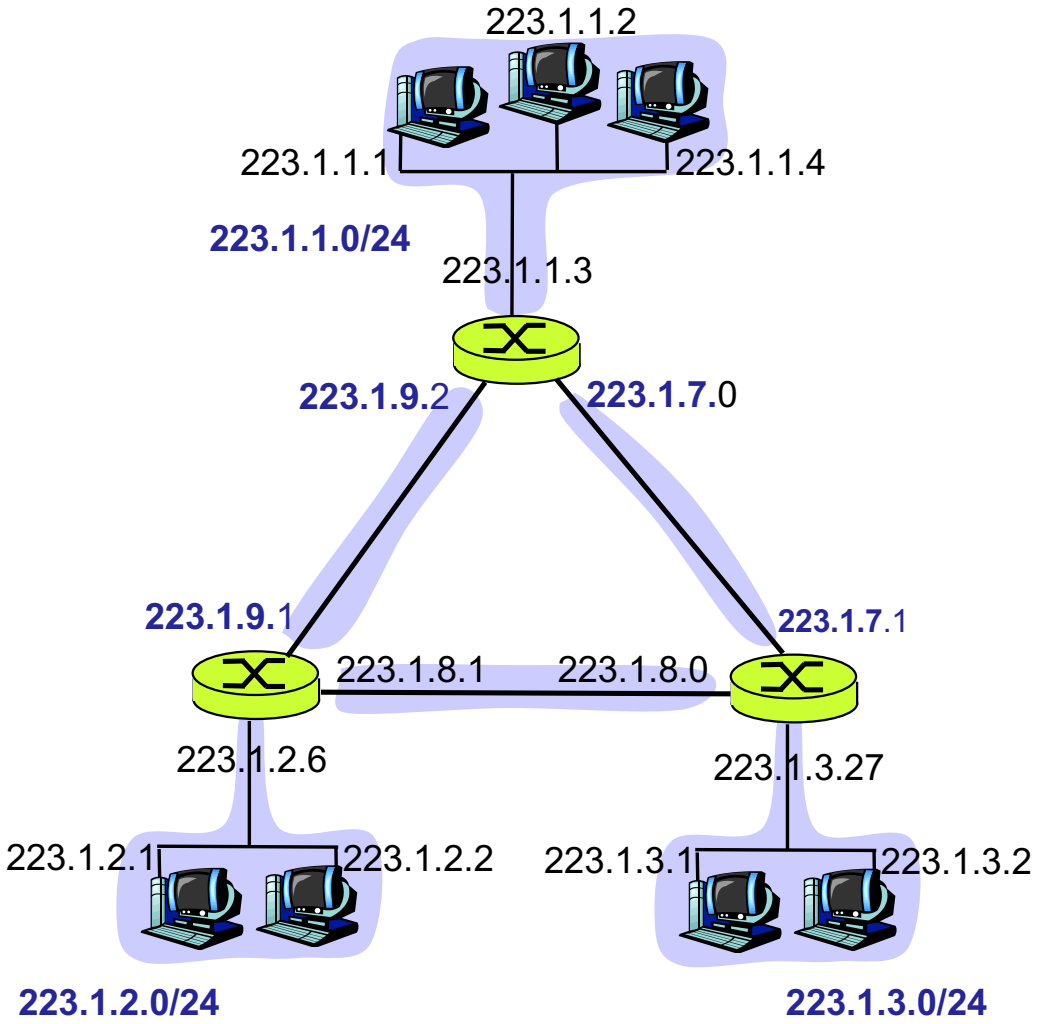
What is the subnet mask in these subnetworks?

223.1.1.1 = 11011111 00000001 00000001 00000001

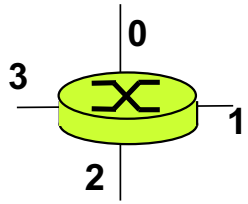
223 1 10 11 1

Subnets

- How many?
- To determine the subnets, detach each interface from its host or router, creating islands of isolated networks. Each isolated network is called a subnet.



Forwarding using longest prefix matching



- When looking for a forwarding table entry for a given destination address, use **longest** address prefix that matches destination address

Destination Address Range	Link interface
11001000 00010111 00010*** *********	0
11001000 00010111 00011000 *********	1
11001000 00010111 00011*** *********	2
otherwise	3

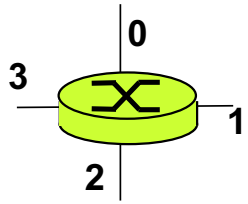
Which interface for IP destination address:

11001000 00010111 00010110 10100001

11001000 00010111 00011000 10101010

11001000 00010111 00011001 10101010

Packet forwarding using the destination host address



Destination:
11001000 00010111 00011001 10101010

Forwarding table

Destination Address Range	Link Interface
11001000 00010111 00010000 00000000 through **11001000 00010111 00010**111 11111111	0
11001000 00010111 00011000 00000000 through **11001000 00010111 00011000** 11111111	1
11001000 00010111 00011001 00000000 through **11001000 00010111 00011**111 11111111	2
otherwise	3

NTNU

Exercise - Using the forwarding table

SubnetNo	SubnetMask	NextHop
128.96.39.0	255.255.255.128	If 0
128.96.39.128	255.255.255.128	If1
128.96.40.0	255.255.255.128	R2
192.4.153.0	255.255.255.192	R3
0.0.0.0	0.0.0.0	R4

- What is the out interface for packets addressed to
  - 128.96.39.10
  - 128.96.40.12
  - 128.96.40.151
  - 192.4.153.17
  - 192.4.153.90

Exercise - Using the forwarding table

SubnetNo	SubnetMask	NextHop
128.96.39.0	255.255.255.128	If 0
128.96.39.128	255.255.255.128	If1
128.96.40.0	255.255.255.128	R2
192.4.153.0	255.255.255.192	R3
0.0.0.0	0.0.0.0	R4

Default route

- What is the out interface for packets addressed to
  - 128.96.39.10
  - 128.96.40.12
  - 128.96.40.151
  - 192.4.153.17
  - 192.4.153.90

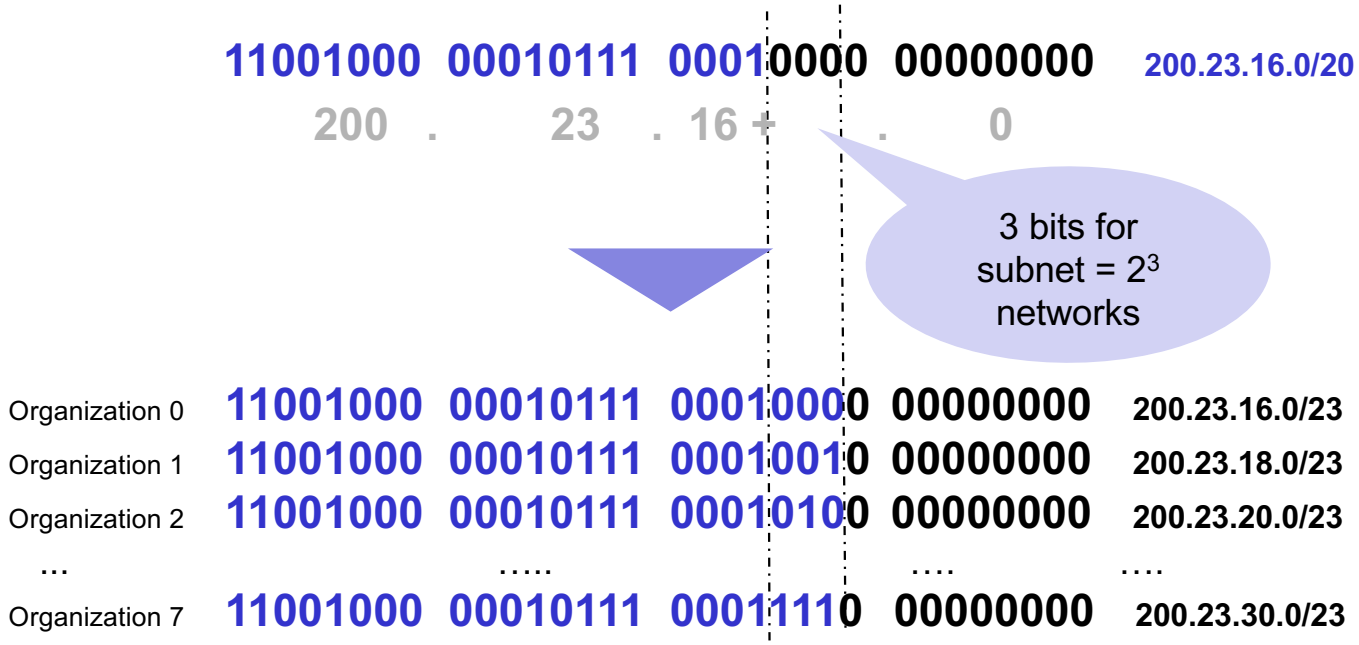
IP-dst address 192.4.153.90 = 11000000.00000100.10011001.01011010

192.4.153.0/26 192.4.153.0 = 11000000.00000100.10011001.00xxxxxx

Match on 4th entry in forwarding table? = match . match . match . nomatch

How to get an IP subnet?

- E.g. get address as allocated portion of provider ISP´s address space



Exam subnetting: Assume that you have been assigned the 200.35.1.0/24 network block

- Define a network prefix that allows the creation of 20 hosts on each subnet
  - **Needs 5 bits ($2^5 > 20$), (32-5 = 27) network prefix of /27**
- What is the maximum number of hosts that can be assigned to each subnet?
  - **$2^5 - 2 = 30$** host part = 0s: subnet, host part = 1s: broadcast
- What is the maximum number of subnets that can be defined?
  - **$2^3 = 8$ 200.35.1.xxx/27**
- Specify the /27 subnets of 200.35.1.0/24
  - Subnet #0: 11001000.00100011.00000001. **000** 00000 **= 200.35.1.0/27**
  - Subnet #1: 11001000.00100011.00000001. **001** 00000 **= 200.35.1.32/27**
  - Subnet #2: 11001000.00100011.00000001. **010** 00000 **= 200.35.1.64/27**

    . . .
  - Subnet #6: 11001000.00100011.00000001. **110** 00000 **= 200.35.1.192/27**
  - Subnet #7: 11001000.00100011.00000001. **111** 00000 **= 200.35.1.224/27**

Exam subnetting: host addresses and broadcast

- List the range of host addresses that can be assigned to subnet #6
  = 11001000.00100011.00000001.110 00000 = **200.35.1.192/27**

  - Host #1: 11001000.00100011.00000001.110 **00001 = 200.35.1.193/27**
  - Host #2: 11001000.00100011.00000001.110 **00010 = 200.35.1.194/27**
  - Host #3: 11001000.00100011.00000001.110 **00011 = 200.35.1.195/27**

    :
  - Host #29: 11001000.00100011.00000001.110 **11101 = 200.35.1.221/27**
  - Host #30: 11001000.00100011.00000001.110 **11110 = 200.35.1.222/27**

- What is the broadcast address for subnet 200.35.1.192/27?
  11001000.00100011.00000001.110 **11111 = 200.35.1.223**

IP addressing: how to get a block?

- How does an ISP get block of addresses?
- ICANN: Internet Corporation for Assigned Names and Numbers http://www.icann.org/
  - IANA: Internet Assigned Numbers Authority
    - number resources, http://www.iana.org/numbers
    - manages DNS, http://www.iana.org/domains
    - protocol registries, http://www.iana.org/protocols

Internet Assigned Numbers Authority

Registry	Area Covered
AfriNIC	Africa Region
APNIC	Asia/Pacific Region
ARIN	North America Region
LACNIC	Latin America and some Caribbean Islands
RIPE NCC	Europe, the Middle East, and Central Asia

Observations

- Is the address space too small?
  - Yes, especially when all mobiles also need a global unique address

- The life time for the address space is extended because of

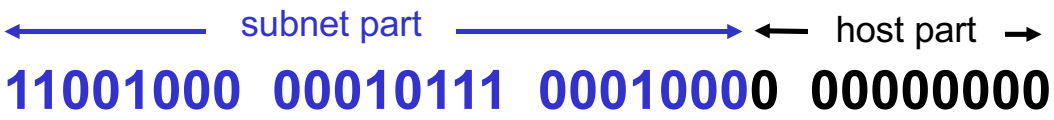
1. CIDR (Classless InterDomain Routing)
2. DHCP (Dynamic Host Configuration Protocol)
3. Private intranets (RFC1918)
   - Private addresses must be managed in the public internet
4. NAT (Network Address Translation)

IP addressing

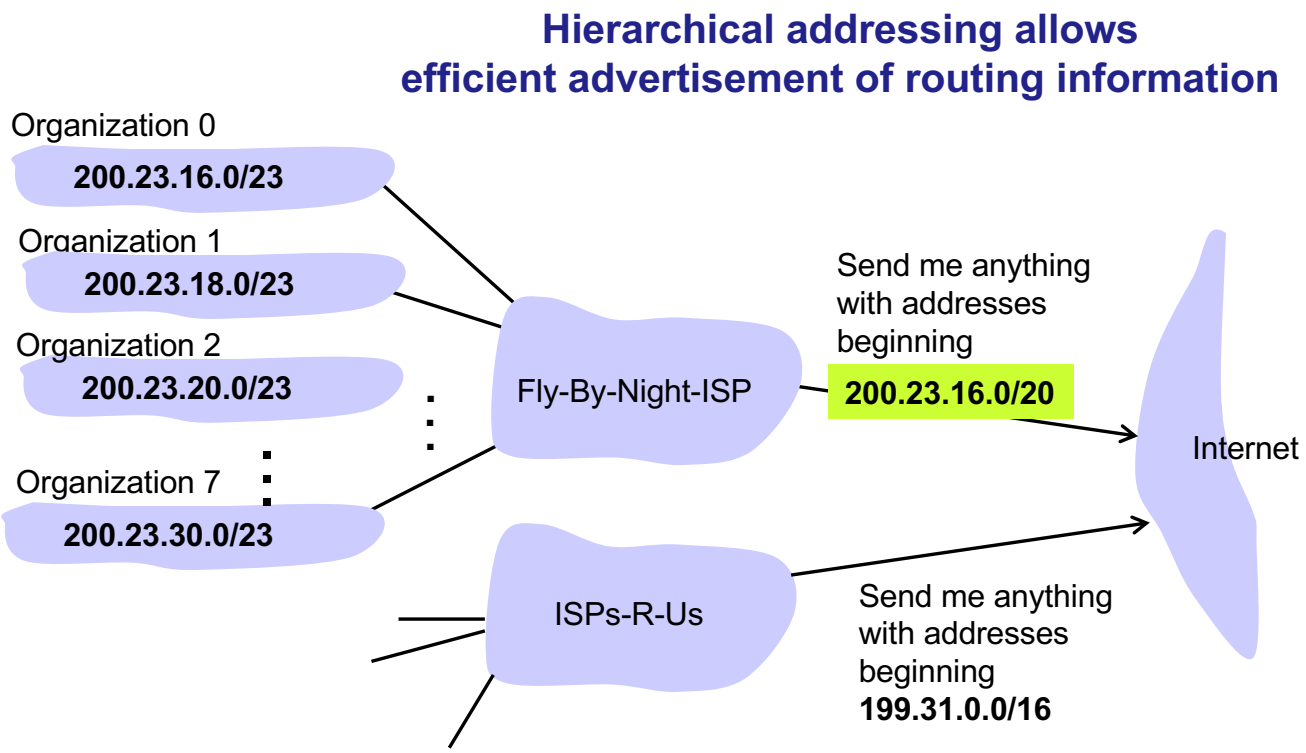
CIDR – Classless InterDomain Routing

- Subnet portion of address of arbitrary length
- Address format: **a.b.c.d/x**, where **x** is # bits in subnet portion of address

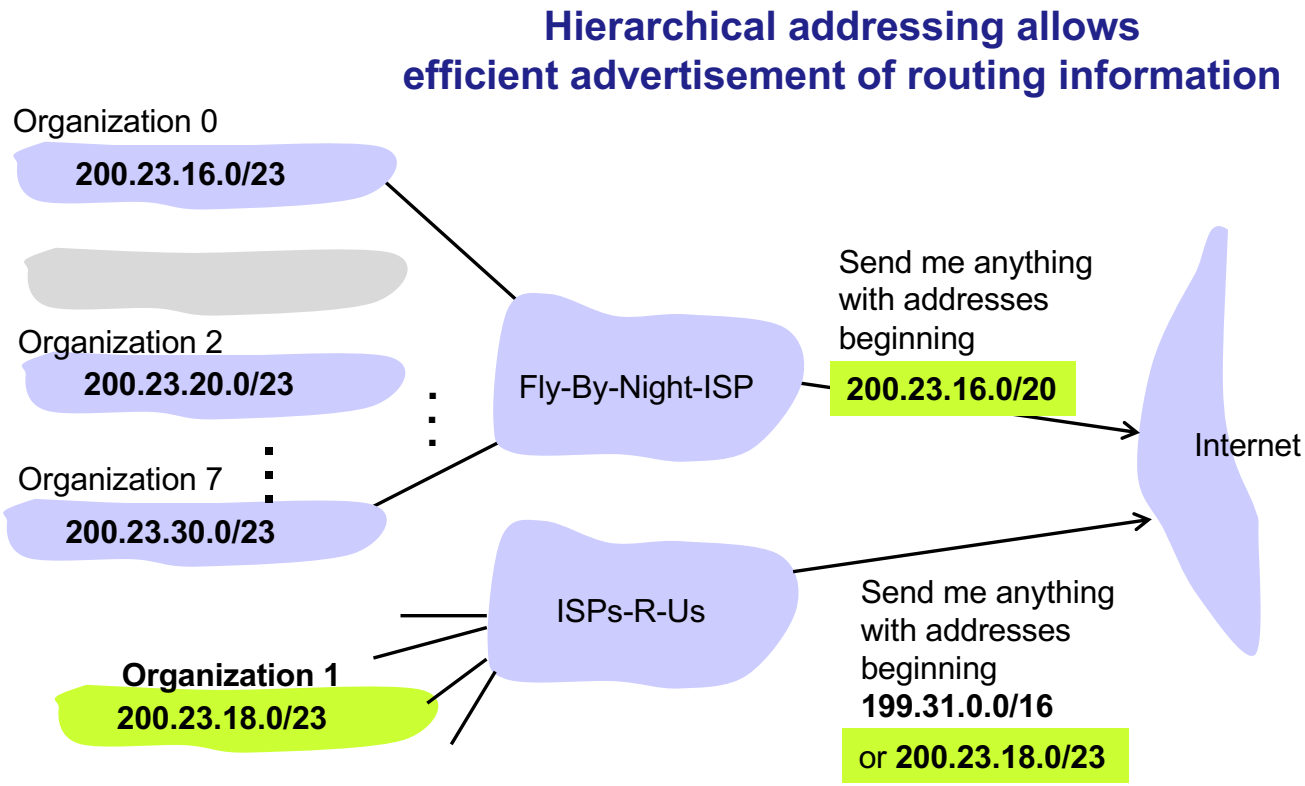
200.23.16.0/23



Route aggregation to reduce routing/forwarding table size



Route aggregation: more specific route

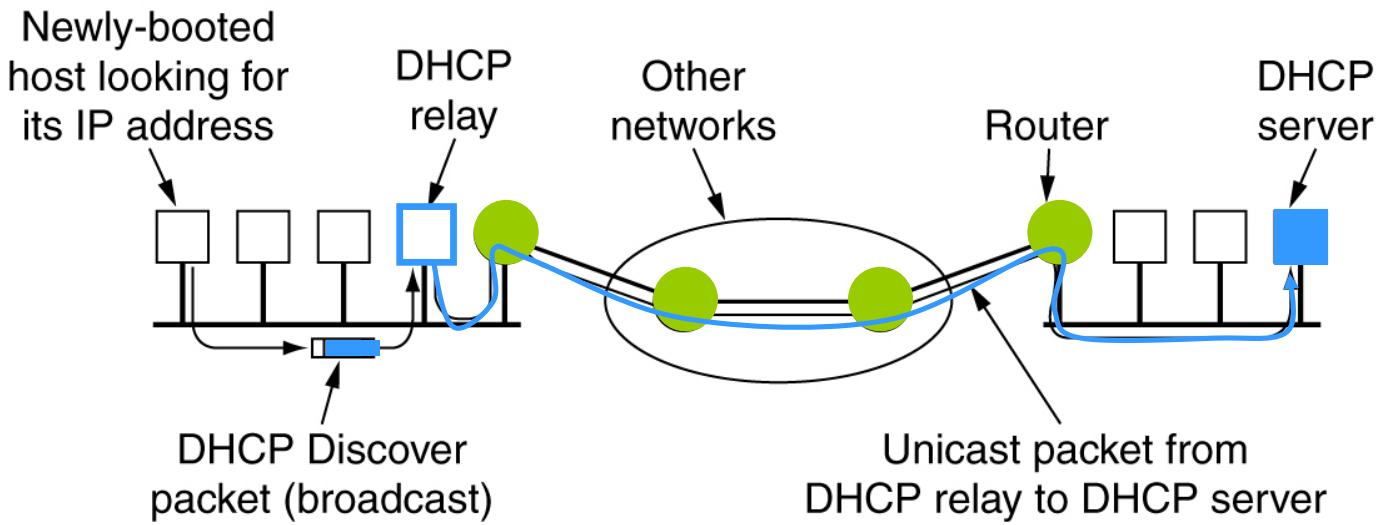


NTNU

2

IP address: how does a host get one?
DHCP: Dynamic Host Configuration Protocol

- Dynamically get address from a server (“plug-and-play”)



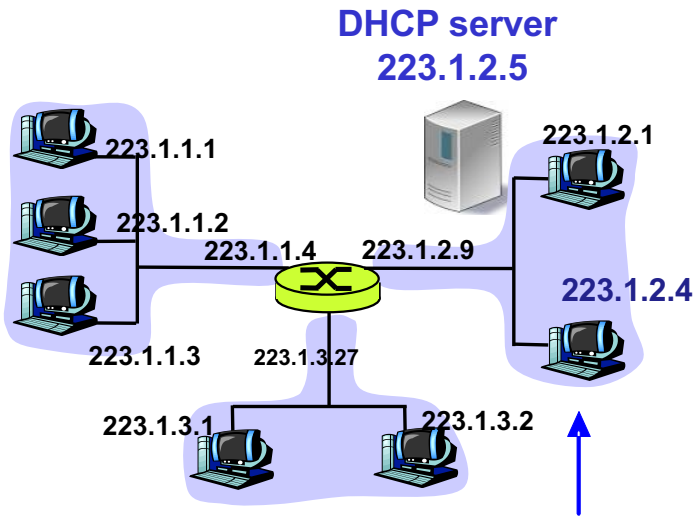
NTNU

Dynamic Host Configuration Protocol (DHCP)

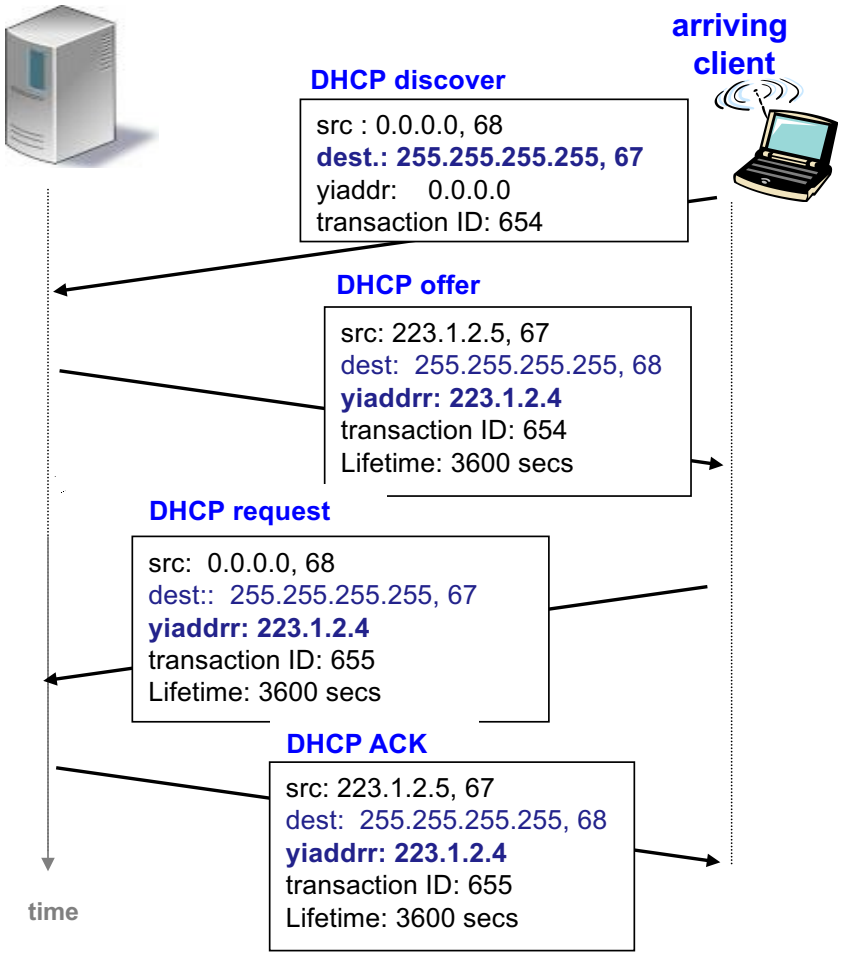
- Hosts dynamically obtain IP address from network server when joins network
  - can renew its lease on address in use
  - allows reuse of addresses (only hold address while connected/“on”)
  - support for mobile users who want to join network (more shortly)

- DHCP messages
  - host broadcasts **“DHCP discover”** msg [optional]
  - DHCP server responds with **“DHCP offer”** msg [optional]
  - host requests IP address: **“DHCP request”** msg
  - DHCP server sends address: **“DHCP ack”** msg

DHCP client-server scenario



arriving DHCP client needs address in this network



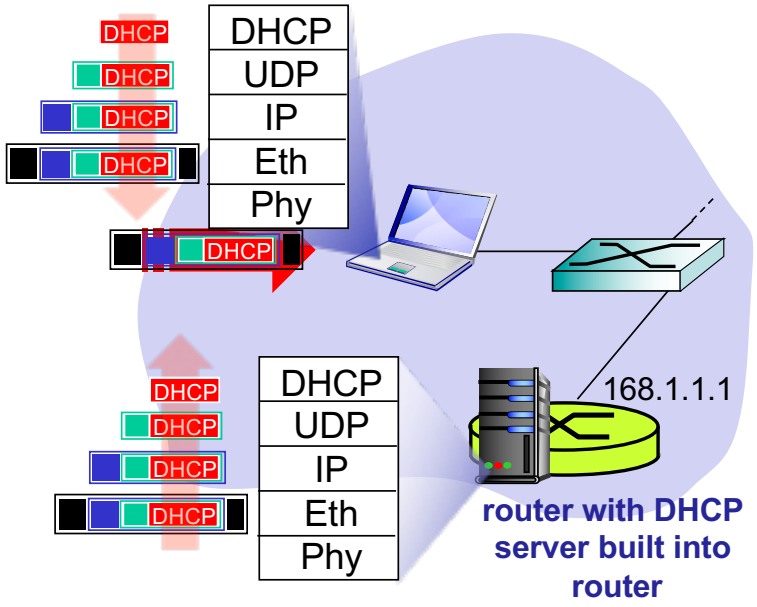
The 'xid' field is used by the client to match incoming DHCP messages with pending requests. A DHCP client MUST choose 'xid's in such a way as to minimize the chance of using an 'xid' identical to one used by another client. For example, a client may choose a different, random initial 'xid' each time the client is rebooted, and subsequently use sequential 'xid's until the next reboot. Selecting a new 'xid' for each retransmission is an implementation decision. A client may choose to reuse the same 'xid' or select a new 'xid' for each retransmitted message.

Normally, DHCP servers and BOOTP relay agents attempt to deliver DHCPOFFER, DHCPACK and DHCPNAK messages directly to the client using uicast delivery. The IP destination address (in the IP header) is set to the DHCP 'yiaddr' address and the link-layer destination address is set to the DHCP 'chaddr' address. Unfortunately, some client implementations are unable to receive such unicast IP datagrams until the implementation has been configured with a valid IP address (leading to a deadlock in which the client's IP address cannot be delivered until the client has been configured with an IP address).

A client that cannot receive unicast IP datagrams until its protocol software has been configured with an IP address SHOULD set the BROADCAST bit in the 'flags' field to 1 in any DHCPDISCOVER or DHCPREQUEST messages that client sends. The BROADCAST bit will provide a hint to the DHCP server and BOOTP relay agent to broadcast any messages to the client on the client's subnet. A client that can receive unicast IP datagrams before its protocol software has been configured SHOULD clear the BROADCAST bit to 0. The BOOTP clarifications document discusses the ramifications of the use of the BROADCAST bit [21].

Example: DHCP request

- Connecting laptop use DHCP to get
  IP address, address of first-hop router, address of DNS server



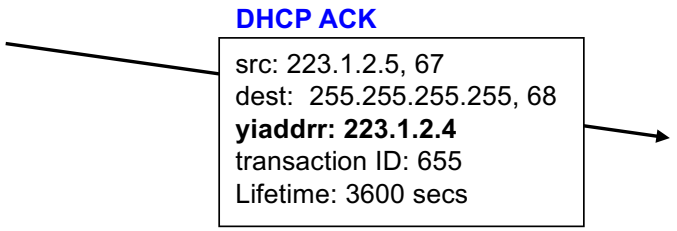
DHCP request

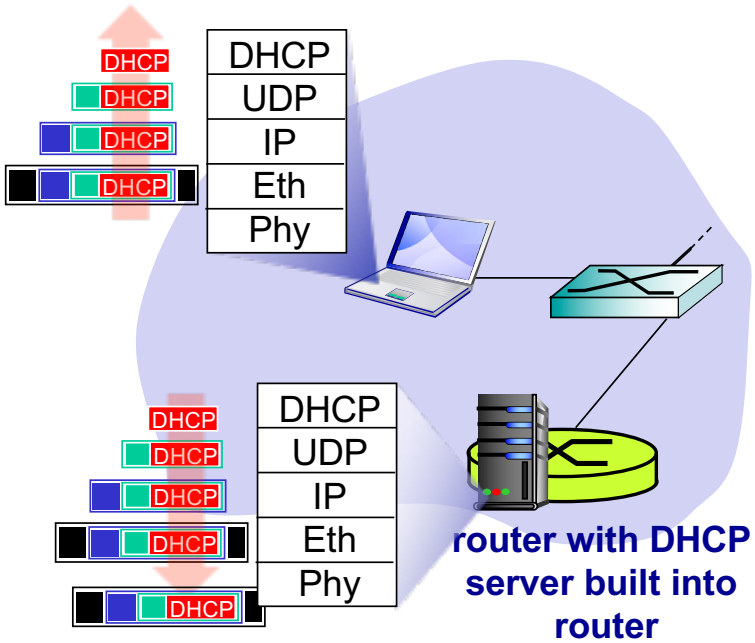
src: 0.0.0.0, 68
dest:: 255.255.255.255, 67
yiaddrr: 223.1.2.4
transaction ID: 655
Lifetime: 3600 secs

- DHCP request encapsulated in UDP, encapsulated in IP, encapsulated in 802.3 Ethernet

- Ethernet frame broadcast (dest: FFFFFFFFFFFF) on LAN, received at router running DHCP server

DHCP: example





- DHCP server formulates DHCP ACK containing
  - client's IP address
  - IP address of first-hop router for client
  - name & IP address of DNS server

- Client now has an
  - IP address
  - address of first hop router
  - DNS server

NTNU

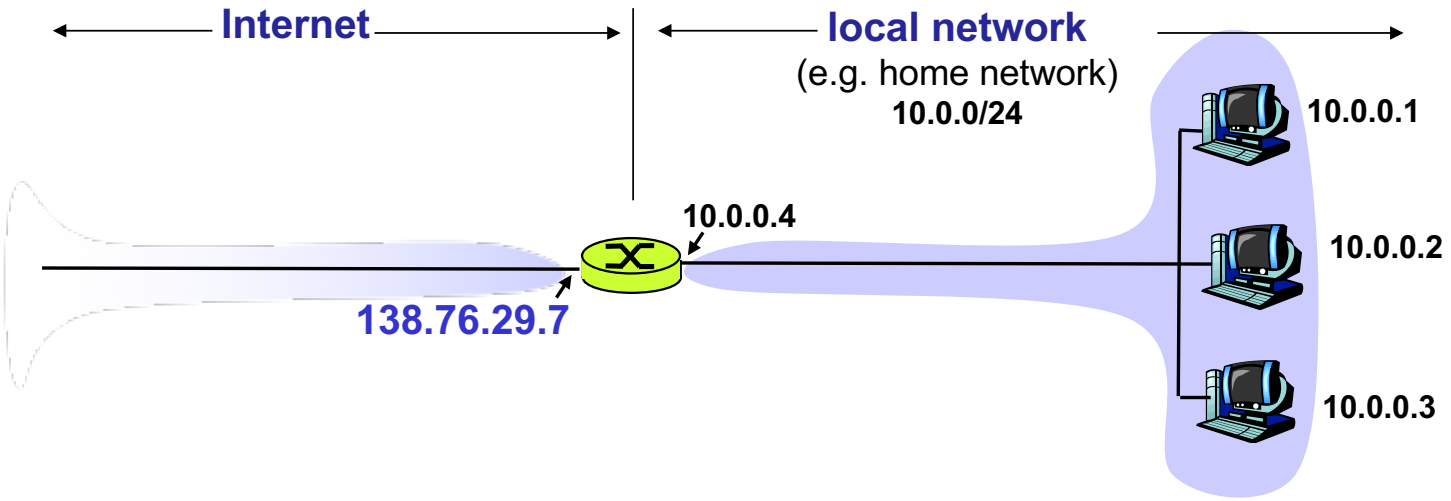
3 Private addresses increased the life time of the IP address space

- IP addresses were globally unique
  - not anymore because of private intranets (RFC 1918) and dynamic IP addressing (DHCP)
- To extend the life time of the IP address space the following addresses are allowed for business internal private internets
  - **10.0.0.0 - 10.255.255.255 (10/8 prefix)**
  - **172.16.0.0 - 172.31.255.255 (172.16/12 prefix)**
  - **192.168.0.0 - 192.168.255.255 (192.168/16 prefix)**
- No coordination and application for private addresses
  - i.e. addresses are not unique
- Public addresses must be applied for (e.g. RIPE for Europe)

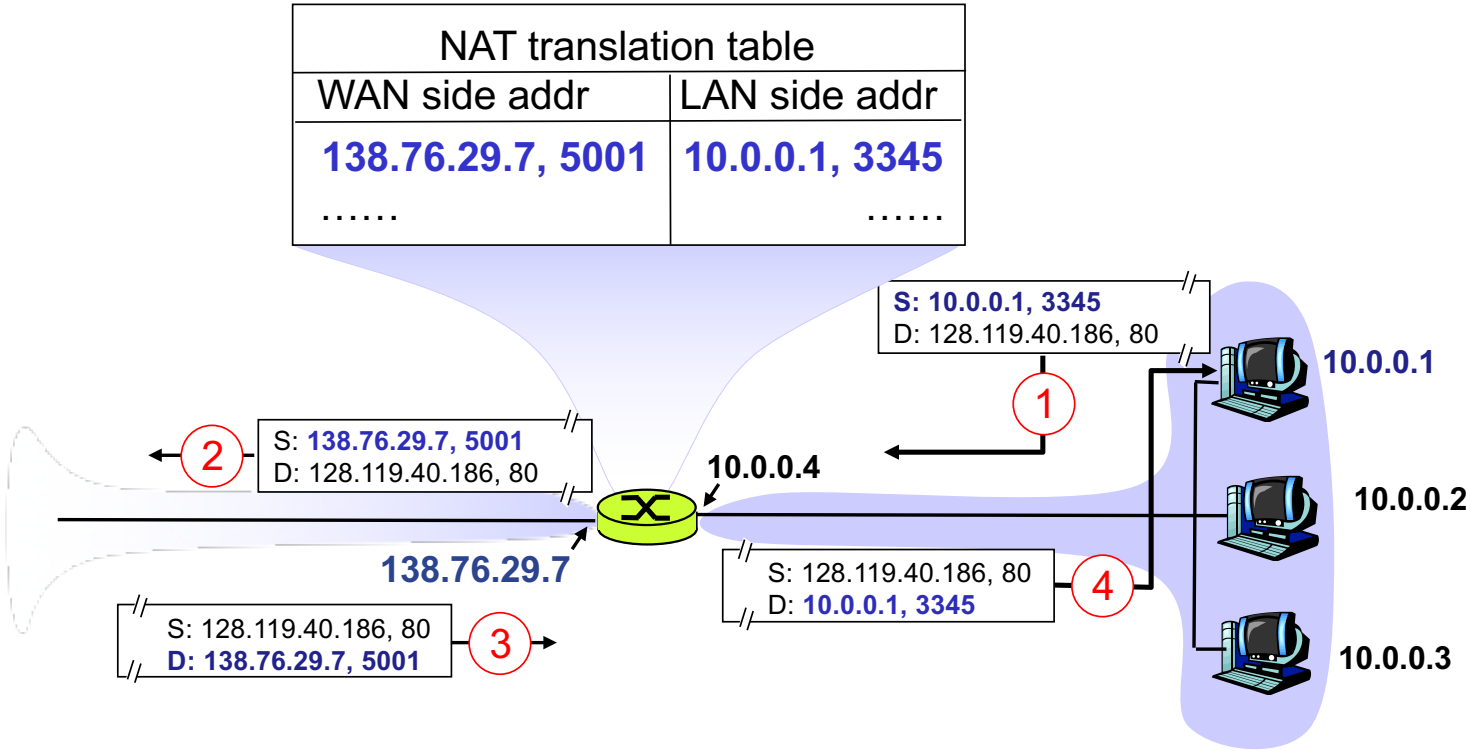
NTNU

4 Network Address Translation (NAT) is required when using private addresses

- Just one public IP address for all devices from ISP: a range of addresses is not needed
- Can change addresses of devices in local network without notifying outside world
- Can change ISP without changing addresses of devices in local network
- Devices inside local net not explicitly addressable, visible by outside world (a security plus)



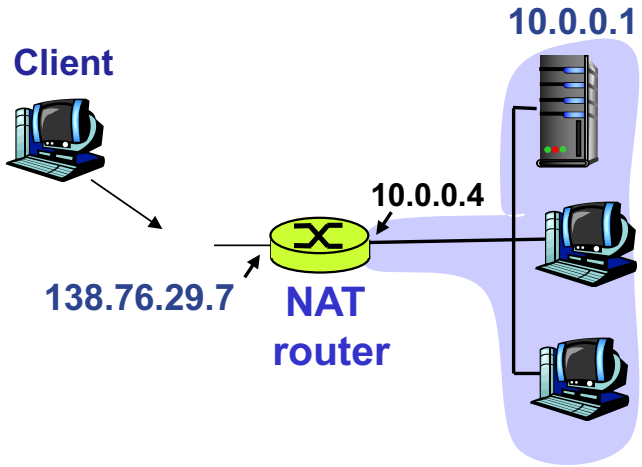
NAT translates between private and public addresses by updating IP address and transport protocol port number



NAT Network Address Translation

NAT traversal problem – static configuration of address mapping

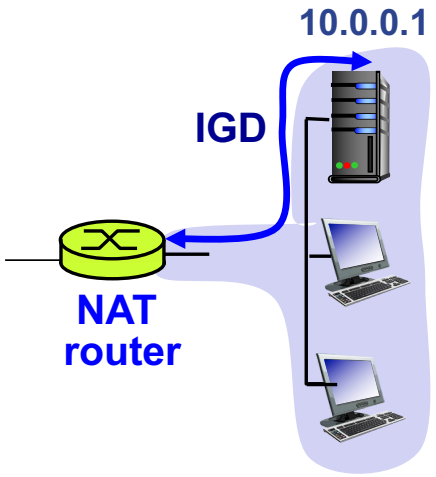
- Client wants to connect to server with address **10.0.0.1**
  - **10.0.0.1** local to LAN
  - only one externally visible (NATted) address: **138.76.29.7**
- Solution 1: **statically configure NAT** to forward incoming connection requests at given port to server
  - e.g. (**138.76.29.7**, port 25000) always forwarded to **10.0.0.1** port 25000



NAT traversal problem – Universal Plug and Play

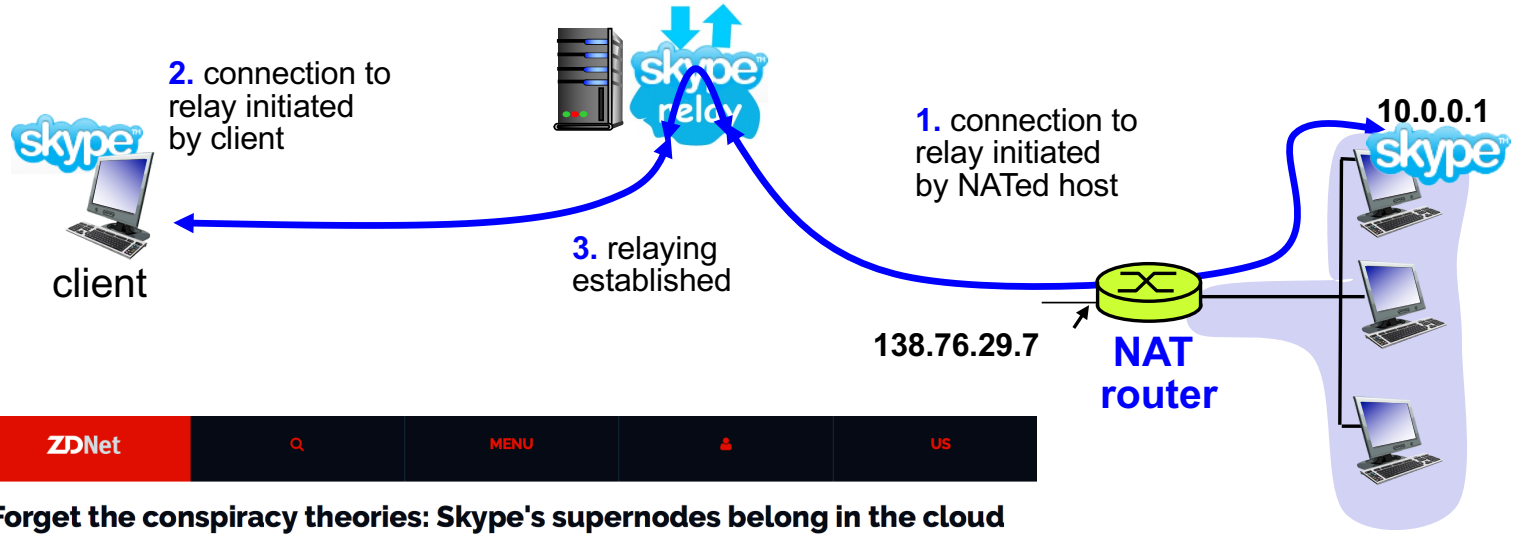
- Solution 2: **Universal Plug and Play (UPnP) Internet Gateway Device (IGD) Protocol**

- Allows NATed host to
  - learn public IP address (**138.76.29.7**)
  - add/remove port mappings (with lease times)
  - i.e. automate static NAT port map configuration



NAT traversal problem – relaying through 3rd party

- Solution 3: NATed client establishes connection to **relay**
  - external client connects to relay
  - relay bridges packets between to connections



ZDNet | MENU | US

Forget the conspiracy theories: Skype's supernodes belong in the cloud

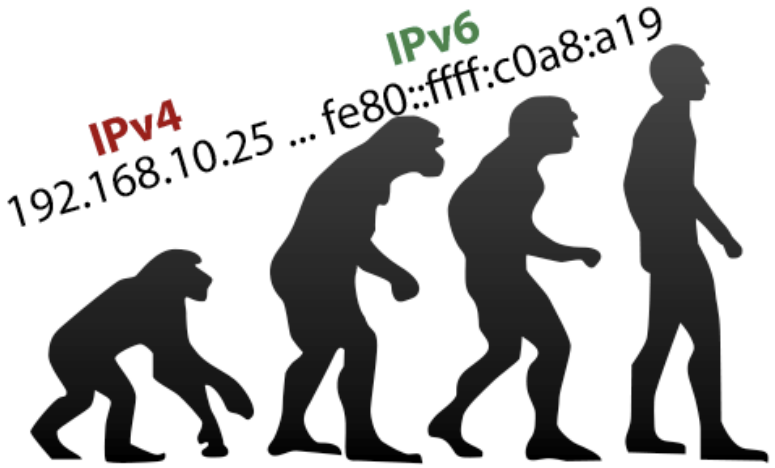
Putting Skype's supernodes in the Microsoft datacentres is about improving performance and not appropriating bandwidth

By Mary Branscombe for 500 words into the future | July 27, 2012 -- 13:52 GMT (06:52 PDT) | Topic: Microsoft

NAT Network Address Translation

NAT (network address translation) is controversial

- 16-bit port-number field
  - 60,000 simultaneous connections with a single LAN-side address!
- Violates end-to-end argument – routers should only process up to layer 3
- NAT possibility must be taken into account by app designers, e.g. P2P applications
- Address shortage should instead be solved by IPv6



Network layer

4.1 Introduction

~~4.2 Virtual circuit and datagram~~

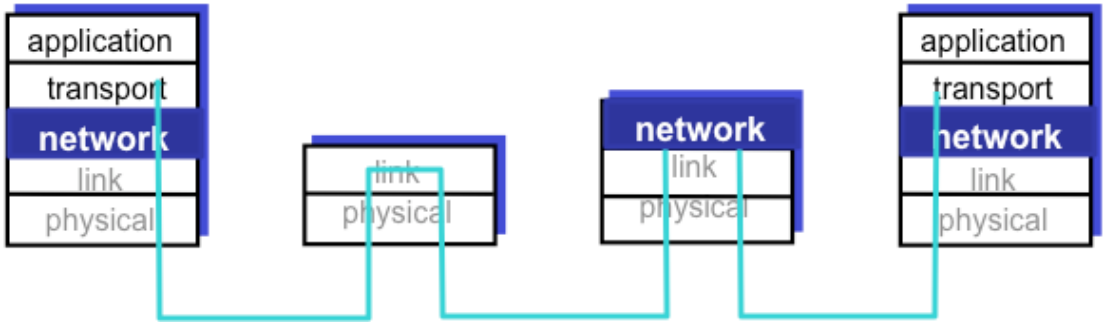
4.3 What's inside a router
- Input processing
- Switching
- Output processing
- Queuing

4.4 IP: Internet Protocol
- Datagram format
- IPv4 addressing
- **ICMP**
- IPv6

4.5 Routing algorithms
- ~~Link state vs Distance vector~~
- ~~Hierarchical routing~~

8.7 Network layer security

8.9.1 Firewalls



Internet Control Message Protocol (ICMP) is used when something goes wrong

- Used by hosts & routers to communicate network-level information
  - **error reporting**: unreachable host, network, port, protocol
  - **echo request/reply** (used by ping)
- Uses IP for transport
- **ICMP message**: type, code plus first 8 bytes of IP datagram causing error

Type	Code	description
0	0	echo reply (ping)
3	0	dest. network unreachable
3	1	dest host unreachable
3	2	dest protocol unreachable
3	3	dest port unreachable
3	6	dest network unknown
3	7	dest host unknown
4	0	source quench (congestion control - not used)
8	0	echo request (ping)
9	0	route advertisement
10	0	router discovery
11	0	TTL expired
12	0	bad IP header

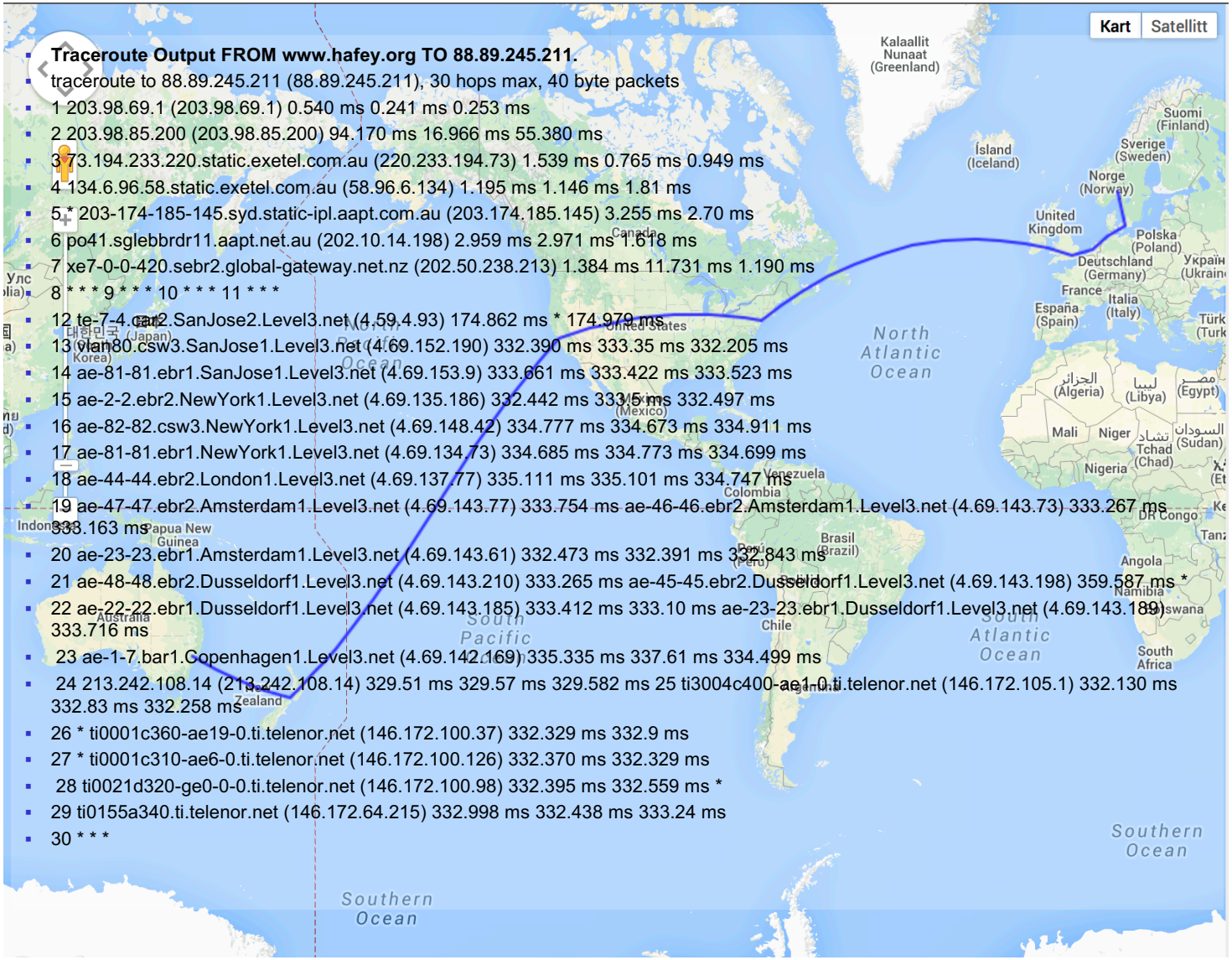
Traceroute and ICMP

- Source sends series of UDP segments to destination
  - First has TTL =1
  - Second has TTL=2, etc.
  - Unlikely port number
  - Each TTL 3 times
- When nth datagram arrives to nth router:
  - Router discards datagram
  - And sends to source an **ICMP** message (type 11, code 0 **TTL expired**)
  - Message includes name of router & IP address

```
 1  192.168.10.1 (192.168.10.1)  4.885 ms  3.056 ms  2.566 ms
 2  129.109-247-190.customer.lyse.net (109.247.190.129)  9.226 ms  2.515 ms  2.265 ms
 3  224.79-161-73.customer.lyse.net (79.161.73.224)  5.701 ms  1.575 ms  3.434 ms
 4  164.79-161-73.customer.lyse.net (79.161.73.164)  7.873 ms  2.699 ms  7.472 ms
 5  246.109-247-30.customer.lyse.net (109.247.30.246)  33.093 ms  37.843 ms  40.978 ms
 6  129.81-167-46.customer.lyse.net (81.167.46.129)  35.376 ms  3.339 ms  3.989 ms
 7  stolav-gw2.uninett.no (193.156.120.1)  8.105 ms  3.741 ms  6.613 ms
 8  hovedbygget-gw.uninett.no (128.39.255.166)  45.774 ms  40.917 ms  42.226 ms
 9  narvik-gw3.uninett.no (128.39.255.102)  55.937 ms  53.831 ms  53.879 ms
10  narvik-gw1.uninett.no (128.39.255.230)  44.709 ms  22.578 ms  22.185 ms
11  harstad-gw1.uninett.no (128.39.255.34)  24.485 ms  21.804 ms  22.212 ms
12  harstad-gw3.uninett.no (128.39.231.86)  22.428 ms  25.036 ms  22.265 ms
13  svalbard-gw.uninett.no (128.39.254.10)  38.111 ms  37.850 ms  36.783 ms
14  unis-gsw.uninett.no (128.39.47.158)  41.828 ms  37.715 ms  36.686 ms
15  srv03.unis.no (158.39.11.240)  39.720 ms  37.102 ms  42.116 ms
```

- When ICMP message arrives, source calculates RTT
- UDP segment eventually arrives at destination host
  - Destination returns **ICMP** "**port unreachable**" packet (type 3, code 3)
  - When source gets this ICMP it stops

traceroute.org


Kart
Satellitt
Traceroute Output FROM www.hafey.org TO 88.89.245.211.
traceroute to 88.89.245.211 (88.89.245.211), 30 hops max, 40 byte packets
1 203.98.69.1 (203.98.69.1) 0.540 ms 0.241 ms 0.253 ms
2 203.98.85.200 (203.98.85.200) 94.170 ms 16.966 ms 55.380 ms
3 73.194.233.220.static.exetel.com.au (220.233.194.73) 1.539 ms 0.765 ms 0.949 ms
4 134.6.96.58.static.exetel.com.au (58.96.6.134) 1.195 ms 1.146 ms 1.81 ms
5 * 203-174-185-145.syd.static-ipl.aapt.com.au (203.174.185.145) 3.255 ms 2.70 ms
6 po41.sglebbrdr11.aapt.net.au (202.10.14.198) 2.959 ms 2.971 ms 1.618 ms
7 xe7-0-0-420.sebr2.global-gateway.net.nz (202.50.238.213) 1.384 ms 11.731 ms 1.190 ms
8 * * * 9 * * * 10 * * * 11 * * *
12 te-7-4.car2.SanJose2.Level3.net (4.59.4.93) 174.862 ms * 174.979 ms
13 vlan80.csw3.SanJose1.Level3.net (4.69.152.190) 332.390 ms 333.35 ms 332.205 ms
14 ae-81-81.ebr1.SanJose1.Level3.net (4.69.153.9) 333.661 ms 333.422 ms 333.523 ms
15 ae-2-2.ebr2.NewYork1.Level3.net (4.69.135.186) 332.442 ms 333.5 ms 332.497 ms
16 ae-82-82.csw3.NewYork1.Level3.net (4.69.148.42) 334.777 ms 334.673 ms 334.911 ms
17 ae-81-81.ebr1.NewYork1.Level3.net (4.69.134.73) 334.685 ms 334.773 ms 334.699 ms
18 ae-44-44.ebr2.London1.Level3.net (4.69.137.77) 335.111 ms 335.101 ms 334.747 ms
19 ae-47-47.ebr2.Amsterdam1.Level3.net (4.69.143.77) 333.754 ms ae-46-46.ebr2.Amsterdam1.Level3.net (4.69.143.73) 333.267 ms 333.163 ms
20 ae-23-23.ebr1.Amsterdam1.Level3.net (4.69.143.61) 332.473 ms 332.391 ms 332.843 ms
21 ae-48-48.ebr2.Dusseldorf1.Level3.net (4.69.143.210) 333.265 ms ae-45-45.ebr2.Dusseldorf1.Level3.net (4.69.143.198) 359.587 ms *
22 ae-22-22.ebr1.Dusseldorf1.Level3.net (4.69.143.185) 333.412 ms 333.10 ms ae-23-23.ebr1.Dusseldorf1.Level3.net (4.69.143.189) 333.716 ms
23 ae-1-7.bar1.Copenhagen1.Level3.net (4.69.142.169) 335.335 ms 337.61 ms 334.499 ms
24 213.242.108.14 (213.242.108.14) 329.51 ms 329.57 ms 329.582 ms 25 ti3004c400-ae1-0.ti.telenor.net (146.172.105.1) 332.130 ms 332.83 ms 332.258 ms
26 * ti0001c360-ae19-0.ti.telenor.net (146.172.100.37) 332.329 ms 332.9 ms
27 * ti0001c310-ae6-0.ti.telenor.net (146.172.100.126) 332.370 ms 332.329 ms
28 ti0021d320-ge0-0-0.ti.telenor.net (146.172.100.98) 332.395 ms 332.559 ms *
29 ti0155a340.ti.telenor.net (146.172.64.215) 332.998 ms 332.438 ms 333.24 ms
30 * * *
Kalaallit Nunaat (Greenland)
Ísland (Iceland)
Suomi (Finland)
Sverige (Sweden)
Norge (Norway)
United Kingdom
Polska (Poland)
Deutschland (Germany)
France
España (Spain)
Italia (Italy)
North Atlantic Ocean
South Atlantic Ocean
South Pacific Ocean
Southern Ocean
Southern Ocean
Canada
United States
Mexico
Venezuela
Colombia
Brasil (Brazil)
Peru
Chile
Argentina
Australia
Papua New Guinea
Indonesia
Zealand
Mali
Niger
Nigeria
Tchad (Chad)
Libya
Egypt
Algeria
Sudan
DR Congo
Angola
Namibia
South Africa

Network layer – in each and every node and end system

4.1 Introduction

~~4.2 Virtual circuit and datagram~~

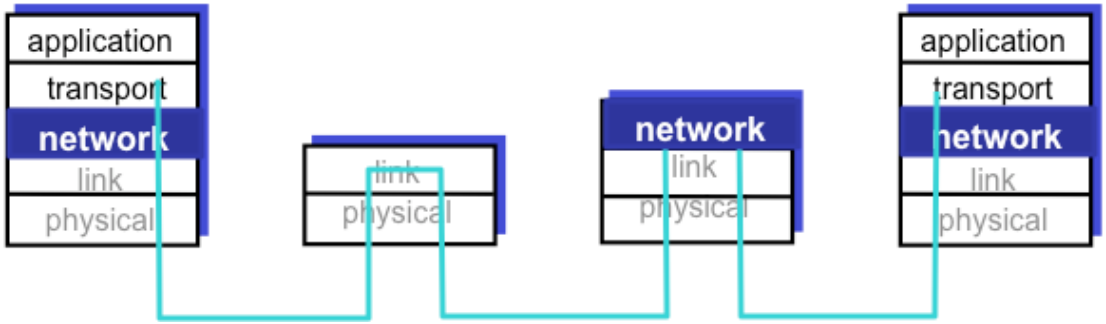
4.3 What's inside a router
- Input processing
- Switching
- Output processing
- Queuing

4.4 IP: Internet Protocol
- Datagram format
- IPv4 addressing
- ICMP
- **IPv6**

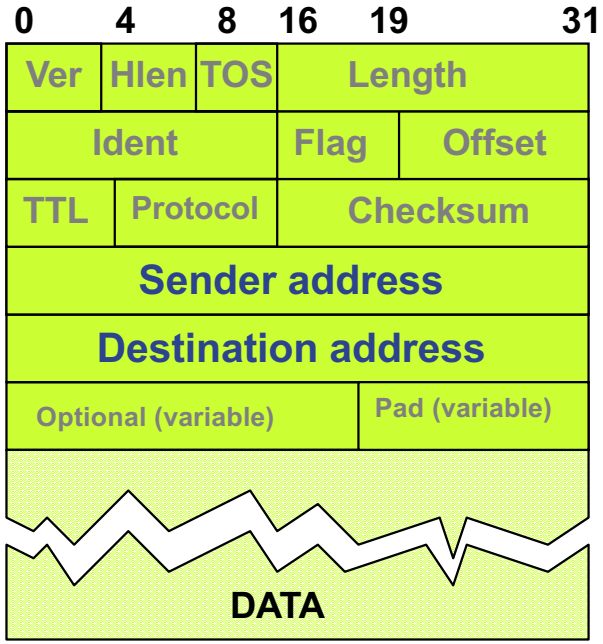
4.5 Routing algorithms
- ~~Link state vs Distance vector~~
- ~~Hierarchical routing~~

8.7 Network layer security

8.9.1 Firewalls



IPv4 32-bit address limits the address space



IP version 4

- < 2^{32} addresses
- Inefficient utilization of address space due to bad planning
- IPv4 **address structure** gives **large routing tables**
  - Bad address hierarchy
  - CIDR targets this
- IPv4 **missing support** for **new services** (but has been added)
  - QoS guarantee
  - Mobility, multicast
  - Security mechanisms
- Header format helps speed in processing/forwarding

Observations

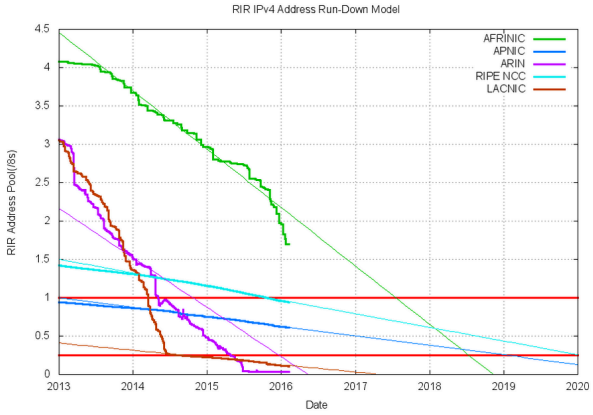
IPv4 Address Report

This report generated at 07-Feb-2016 08:18 UTC.

IANA Unallocated Address Pool Exhaustion:
03-Feb-2011

Projected RIR Address Pool Exhaustion Dates:

RIR	Projected Exhaustion Date	Remaining Addresses in RIR Pool (/8s)
APNIC:	**19-Apr-2011** (actual)	0.6094
RIPE NCC:	**14-Sep-2012** (actual)	0.9405
LACNIC:	**10-Jun-2014** (actual)	0.1066
ARIN:	**24 Sep-2015** (actual)	
AFRINIC:	**08-May-2018**	1.6947



Projection of consumption of Remaining RIR Address Pools

http://www.potaroo.net/tools/ipv4/

The central IANA pool of IPv4 was depleted on February 3 of 2011

The remaining IPv4 addresses in the world are now in possession of the five Regional Registrars and its members. While no one can predict the final exhaustion date of IPv4 in each region with certainty, we all know it is inevitable.

http://www.ipv4depletion.com/?page

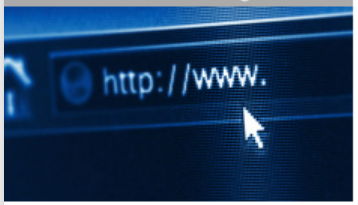
http://www.ripe.net/internet-coordination/ipv4-exhaustion/ipv4-available-pool-graph
https://www.ripe.net/publications/docs/ripe-649

NTNU

mobilen.no

NYHETER | GUIDER | TESTER | ARKIVET | TEKNOFIL.NO | SPEEDOMETER | PRISSØK

Tomt for IP-adresser om få dager



Tomt for IP-adresser om få dager

Skrevet av Arsæll Benediktsson

Publisert : 31.01.2011 13:18 / Oppdatert : 31.01.2011 13:54

Kollapser internett?

Nettstedet ipv6.he.net viser nedtelling av IPv4, det vil si den teller ned til den dagen det ikke er flere IPv4-adresser igjen. I skrivende stund er det kun et par dager igjen.

Eksperter er stort sett enige i at internett ikke kommer til å kollapse selv om IPv4 brukes opp. Noen forstyrrelser er riktignok ventet når IPv6-systemet tar over, i form av at enkelte nettstedet blir tregere eller ikke tilgjengelige, men dette blir mer på grunn av feilkonfigurasjon av nettverksutstyr.

Les også: – Det blir ingen internettkrise

Siste IPv4-adressene blir altså etter alt å dømme delt ut i løpet av denne uken, men det vil ta flere måneder før disse adressene faktisk blir tatt i bruk. Onsdag 8. juni er det planlagt en test av det nye systemet, men allerede nå er det mulig for enkeltbrukere å sjekke deres IPv6-status via nettstedet test-ipv6.com

Les også: **Bekymret for IPv6-overgangen**

(*Kilde: Cnet*)

TAGS: Internett og nettverk

– Det blir ingen internettkrise

Norske eksperter mener det ikke vil oppstå noen krise ved overgangen fra IPv4.

annonse:

ARSÆLL BENEDIKTSSON 10. jan 2011 09:28

Del på Facebook 4

Det er for lenge siden blitt varslet at det snart ikke vil være flere IPv4-adresser tilgjengelige, og derfor blir IPv6-adresser mer og mer aktuelle. I slutten av februar vil de siste IPv4-adressene deles ut.

- Les også: Siste IPv4-adressene

Gisle Hannemyr, fra Universitetet i Oslo, ønsker å avdramatisere situasjonen. Riktinok viser løpende beregninger fra IPv4 Exhaustion Counter at det er tomt for IPv4-adresser den 21. februar 2011.

- Les også: Nedtellingen er i gang

Likevel mener Hannemyr det er forholdsvis udramatisk i og med at IPv6 er på plass. I verste fall er det snakk om litt kompetanseheving, det vil si diftsavdelinger må få lære seg å konfigurere IPv6-systemet. Videre må gammelt utstyr byttes ut.

Universitetet i Oslo har drevet IPv4 og IPv6 parallell ganske lenge. Blant fordelene med IPv6, sammenlignet med IPv4, er tilgang til flere funksjoner som kvalitetskontroll og multimedia, pluss tilgang til langt flere IP-adresser.

Mangel på IPv4-adresser vil i første rekke få konsekvenser for nettleverandører. De må begynne å selge IPv6-adresser til nye kunder, sier Hannemyr til nettstedet digi.no.

Heller ikke Telenor er bekymret. Det er ventet at IPv4 og IPV6 skal eksistere sammen i en forholdsvis lang periode, i hvert fall tre år. Selskapet har i løpet av første kvartal 2011 planer om å tilby prøvetjenester med IPv6 for bedriftsmarkedet. Telenor har ambisjon om at alle kunder som ønsker det, skal få IPv6 innen 2013.

- Les også: Overgangen er i gang
- Les også: Kritisk IPv4-nivå

UninettNytt nr 3 0912

IPv6 permanent påslått

Tekst: Lars Fuglevaag, UNINETT

En av de største endringene i internetthistorien er gjennomført.

8. juni 2012 sto de største nettstedene, nettverksleverandørene og utstyrsprodusentene sammen om å gjøre IPv6 (Internet Protocol version 6) tilgjengelig på permanent basis. Den globale markeringen var organisert av Internet Society, og var en viktig milepæl i den videre utviklingen av Internett. Facebook, Google og Microsoft var blant aktørene som sto bak lanseringen. Her i Norge har VG-nett lenge vært tilgjengelig over IPv6, og de fleste større internettleverandørene er i gang med å tilby IPv6 til sine kunder.

Avgjørende å erstatte IPv4

IPv4 har begrenset kapasitet, og på grunn av sterk vekst i bruken av Internett, er det nå praktisk talt tomt for nye IPv4-adresser. Følgen av dette vil være redusert funksjonalitet og økte kostnader for internettbrukere over hele kloden. Myndigheter og organisasjoner verden over er enige om at IPv6 er den eneste reelle løsningen på denne utfordringen.

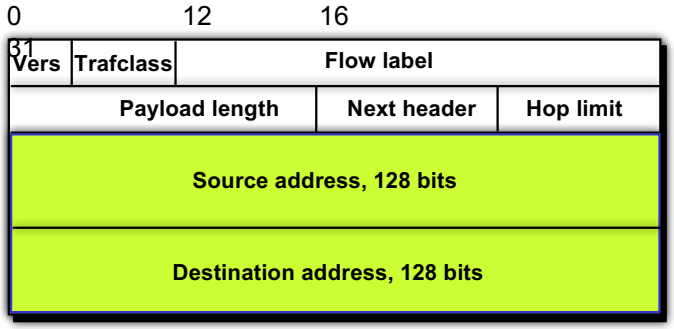
Forskningsnettet klart for IPv6

UNINETT har hatt IPv6 i drift siden 1998, og fra 2005 har UNINETTs kunder hatt tilbud om IPv6. Per i dag er godt over 60 prosent av høgskoler og universiteter i Norge klare for IPv6, og hos noen av disse institusjonene er over halvparten av brukerne på IPv6.

Mer informasjon:
Verdens IPv6-dag: www.worldipv6launch.org
UNINETT og IPv6: openwiki.uninett.no/gigacampus:ipv6
IPv6 i Norge: ipv6forum.no

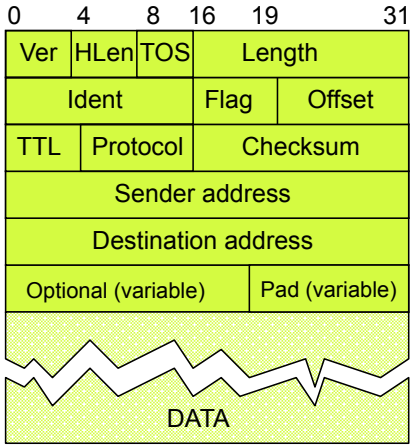
Kontaktpunkt: ipv6-drift@uninett.no

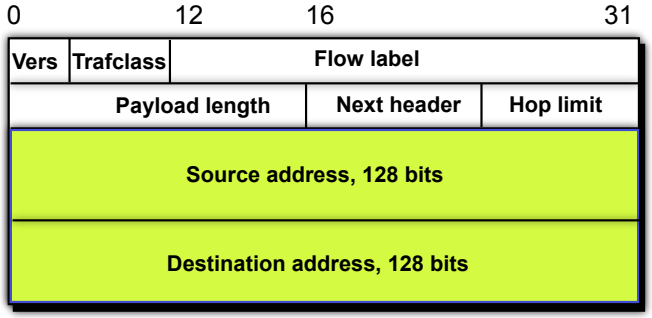
IPv6 fixed size header with 128-bit addresses



- Optimization and simplification
- **Traffic class**: identify priority among datagrams in flow
- **Flow Label**: identify flow of datagrams; from **IPv4 `TOS`** to **IPv6 `Traffic Class`** and **`Flow label`**
- **Payload length -** fixed 40-byte header, **`HLen`** not necessary; from **IPv4 `Total`** to **IPv6 `Payload`** length
- **Next header**: identify protocol for which data will be delivered; from **IPv4 `Protocol`** to **IPv6 `Next header`**
- **Hop Limit**; from **IPv4 `TTL`** to **IPv6 `Hop limit`**

Other IPv6 changes from IPv4





No longer present

- **Fragmentation and reassembly**: only by source and destination
- **Checksum**: removed entirely to reduce processing time at each hop
- **Options**: allowed, but outside of header, indicated by "**Next Header**" field

NTNU

IPv6 address: one interface or set of interfaces

- **Unicast**: An identifier for a single interface. A packet sent to a unicast address is delivered to the interface identified by that address.

- **Anycast**: An identifier for a set of interfaces (typically belonging to different nodes). A packet sent to an anycast address is delivered to one of the interfaces identified by that address (the "nearest" one, according to the routing protocols' measure of distance). Anycast addresses are taken from the unicast address spaces.

- **Multicast**: An identifier for a set of interfaces (typically belonging to different nodes). A packet sent to a multicast address is delivered to all interfaces identified by that address.

IPv6: 128-bit addressing

- Address notation eight groups of four hex ciphers
  x:x:x:x:x:x:x:x (x = 16 bits)

Address type	Binary prefix	IPv6 notation
Unspecified	00...0 (128 bits)	::/128
Loopback	00...1 (128 bits)	::1/128
Multicast	11111111	FF00::/8
Link-Local unicast	1111111010	FE80::/10
Global Unicast	(everything else)	

- A /64 subnet is the smallest IPv6 assignment that an end user can receive according to the current IPv6 assignment policy. So, an end user assigned the smallest IPv6 assignment will receive 2^{64} IPv6 addresses.
- An end user assigned the smallest IPv4 assignment will receive one single IPv4 address.

Source: RFC 4291 IPv6 addressing architecture

IPv6: 128-bit address types

Address notation eight groups of four hex ciphers (8*16=128 bits x:x:x:x:x:x:x:x (x = 16 bits)

Prefix	Designation and Explanation	IPv4 Equivalent
::/128	**Unspecified** This address may only be used as a source address by an initialising host before it has learned its own address.	0.0.0.0
::1/128	**Loopback** This address is used when a host talks to itself over IPv6. This often happens when one program sends data to another.	127.0.0.1
::ffff/96 Example: ::ffff:192.0.2.47	**IPv4-Mapped** These addresses are used to embed IPv4 addresses in an IPv6 address. One use for this is in a dual stack transition scenario where IPv4 addresses can be mapped into an IPv6 address. See RFC 4038 for more details.	There is no equivalent. However, the mapped IPv4 address can be looked up in the relevant RIR's Whois database.
fc00::/7 Example: fdf8:f53b:82e4::53	**Unique Local Addresses (ULAs)** These addresses are reserved for local use in home and enterprise environments and are not public address space. These addresses might not be unique, and there is no formal address registration. Packets with these addresses in the source or destination fields are not intended to be routed on the public Internet but are intended to be routed within the enterprise or organisation. See RFC 4193 for more details.	Private, or RFC 1918 address space: 10.0.0.0/8 172.16.0.0/12 192.168.0.0/16

Source: www.ripe.net/ipv6-address-types

IPv6: 128-bit address types

fe80::/10 Example: fe80::200:5aee:feaa:20a2	**Link-Local Addresses** These addresses are used on a single link or a non-routed common access network, such as an Ethernet LAN. They do not need to be unique outside of that link. Link-local addresses may appear as the source or destination of an IPv6 packet. Routers must not forward IPv6 packets if the source or destination contains a link-local address.	169.254.0.0/16

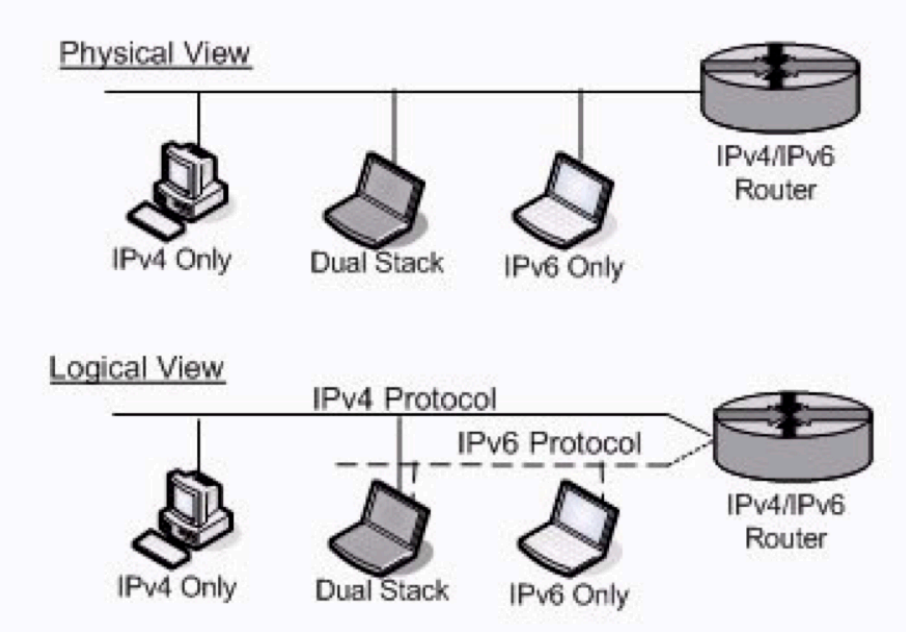
2000::/3	**Global Unicast** Other than the exceptions documented in this table, the operators of networks using these addresses can be found using the Whois servers of the RIRs listed in the registry at: http://www.iana.org/assignments/ipv6-unicast-address-assignments	No equivalent single block
ff00::/8 Example: ff01:0:0:0:0:0:0:2	**Multicast** These addresses are used to identify multicast groups. They should only be used as destination addresses, never as source addresses.	224.0.0.0/4

IPv6 Global Unicast Address Assignments
http://www.iana.org/assignments/ipv6-unicast-address-assignments/ipv6-unicast-address-assignments.xml

Source: www.ripe.net/ipv6-address-types

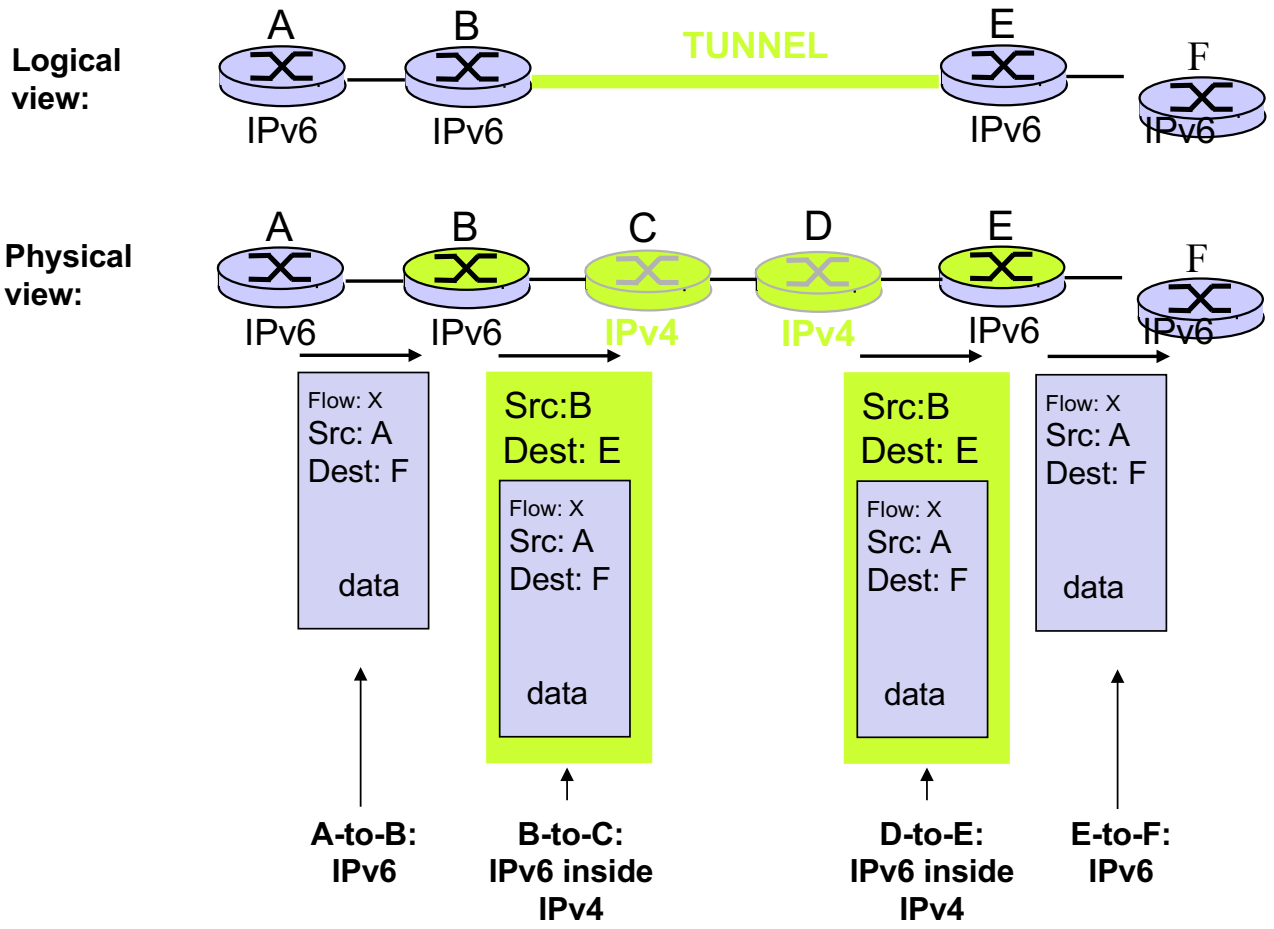
Transition from IPv4 to IPv6 – not all routers can be upgraded simultaneously

- **Dual-stack**
  - IPv6 nodes also have an IPv4 implementation
  - IPv4 and IPv6 over same physical links



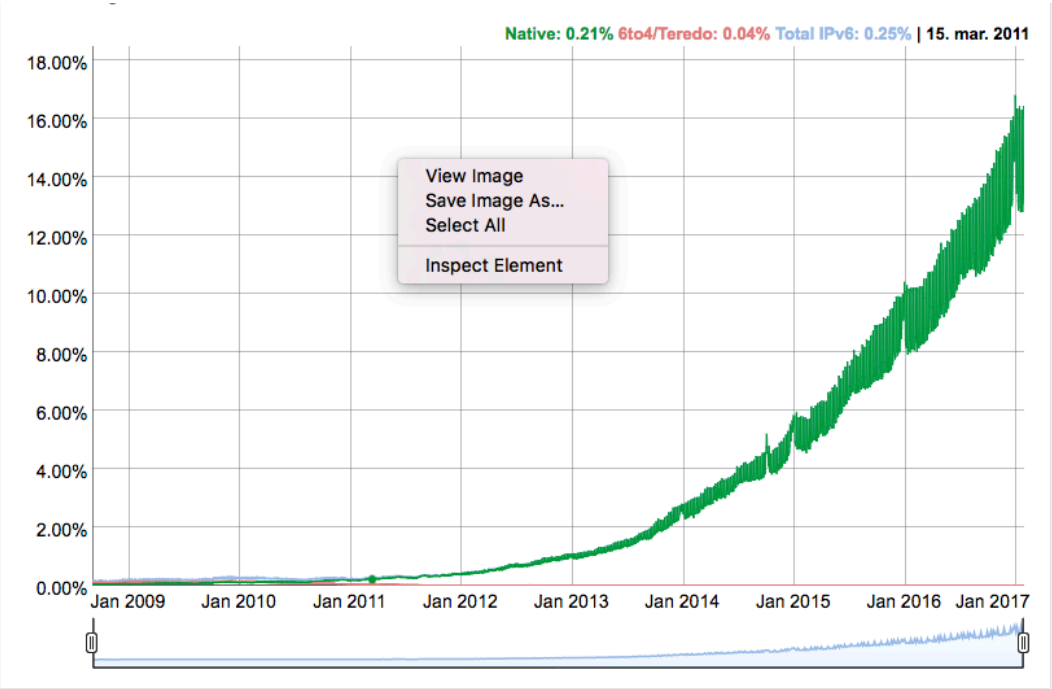
Source: BT, IPv4-to-IPv6 Transition and Co-Existence Strategies

Tunneling to interconnect IPv6 networks

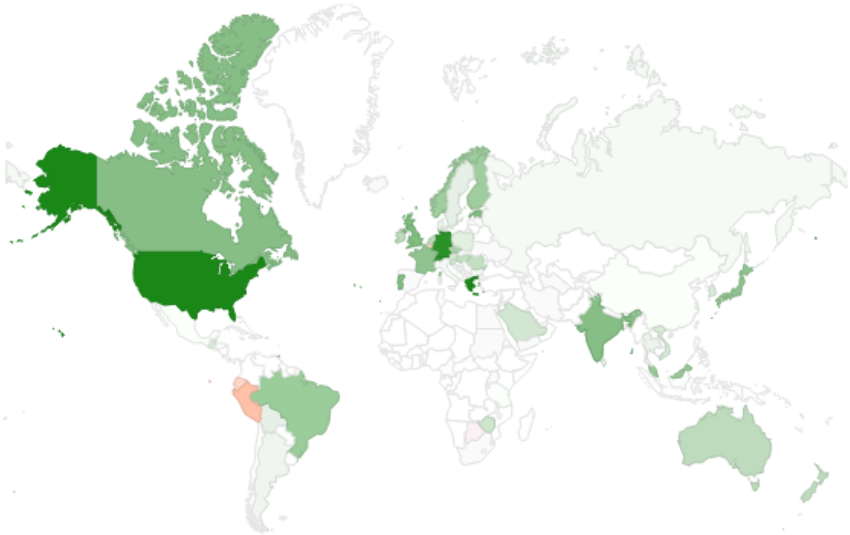


Google statistics on IPv6 adoption

% of users that access Google over IPv6



Per country IPv6 adoption



Source: https://www.google.com/intl/en/ipv6/statistics.html

Network layer – in each and every node and end system

4.1 Introduction

~~4.2 Virtual circuit and datagram~~

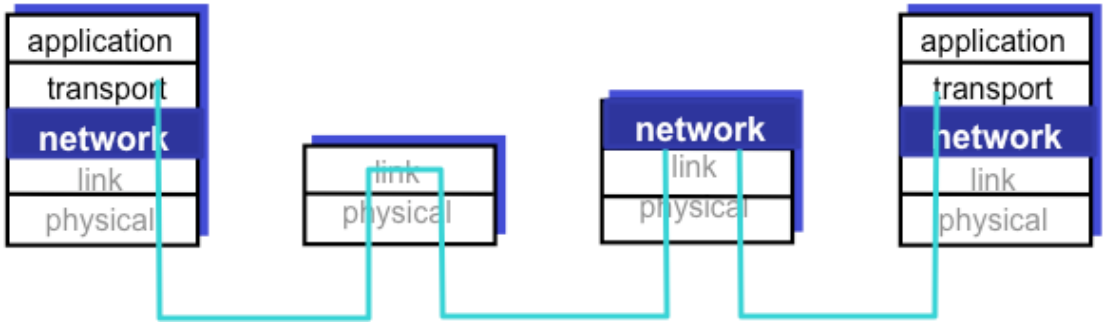
4.3 What's inside a router
- Input processing
- Switching
- Output processing
- Queuing

4.4 IP: Internet Protocol
- Datagram format
- IPv4 addressing
- ICMP
- **IPv6**

4.5 Routing algorithms
- ~~Link state vs Distance vector~~
- ~~Hierarchical routing~~

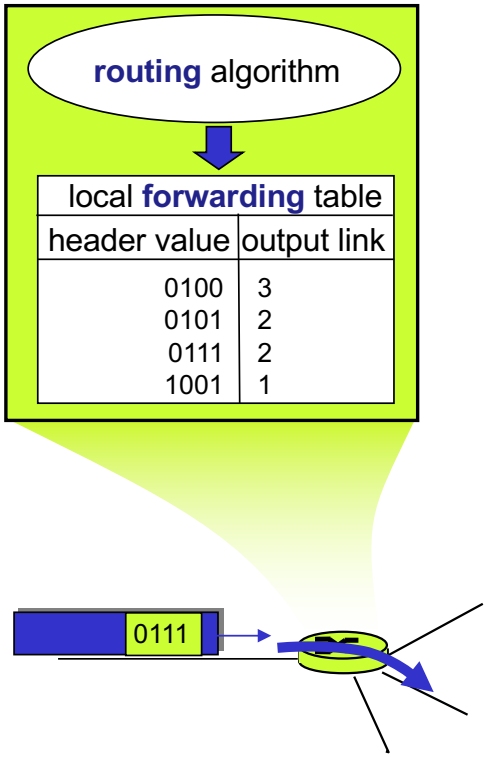
8.7 Network layer security

8.9.1 Firewalls

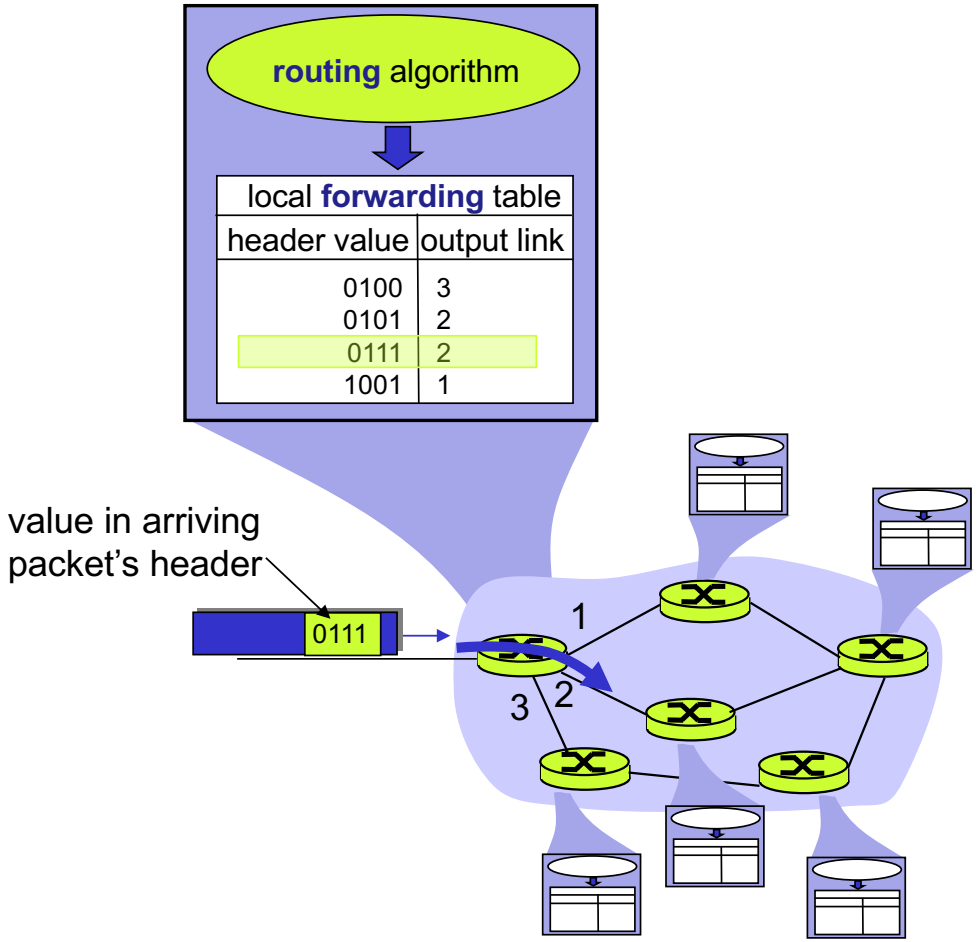


Routing protocols deal with the “shortest path problem”

- **Routing algorithms** for building and managing the routing/forwarding table
  - Must avoid loops
- **Convergence**: How fast do the routers react on topology changes
- **Overhead** of routing traffic: CPU, memory, bandwidth
- Parameters related to choice of route based on specific criteria – the **link cost**
  - Routing hops, bandwidth, delay, utilization, etc
- Routing algorithm **scalability** defines their suitability for large networks

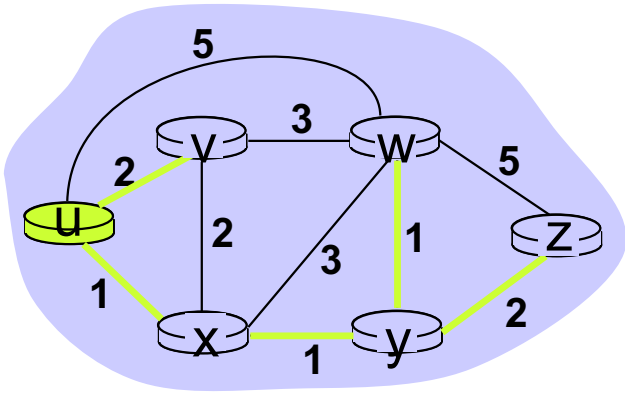


The distributed routing protocol establishes the routing tables in the routers

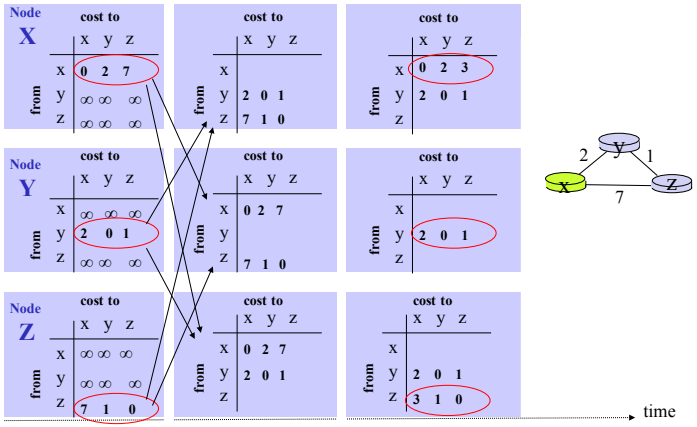


Dynamic routing within a domain: intra-domain routing

1. **"Link state":** each node sends all nodes information on state of directly connected links
   - Dikjstras algorithm
   - Eg. protocol OSPF: Open Shortest Path First

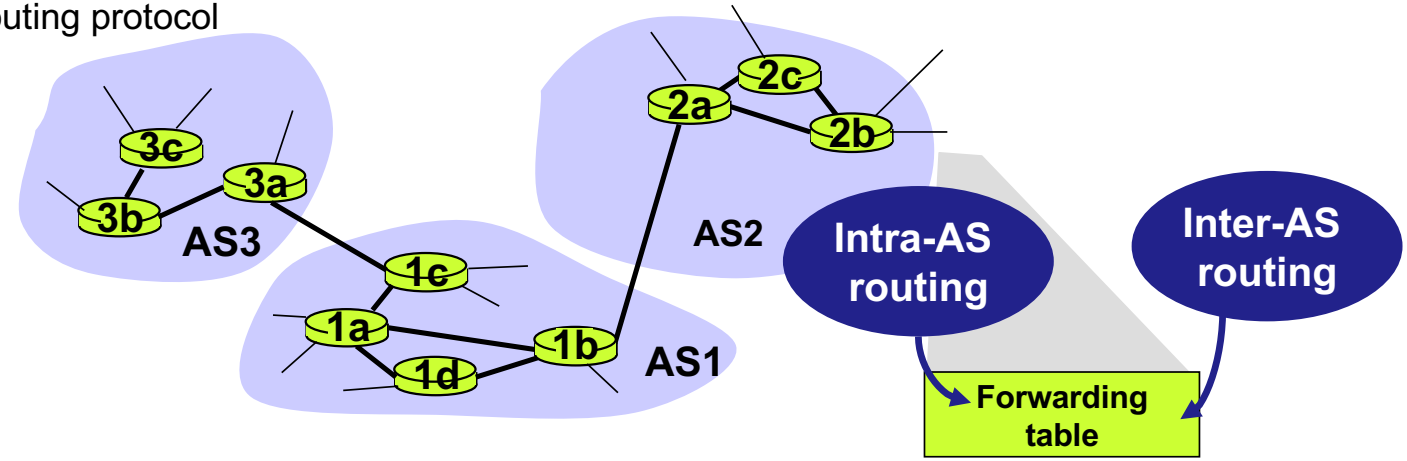


2. **"Distance vector":** each node sends neighbor nodes information about its distances to the other nodes
   - Eg. protocol RIPv2: Route Information Protocol



Hierarchical routing aggregates domains = “autonomous systems” (AS)

- **Routers in same AS** run same routing protocol
  - **“Intra-AS”** routing protocol
  - Routers in different AS´s can run different intra-AS routing protocols
- **Gateway router** has a direct link to router in another AS
  - **“Inter-AS”** routing protocol
- Forwarding table configured by both intra- and inter-AS routing algorithm
  - intra-AS sets entries for internal destinations
  - inter-AS & intra-AS sets entries for external destinations



Network layer: Roadmap

4.1 Introduction

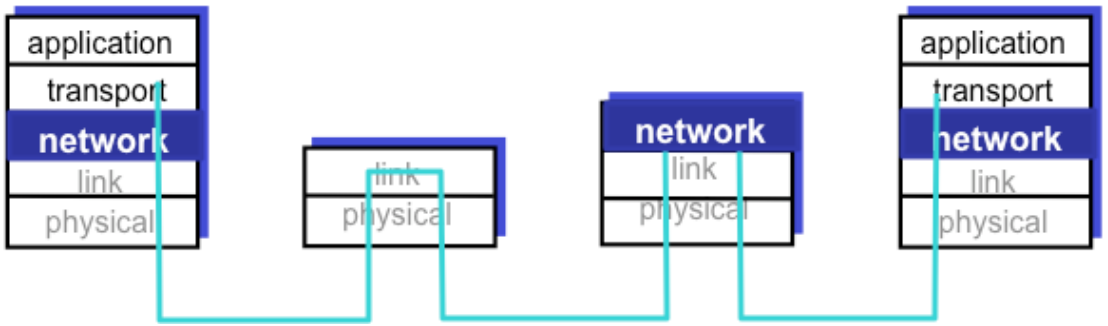
4.3 What's inside a router
- Input processing
- Switching
- Output processing
- Queuing

4.4 IP: Internet Protocol
- Datagram format
- IPv4 addressing
- ICMP
- IPv6

4.5 Routing algorithms
- ~~Link state vs Distance vector~~
- ~~Hierarchical routing~~

8.7.1 Network layer security
- **IPSEC (IP security)**
- **VPN (virtual private networks)**

8.9.1 Firewalls



NTNU

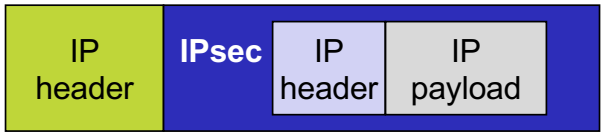
Virtual Private Networks (VPNs) to secure network and reduce cost

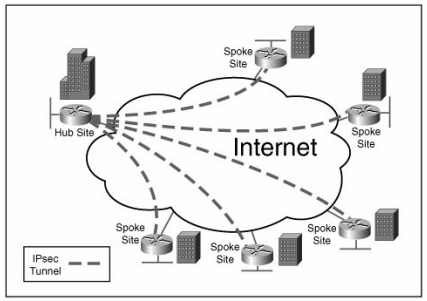
- A virtual private network (VPN) is a private data network that makes use of the public telecommunication infrastructure, maintaining privacy through the use of a tunneling protocol and security procedures
  - Virtual - connectivity deployed on a shared infrastructure
    - reduce cost for separate routers, links, DNS infrastructure
  - Private - for the user the networks looks like a private network with the same policies and performance as a private network
    - possibly encrypted before entering public Internet
    - logically separate from other traffic - tunnels
  - Communication to/from outside the VPN is restricted
    - access control, firewalls, tunnels

VPN (virtual private network) tunnels can be provided by IPSEC

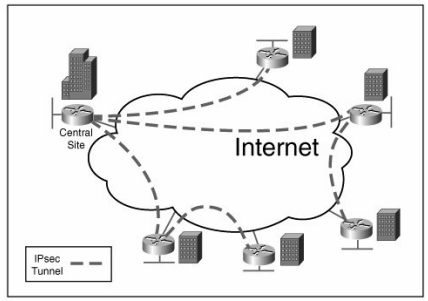
IPSEC provides services between hosts and routers by adding an IPSEC header

- IPSEC peers can be two end systems, two routers/firewalls, or a router/firewall and an end system
- Routers in the internet only see the outermost ordinary IP header
- IPSEC protects the IP payload (upper layer protocols, e.g. ICMP, routing messages, UDP, TCP)
- Two modes: transport and tunneling
  - Tunneling mode more widely used for IPVPN

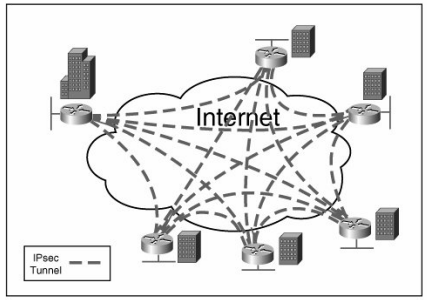




Hub-and-Spoke



Partial-Mesh

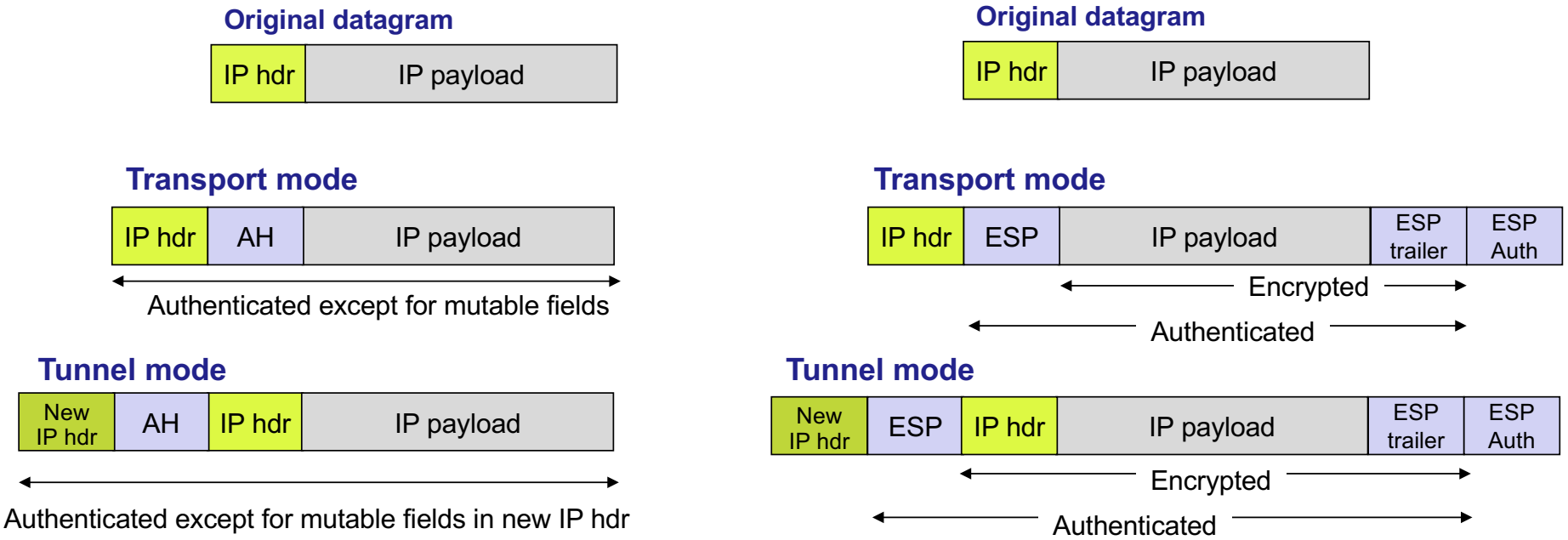


Full-Mesh

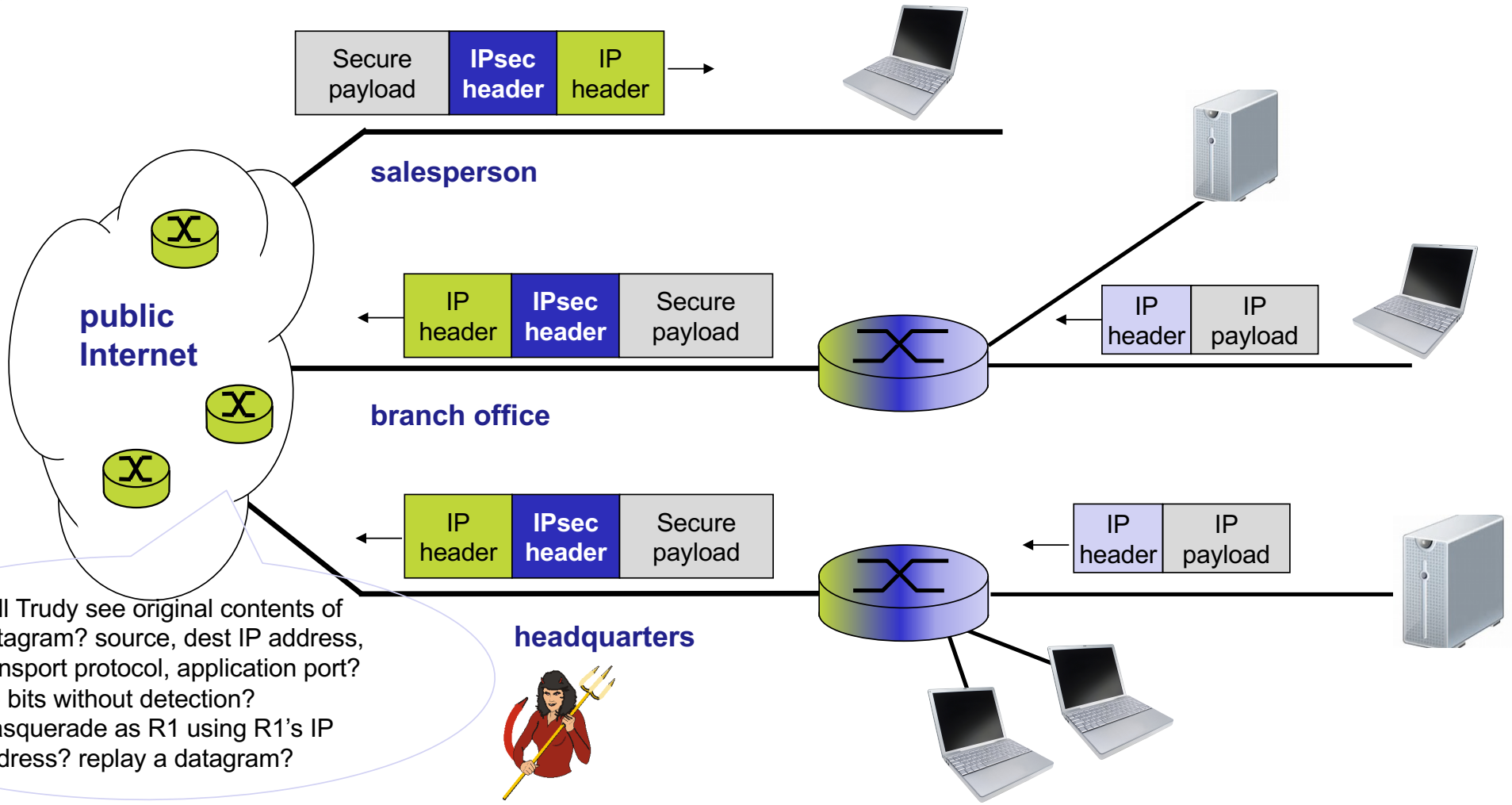
IPSEC adds headers and trailers to original IP datagram

IPSEC Authentication Header (AH) provides
- Data integrity
- Source authentication

IPSEC Encapsulation Security Payload (ESP) also provides
- Confidentiality: All data sent from one entity (host or router) to other is encrypted



Virtual Private Network tunnels between branches at different geographic sites



Will Trudy see original contents of datagram? source, dest IP address, transport protocol, application port? flip bits without detection? masquerade as R1 using R1's IP address? replay a datagram?

Network layer: Roadmap

4.1 Introduction

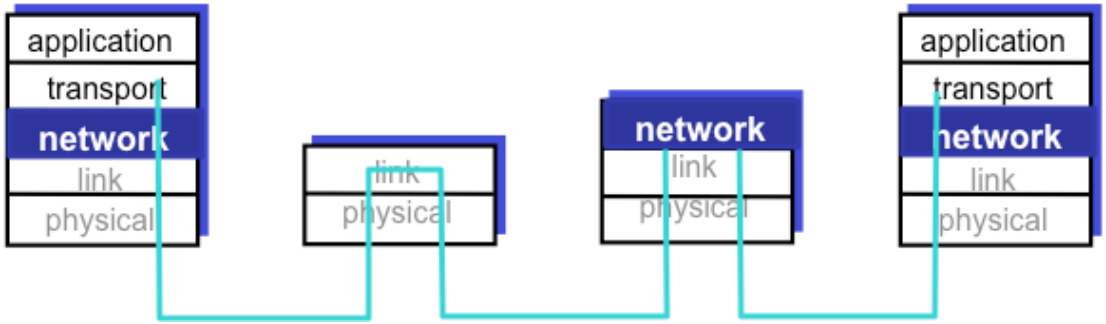
4.3 What's inside a router
- Input processing
- Switching
- Output processing
- Queuing

4.4 IP: Internet Protocol
- Datagram format
- IPv4 addressing
- ICMP
- IPv6

4.5 Routing algorithms
- ~~Link state vs Distance vector~~
- ~~Hierarchical routing~~

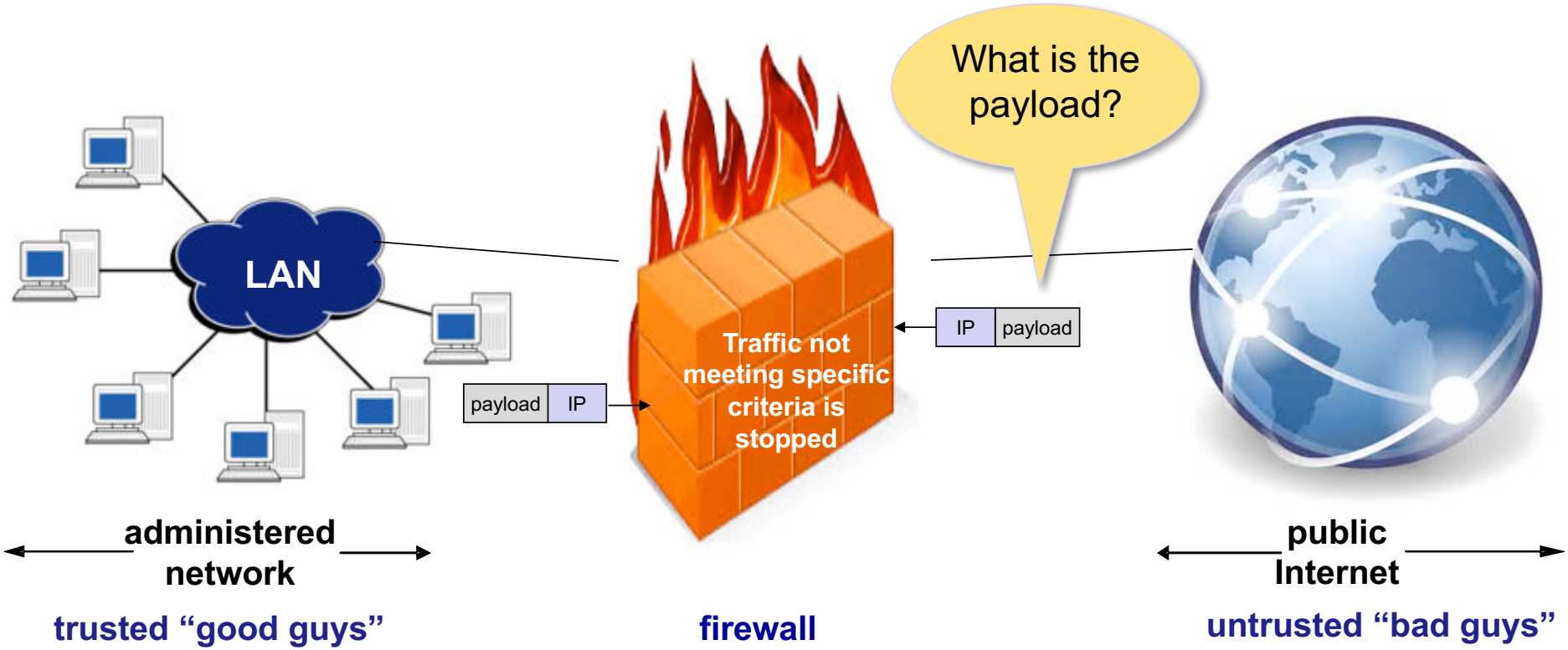
8.7.1 Network layer security
- IPSEC (IP security)
- VPN (virtual private networks)

8.9.1 Firewalls



Firewalls isolates organization's internal network from larger Internet

allowing some packets to pass, blocking others

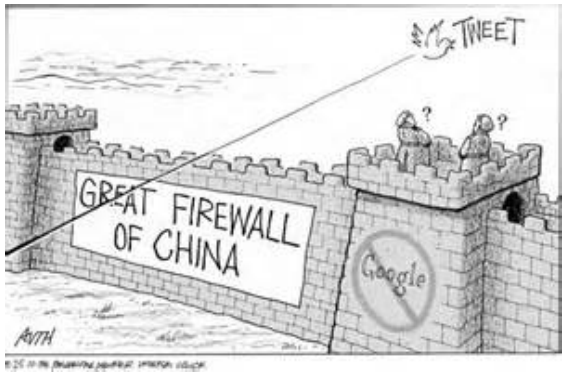


Firewalls: why do we need them?

- Prevent denial of service attacks
  - SYN flooding: attacker establishes many bogus TCP connections, no resources left for “real” connections
- Prevent illegal modification/access of internal data
  - e.g. attacker replaces homepage with something else
- Allow only authorized access to resources within the network
  - there is an identified set of authenticated users/hosts

Three types of firewalls

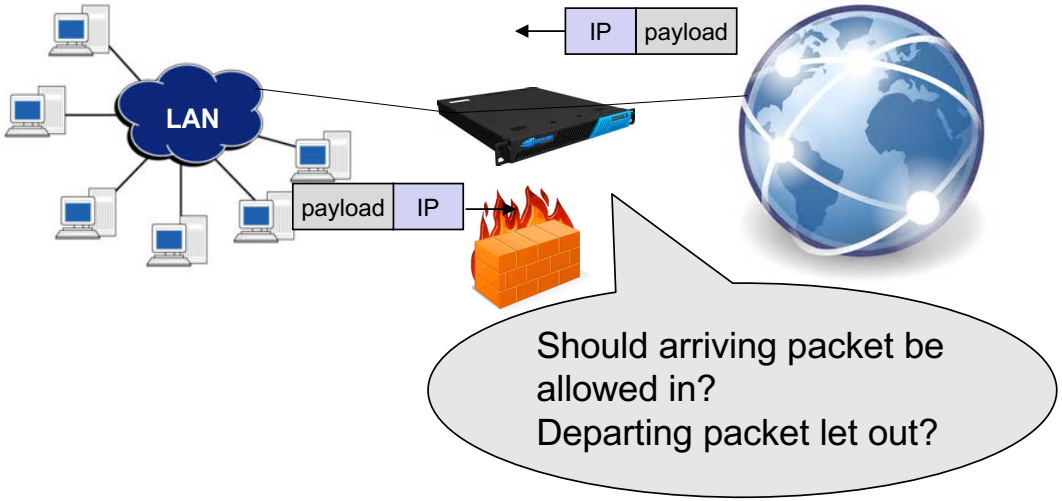
- stateless packet filters
- stateful packet filters
- application gateways



Three types of firewalls

Stateless packet filtering

- Router filters **packet-by-packet**
  - Different rules for datagrams leaving and entering the network
  - Different rules for the different router interfaces
- Decision to forward/drop packet based on:
  - source IP address, destination IP address
  - TCP/UDP source and destination port
  - ICMP message type
  - TCP SYN and ACK bits
- example 1: block incoming and outgoing datagrams with source or dest port = 23 (telnet)
  - all incoming, outgoing telnet connections blocked



- example 2: block inbound TCP segments with ACK=0
  - prevents external clients from making TCP connections with internal clients, but allows internal clients to connect to outside

NTNU

Firewall is configured according to security policy

Policy	Firewall Setting
No outside Web access	Drop all outgoing packets to any IP address, port 80
No incoming TCP connections, except those for institution's public Web server only	Drop all incoming TCP SYN packets to any IP except 130.207.244.203, port 80
Prevent Web-radios from eating up the available bandwidth	Drop all incoming UDP packets - except DNS and router broadcasts.
Prevent your network from being used for a smurf DoS attack	Drop all ICMP packets going to a "broadcast" address (e.g. 130.207.255.255).
Prevent your network from being tracerouted	Drop all outgoing ICMP TTL expired traffic

Access Control Lists (ACL) implement the firewall rules

- ACL: table of rules, applied top to bottom to incoming packets: (action, condition) pairs on a router interface

action	source address	dest address	protocol	source port	dest port	flag bit
allow	222.22/16	outside of 222.22/16	TCP	> 1023	80	any
allow	outside of 222.22/16	222.22/16	TCP	80	> 1023	ACK
allow	222.22/16	outside of 222.22/16	UDP	> 1023	53	---
allow	outside of 222.22/16	222.22/16	UDP	53	> 1023	----
deny	all	all	all	all	all	all

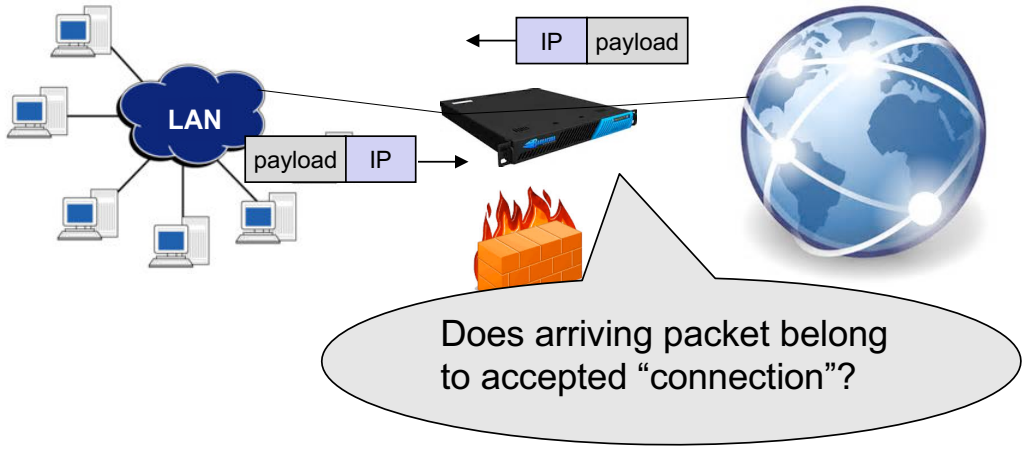
Which traffic is let through?

All segments relevant?

Three types of firewalls

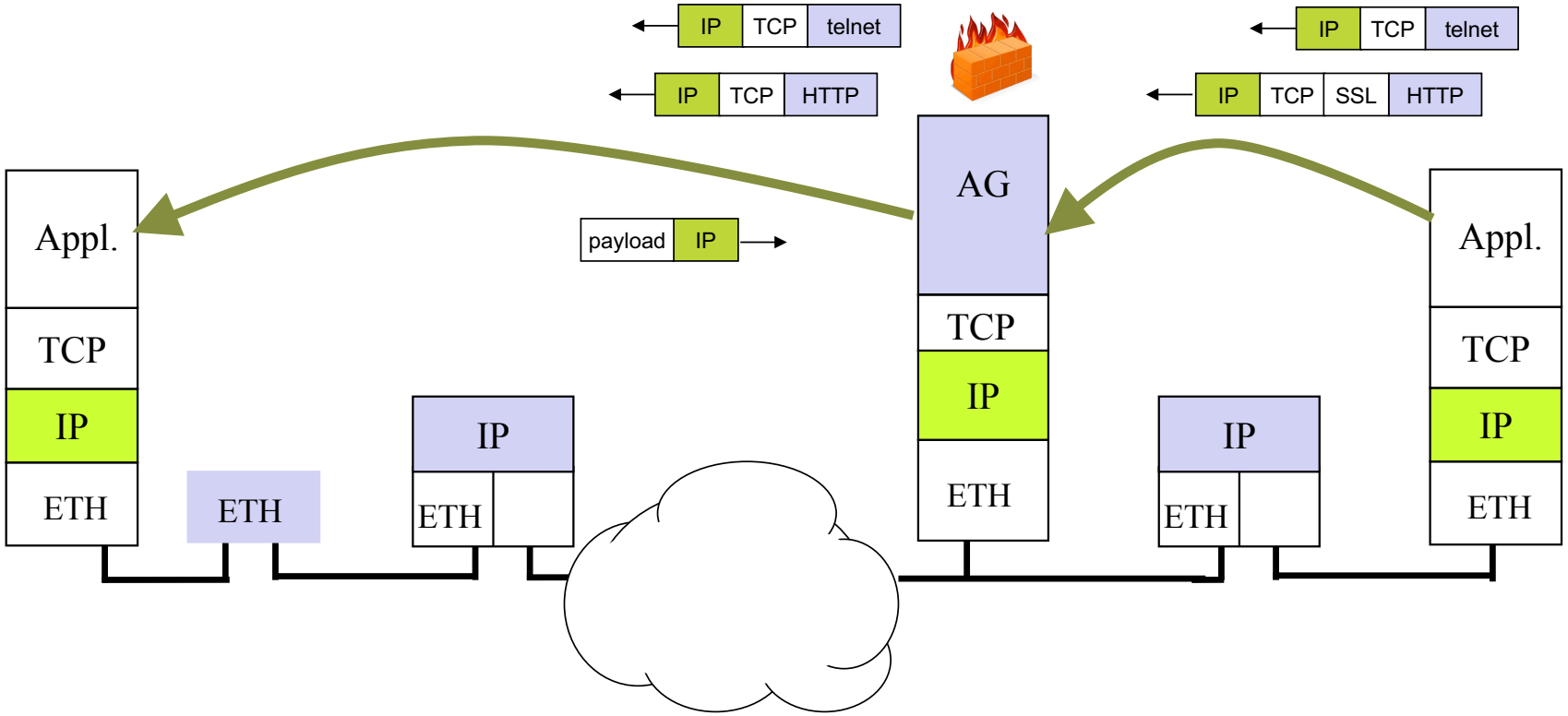
Stateful packet filtering holds track of “connections”

- ACL (access control lists) augmented to indicate need to check connection state table before admitting packet
  - Track status of every TCP connection: connection setup (SYN), teardown (FIN): incoming, outgoing packets “making sense”?
  - UDP “connections”
    - DNS guard allows only one DNS response to a DNS reply (match trans-id)
    - Matching ICMP echo request/reply sequence number
    - UDP timeout on inactive “connections”



action	source address	dest address	proto	source port	dest port	flag bit	check conxion
allow	222.22/16	outside of 222.22/16	TCP	> 1023	80	any	
allow	outside of 222.22/16	222.22/16	TCP	80	> 1023	ACK	X
allow	222.22/16	outside of 222.22/16	UDP	> 1023	53	---	
allow	outside of 222.22/16	222.22/16	UDP	53	> 1023	----	X
deny	all	all	all	all	all	all	

Application gateways filter packets on application data as well as on IP/TCP/UDP fields

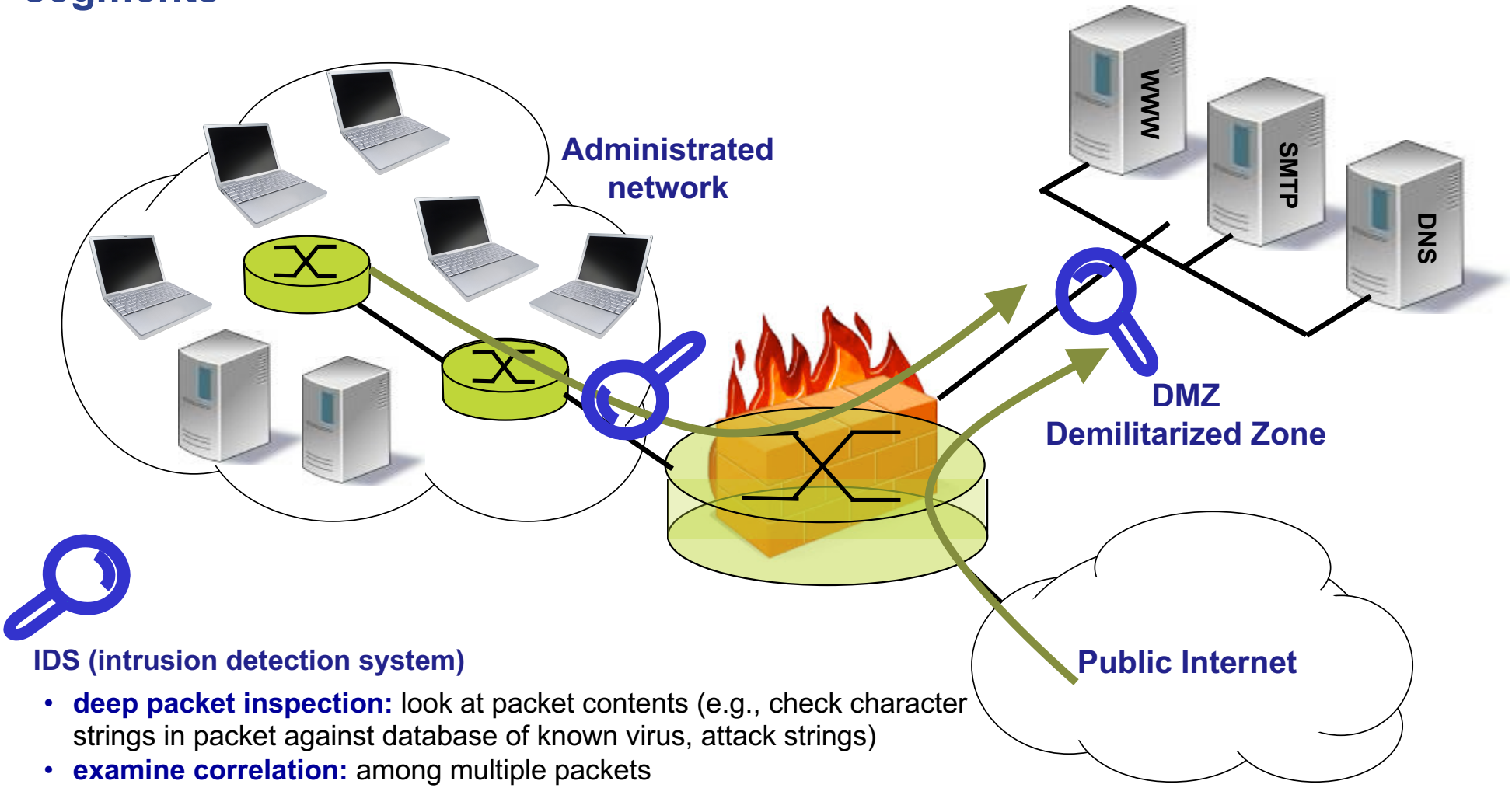


NTNU

Limitations of firewalls, gateways

- A firewall can not protect against
  - malicious insiders
  - connections that circumvent it
  - completely new threats
  - all websites with malicious code
  - poor decisions or a too relaxed security policy
  - the administrator that does not correctly set it up
  - a bad password policy or misuse of passwords.
- IP spoofing: router can't know if data "really" comes from claimed source
- Client software must know how to contact application gateways
  - e.g., must set IP address of proxy in Web browser
- Tradeoff: degree of communication with outside world, level of security
  - Filters often use all or nothing policy for UDP

Increased security through physical isolation of trusted and untrusted segments



IDS (intrusion detection system)

- **deep packet inspection:** look at packet contents (e.g., check character strings in packet against database of known virus, attack strings)
- **examine correlation:** among multiple packets

Summary network layer

1. **Addressing**
2. **Fragmentation and reassembly**
3. **Routing and forwarding**
4. **Network interconnect**

Routing: determine route taken by packets from source to destination

Forwarding: move packets from router's input to appropriate router output

- Longest prefix match

Internet **datagram** model is connectionless

IPdst address is used by routers to forward datagrams