



Computer Networking

Network Layer - IP

Prof. Andrzej Duda
duda@imag.fr

<http://duda.imag.fr>

1

The transport layer relies on the services of the network layer, which provides a communication service between hosts. In particular, the network layer moves transport-layer segments from one host to another. At the sending host, the transport-layer segment is passed to the network layer. It is then the job of the network layer to get the segment to the destination host and pass the segment up the protocol stack to the transport layer.

Network Layer

Chapter goals:

- understand principles behind network layer services:
 - virtual circuits vs. datagrams
 - addressing
 - packet forwarding
- instantiation and implementation in the Internet

Overview:

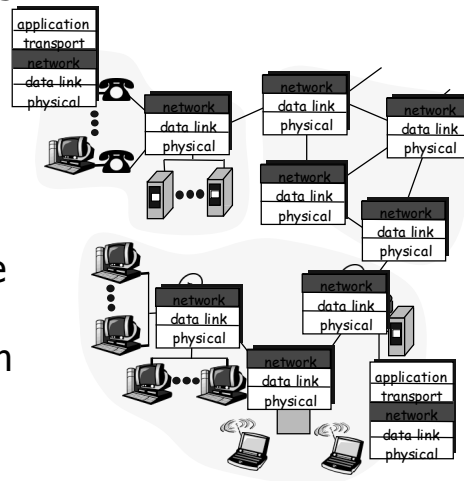
- network layer services
- IP addresses
- packet forwarding principles
- details of IP
- overview of DHCP, ICMP, ARP
- IPsec and VPN

Network layer functions

- transport packet from sending to receiving hosts
- network layer protocols in *every* host, router

three important functions:

- *path determination*: route taken by packets from source to dest. *Routing algorithms*
- *switching*: move packets from router's input to appropriate router output
- *call setup*: some network architectures require router call setup along path before data flows



3

The network layer *involves each and every host and router in the network*. The role of the network layer in a sending host is to begin the packet on its journey to the receiving host. The role of the network layer is thus deceptively simple--to transport packets from a sending host to a receiving host. To do so, three important network-layer functions can be identified:

- *Path determination*. The network layer must determine the route or path taken by packets as they flow from a sender to a receiver. The algorithms that calculate these paths are referred to as **routing algorithms**.
- *Switching*. When a packet arrives at the input to a router, the router must move it to the appropriate output link.
- *Call setup*. With TCP, a three-way handshake is required before data actually flow from sender to receiver. This allowed the sender and receiver to set up the needed state information (for example, sequence number and initial flow-control window size). In an analogous manner, some network-layer architectures (for example, ATM) require that the routers along the chosen path from source to destination handshake with each other in order to setup state before data actually begins to flow. In the network layer, this process is referred to as **call setup**. The network layer of the Internet architecture does not perform any such call setup.

Network service model

- The *network service model* defines edge-to-edge channel
- The most important abstraction provided by network layer:
 - **network-layer connection-oriented service:** virtual circuit (X.25, Frame Relay, ATM, MPLS)
 - **network-layer connectionless service:** datagram (IP, IPX)

4

The **network-service model** defines the characteristics of end-to-end transport of data between one "edge" of the network and the other, that is, between sending and receiving end systems. The most important abstraction that the network layer provides to the transport layer is whether the network layer uses a **connection-oriented service (virtual circuits)** or a **connectionless service (datagram)** model.

With a **virtual circuits layer** a circuit management (setup, data-transfer, teardown) and signaling are needed.

With a **datagram network layer**, each time an end system wants to send a packet, it stamps the packet with the address of the destination end system, and then pops the packet into the network

Virtual circuits

“source-to-dest path behaves much like telephone circuit”

- performance-wise
 - network actions along source-to-dest path
- call setup, teardown for each call *before* data can flow
 - each packet carries VC identifier (not destination host ID)
 - *every* router on source-dest path maintains “state” for each passing connection
 - transport-layer connection only involved two end systems
 - link, router resources (bandwidth, buffers) may be *allocated* to VC
 - to get circuit-like performance

5

There are three identifiable phases in a virtual circuit:

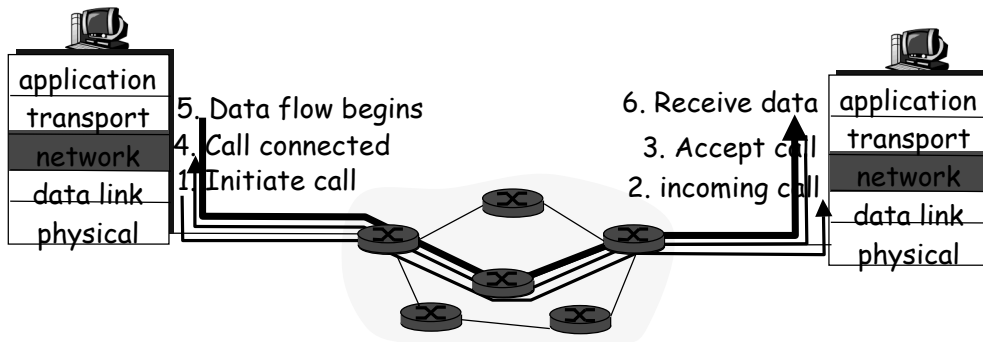
• *VC setup*. During the setup phase, the sender contacts the network layer, specifies the receiver address, and waits for the network to set up the VC. The network layer determines the path between sender and receiver, that is, the series of links and packet switches through which all packets of the VC will travel. This typically involves updating tables in each of the packet switches in the path. During VC setup, the network layer may also reserve resources (for example, bandwidth) along the path of the VC.

• *Data transfer*. Once the VC has been established, data can begin to flow along the VC. Each packet knows the VC (carries the VC identifier), and every switch on the VC path is aware of the existence of the VC itself.

• *Virtual-circuit teardown*. This is initiated when the sender (or receiver) informs the network layer of its desire to terminate the VC. The network layer will then typically inform the end system on the other side of the network of the call termination and update the tables in each of the packet switches on the path to indicate that the VC no longer exists.

Virtual circuits: signaling protocols

- used to setup, maintain teardown VC
- used in ATM, Frame-Relay, X.25
- not used in today's Internet
 - but MPLS at 2.5 (between Link and Network Layer)

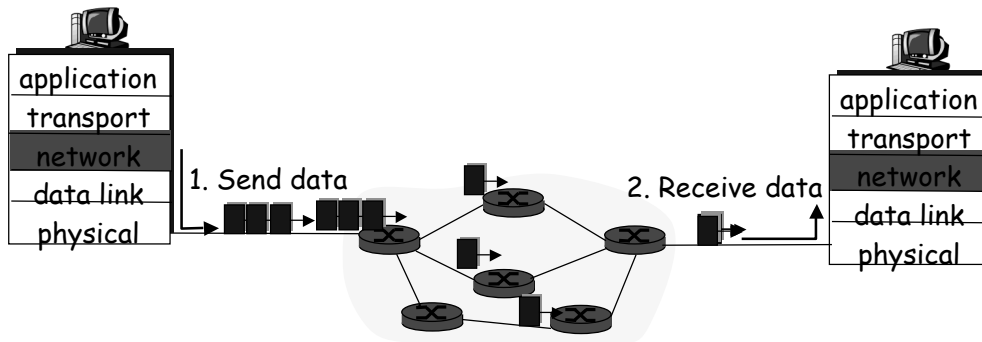


6

The messages that the end systems send to the network to indicate the initiation or termination of a VC, and the messages passed between the switches to set up the VC (that is, to modify switch tables) are known as **signaling messages** and the protocols used to exchange these messages are often referred to as **signaling protocols**. In the Internet virtual circuits is not used at network layer, while ATM, frame relay and X.25, are three other networking technologies that use virtual circuit.

Datagram networks: the Internet model

- no call setup at network layer
- routers: no state about end-to-end connections
 - no network-level concept of "connection"
- packets typically routed using destination host ID
 - packets between same source-dest pair may take different paths



7

Datagram routing is similar to routing ordinary postal mail: packet switches route a packet toward its destination by examining the packet's destination address, indexing a routing table with the destination address, and forwarding the packet in the direction of the destination. There is no "connection management" as well as no tables or state are needed inside the network. Different packets to the same destination can be routed via a different route. The current Internet architecture provides only one service model, the datagram service, which is also known as "**best-effort service.**" From the table, it might appear that best effort service is a euphemism for "no service at all." With best-effort service, timing between packets is not guaranteed to be preserved, packets are not guaranteed to be received in the order in which they were sent, nor is the eventual delivery of transmitted packets guaranteed. Given this definition, a network that delivered *no* packets to the destination would satisfy the definition of best-effort delivery service. However, there are sound reasons for such a minimalist network service model:

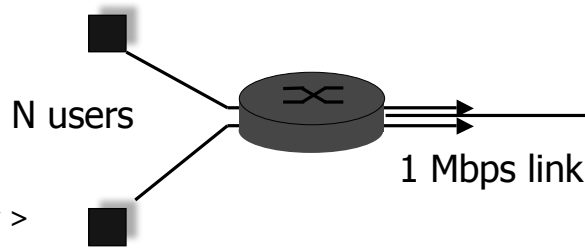
- it is easier to *interconnect* networks that used very different link-layer technologies
- it is easier to *add a new service* simply by attaching a host to the network and defining a new higher-layer protocol

The Internet's best-effort only service model is currently being extended to include so-called integrated services and differentiated service.

Packet switching vs. Circuit switching

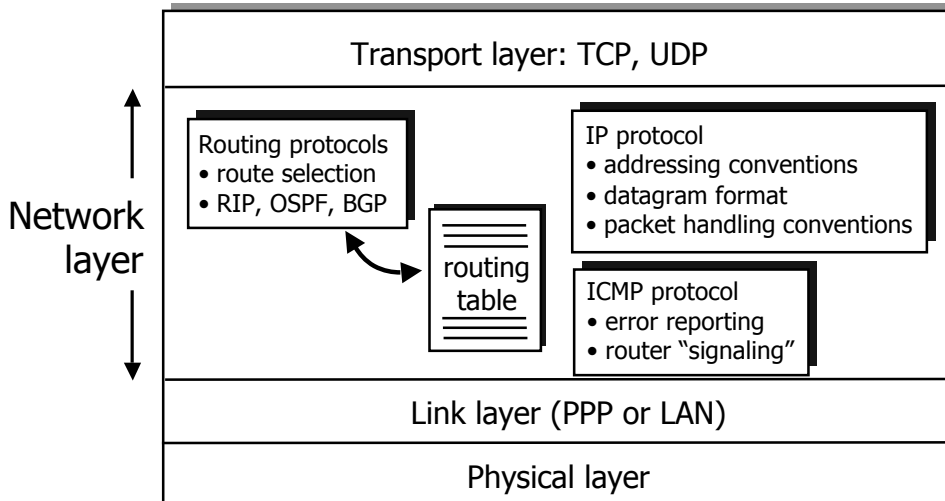
Packet switching allows more users to use network!

- Eg. 1 Mbit link
- each user:
 - 100 Kb/s when "active"
 - active 10% of time
- circuit-switching:
 - 10 users
- packet switching:
 - with 35 users, probability > 10 active less than .004



The Internet Network layer

Host, router network layer functions:



9

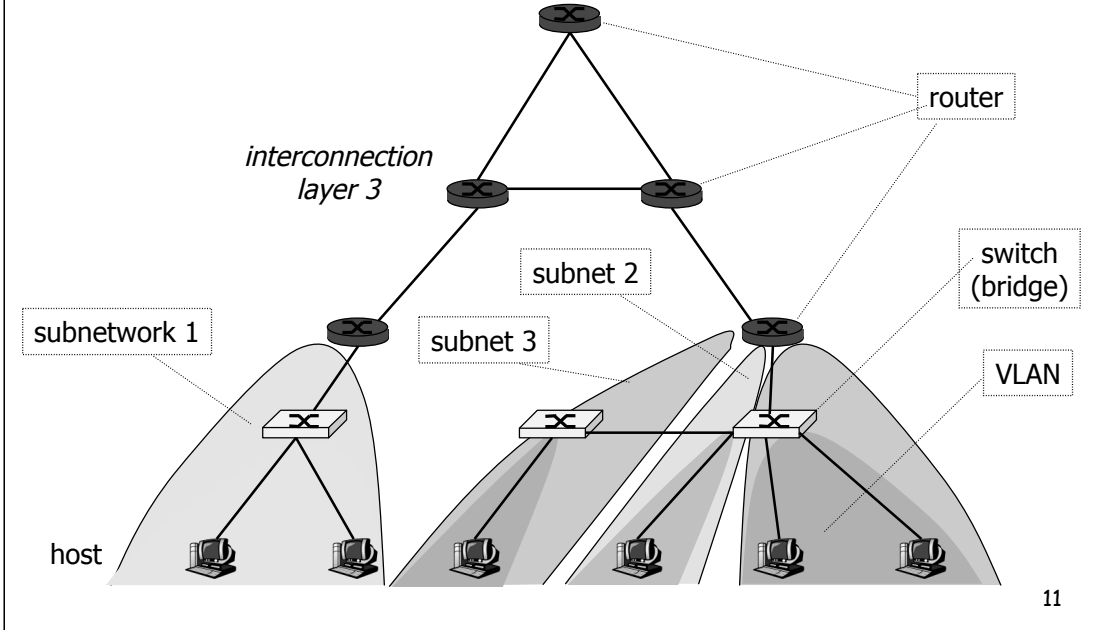
The pieces of the network layer of the Internet are often collectively referred to as the IP layer (named after the Internet's IP protocol). We'll see, though, that the IP protocol itself is just one piece (albeit a very important piece) of the Internet's network layer. The Internet's network layer provides connectionless datagram service rather than virtual-circuit service. When the network layer at the sending host receives a segment from the transport layer, it encapsulates the segment within an IP datagram, writes the destination host address as well as other fields in the datagram, and sends the datagram to the first router on the path toward the destination host. The Internet's network layer has three major components:

- The Internet Protocol, or more commonly, the **IP Protocol**, which defines network-layer addressing, the fields in the datagram (that is, the network-layer PDU), and the actions taken by routers and end systems on a datagram based on the values in these fields. There are two versions of the IP protocol in use today: **IPv4** [[RFC 791](#)] and **IPv6** [[RFC 2373](#); [RFC 2460](#)], which has been proposed to replace IPv4 in upcoming years.
- The second major component of the network layer is the path determination component; it determines the route a datagram follows from source to destination. Examples of such components used in the Internet are RIP, OSPF, BGP.
- The Internet's network-layer error and information reporting protocol, **ICMP**, is a facility to report errors in datagrams and respond to requests for certain network-layer information..

IP principles

- Elements
 - **host** = end system; **router** = intermediate system;
subnetwork = a collection of hosts that can communicate directly without routers
- Routers are between subnetworks only:
 - a subnetwork = a collection of systems with a common prefix
- Packet forwarding
 - **direct**: inside a subnetwork hosts communicate directly without routers, router delivers packets to hosts
 - **indirect**: between subnetworks one or several routers are used
- Host either sends a packet to the destination using its LAN, or it passes it to the router for forwarding

Interconnection structure - layer 3



Interconnection at layer 3

- Routers
 - interconnect subnetworks
 - logically separate groups of hosts
 - managed by one entity
- Forwarding based on IP address
 - structured address space
 - routing tables: aggregation of entries
 - works if no loops - routing protocols
 - scalable inside one administrative domain

Internet and intranet

- An intranet
 - a collection of end and intermediate systems interconnected using the TCP/IP architecture*
 - normally inside one organization*
- The Internet
 - the global collection of all hosts and routers interconnected using the TCP/IP architecture*
 - coordinated allocation of addresses and implementation requirements by the Internet Society*
- Intranets are often connected to the Internet by firewalls
 - routers that act as protocol gateways (address and port translation, application level relay)

13

- an internet can use its own addresses

- Internet addresses are managed worldwide

There is no global Internet organization: like for telephony, the Internet service is provided by a collection of competing Internet Service Providers (ISPs)

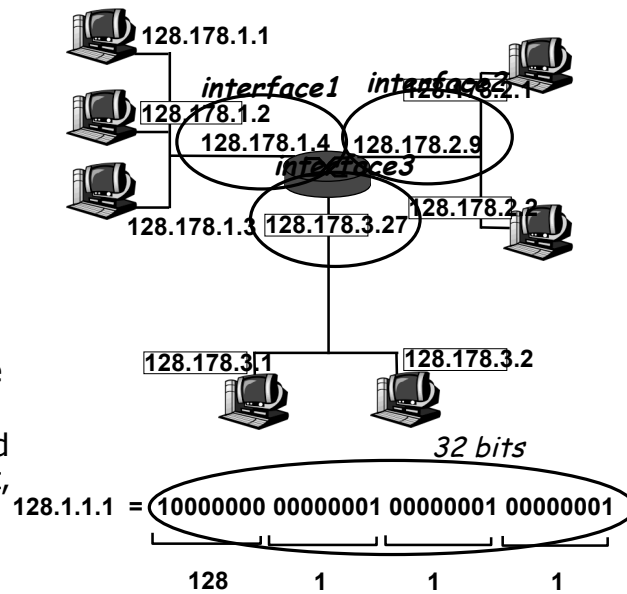
Only addresses and standards are managed world-wide.

IP addresses

- Unique addresses in the world, decentralized allocation
- An IP address is 32 bits, noted in dotted decimal notation: 192 . 78 . 32 . 2
- An IP address has a prefix and a host part:
 - `prefix:host`
- Two ways of specifying prefix
 - subnet mask identifies the prefix by bitwise & operation
 - CIDR: bit length of the prefix
- Prefix identifies a subnetwork
 - used for locating a subnetwork - routing

IP Addressing: introduction

- IP address: 32-bit identifier for host, router *interface*
- *interface*: connection between host, router and physical link
 - router's typically have multiple interfaces
 - host may have multiple interfaces
 - IP addresses associated with interface, not host, router



15

An IP address is technically associated with an interface, rather than with the host or router containing that interface. When IP in the host wants to send a datagram, it will do so over this link. The boundary between the host and the physical link is called an **interface**. A router's job is to receive a datagram on an "incoming" link and forward the datagram on some "outgoing" link, thus a router has multiple interfaces, one for each of its links.

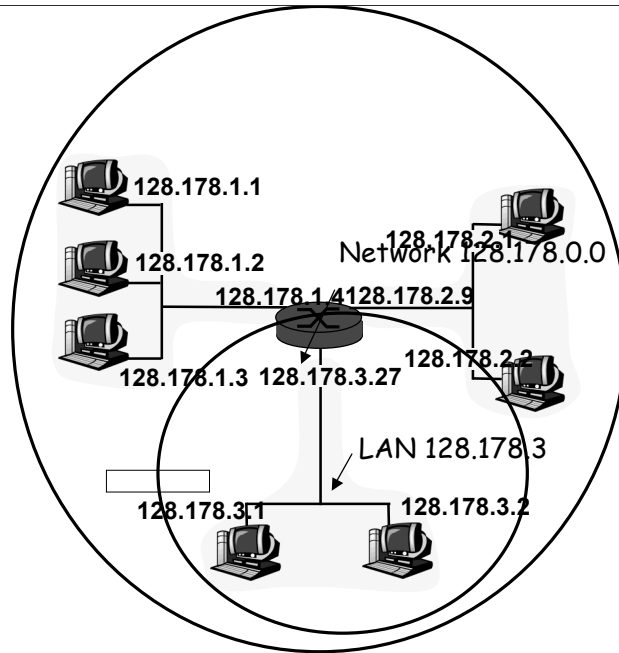
Each IP address is 32 bits long (equivalently, four bytes), and there are thus a total of 2^{32} possible IP addresses. These addresses are typically written in so-called **dotted-decimal notation**, in which each byte of the address is written in its decimal form and is separated by a period ("dot") from other bytes in the address. For example, consider the IP address 193.32.216.9. The 193 is the decimal equivalent of the first eight bits of the address; the 32 is the decimal equivalent of the second eight bits of the address, and so on. Thus, the address 193.32.216.9 in binary notation is:

11000001 00100000 11011000 00001001.

Each interface on every host and router in the global Internet must have an IP address that is globally unique.

IP Addressing

- IP address:
 - network (or prefix) part (high order bits)
 - host part (low order bits)
- *What's a subnetwork?* (from IP address perspective)
 - device interfaces with same network part of IP address
 - can physically reach each other without intervening router



network consisting of 3 IP networks
(for IP addresses starting with 128,
first 24 bits are network address)

16

IP addresses cannot be chosen in a willy-nilly manner, however. In primis, an interface's IP address will be determined by the "network" to which it is connected. The three hosts in the upper-left portion are on the same "IP network" identified by the initial part of their address 128.178.1, and the router interface to which they are connected all have an IP address of the form 128.178.1.xxx. That is, they share a common leftmost 24 bits of their IP address. The 24 address bits that they share in common constitute the network portion of their IP address; the remaining eight bits are the host portion of the IP address. The network itself also has an address: 128.178.1.0/24, where the "/24" notation, sometimes known as a **network mask**, indicates that the leftmost 24 bits of the 32-bit quantity define the network. Sometimes, in the network mask the numbers are substituted with 255. Examples:

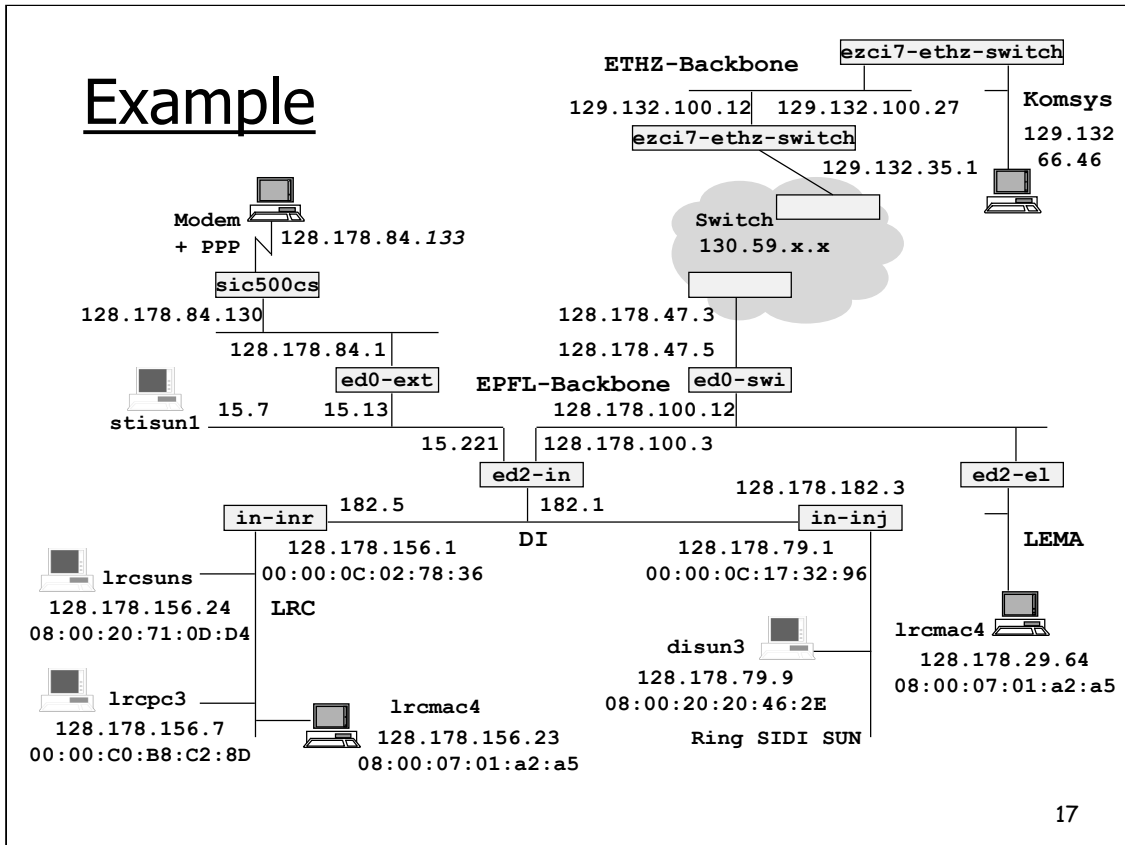
subnet mask at EPFL = **255.255.255.0**

What are the net:subnet and host parts of : **lrcsuns.lrc.epfl.ch** ?

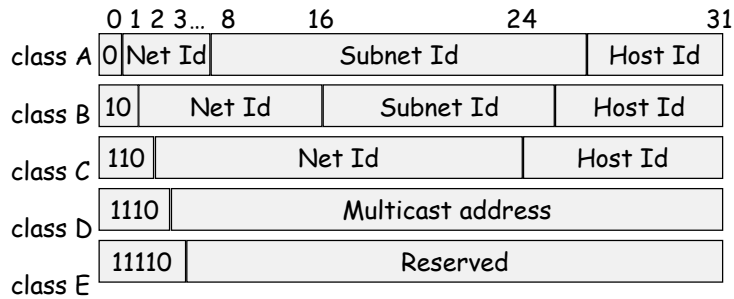
The address is **128.178.156.24**

the prefix is **128.178.156.0**

Example



IP Address Classes



Examples: 128.178.x.x = EPFL host; 129.132.x.x = ETHZ host
 9.x.x.x = IBM host 18.x.x.x = MIT host

| <i>Class</i> | <i>Range</i> |
|--------------|------------------------------|
| A | 0.0.0.0 to 127.255.255.255 |
| B | 128.0.0.0 to 191.255.255.255 |
| C | 192.0.0.0 to 223.255.255.255 |
| D | 224.0.0.0 to 239.255.255.255 |
| E | 240.0.0.0 to 247.255.255.255 |

- Class B addresses are close to exhausted; new addresses are taken from class C, allocated as continuous blocks

18

At the origin, the prefix of an IP address was defined in a very rigid way. For class A addresses, the prefix was 8 bits. For class B, 16 bits. For class C, 24 bits. The interest of that scheme was that by simply analyzing the address you could find out what the prefix was.

The requirement that the network portion of an IP address be exactly one, two, or three bytes long turned out to be problematic for supporting the rapidly growing number of organizations with small and medium-sized networks. A class C (/24) network could only accommodate up to $2^8 - 2 = 254$ hosts (two of the $2^8 = 256$ addresses are reserved for special use)--too small for many organizations. However, a class B (/16) network, which supports up to 65,534 hosts was too large. Under classful addressing, an organization with, say, 2,000 hosts was typically allocated a class B (/16) network address. This led to a rapid depletion of the class B address space and poor utilization of the assigned address space. It was soon recognized that this form was too rigid. Then subnets were added. It was no longer possible to recognize from the address alone where the subnet prefix ends and where the host identifier starts. For example, the host part at EPFL is 8 bits; it is 6 bits at ETHZ. Therefore, an additional information, that is the subnet mask, is necessary.

Class C addresses were meant to be allocated one per network. Today they are allocated in contiguous blocks.

Special case IP addresses

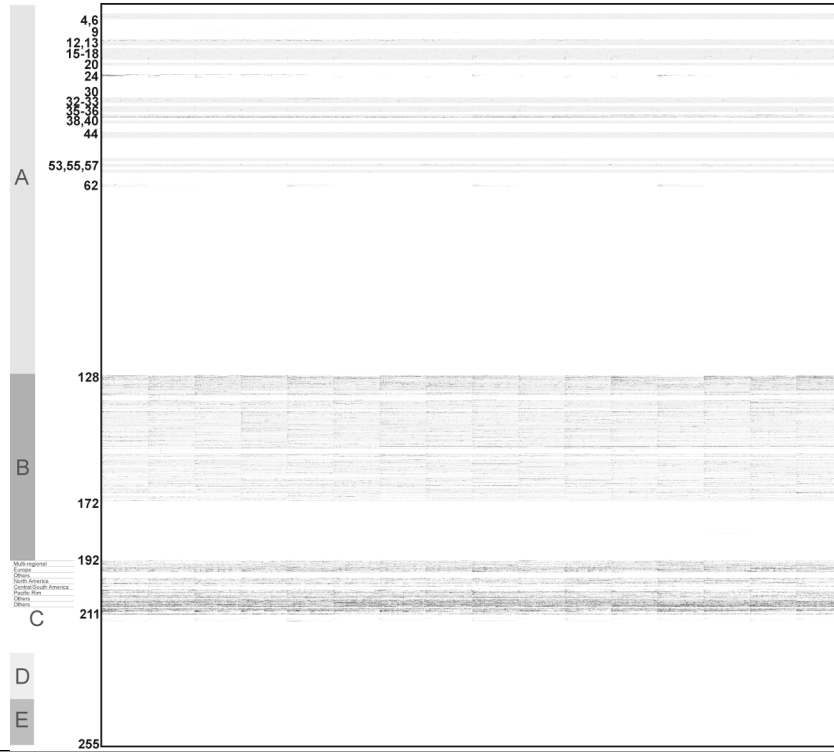
| | |
|---|--|
| 1. 0.0.0.0 | this host, on this network |
| 2. 0.hostId | specified host on this net (initialization phase) |
| 3. 255.255.255.255 | limited broadcast (not forwarded by routers) |
| 4. subnetId.all 1's | broadcast on this subnet |
| 5. subnetId.all 0's | BSD used it for broadcast on this subnet (obsolete) |
| 6. 127.x.x.x | loopback |
| 7. 10/8 | reserved networks for |
| 172.16/12 | internal use (Intranet) |
| 192.168/16 | |
| <ul style="list-style-type: none"> ■ 1,2: source IP@ only; 3,4,5: destination IP@ only | |

19

The following address blocks are reserved and cannot be used in the Internet. They are typically used in experimental or closed environments:

10.0.0.0 - 10.255.255.255 (10/8)
 172.16.0.0 - 172.31.255.255 (172.16/12)
 192.168.0.0 - 192.168.255.255 (192.168/16)

Used addresses in Internet



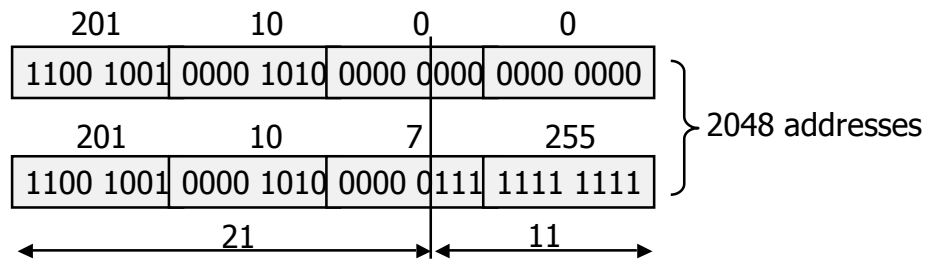
CIDR: IP Address Hierarchies

- The prefix of an IP address is itself structured in order to support aggregation
 - For example: 128.178.x.y represents an EPFL host
128.178.156 / 24 represents the LRC subnet at EPFL
128.178/15 represents EPFL
 - Used between routers by routing algorithms
 - This way of doing is called classless and was first introduced in inter domain routing under the name of CIDR (Classless Interdomain Routing)
- Notation: 128.178.0.0/16 means : the prefix made of the 16 first bits of the string
- It is equivalent to: 128.178.0.0 with netmask=255.255.0.0
- In the past, the class based addresses, with networks of class A, B or C was used; now only the distinction between class D and non-class D is relevant.

21

With so-called CIDRized (**CIDR: Classless Interdomain Routing**) network addresses, the network part of an IP address can be any number of bits long, rather than being constrained to 8, 16, or 24 bits. A CIDRized network address has the dotted-decimal form $a.b.c.d/x$, where x indicates the number of leading bits in the 32-bit quantity that constitutes the network portion of the address. In our example above, the organization needing to support 2,000 hosts could be allocated a block of only 2,048 host addresses of the form $a.b.c.d/21$, allowing the approximately 63,000 addresses that would have been allocated and unused under classful addressing to be allocated to a different organization. In this case, the first 21 bits specify the organization's network address and are common in the IP addresses of all hosts in the organization. The remaining 11 bits then identify the specific hosts in the organization. In practice, the organization could further divide these 11 rightmost bits using a procedure known as **subnetting** to create its own internal networks within the $a.b.c.d/21$ network.

CIDR



201.10.0.0/21: 201.10.0.0 - 201.10.0.255

201.10.1.0 - 201.10.1.255

...

201.10.7.0 - 201.10.7.255

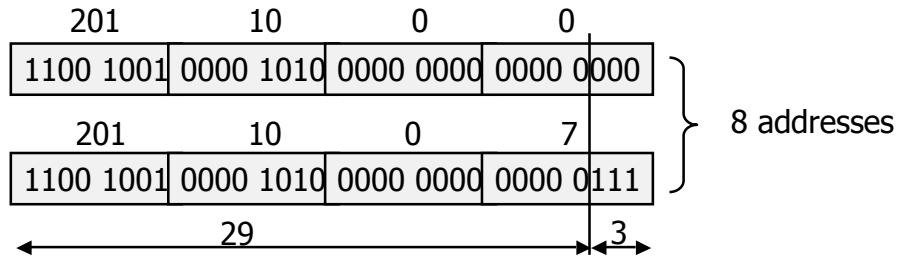
1 C class network: 256 addresses

$256 \times 8 = 2048$ addresses

22

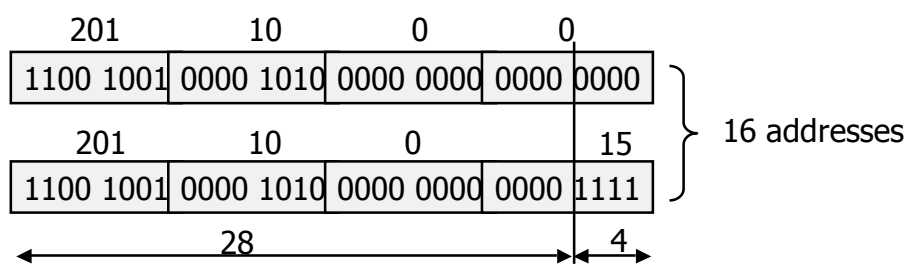
Notation 201.10.0.0/21 means: the prefix made of the 21 first bits of the string.

Choosing prefix length



- prefix = 201.10.0.0/29
 - 8 addresses
 - 2 broadcast addresses: 201.10.0.0, 201.10.0.7
 - only 6 addresses can be used for hosts

Choosing prefix length



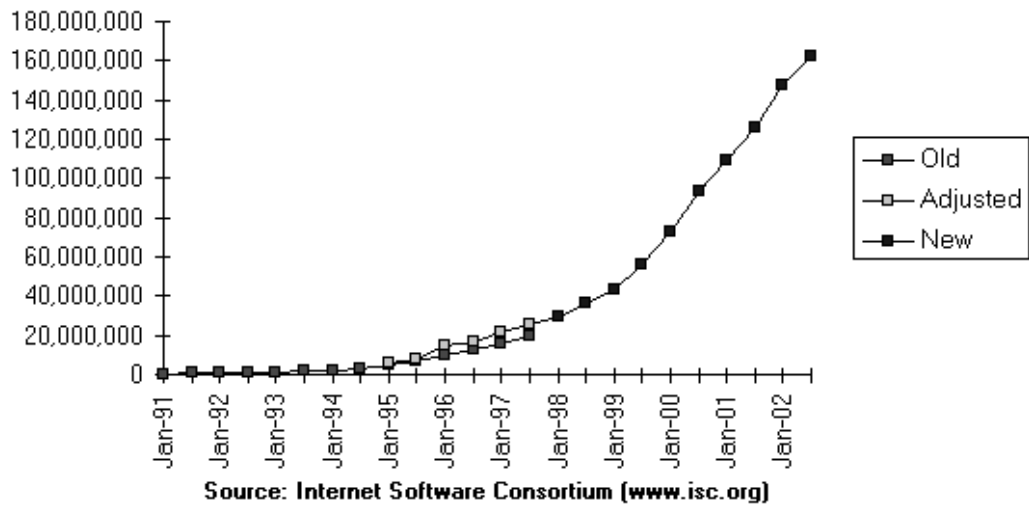
- prefix = 201.10.0.0/28
 - 201.10.0.16/28, 201.10.0.32/28, 201.10.0.48/28...
 - 16 addresses
 - 2 broadcast addresses: 201.10.0.0, 201.10.0.15
 - only 14 addresses can be used for hosts

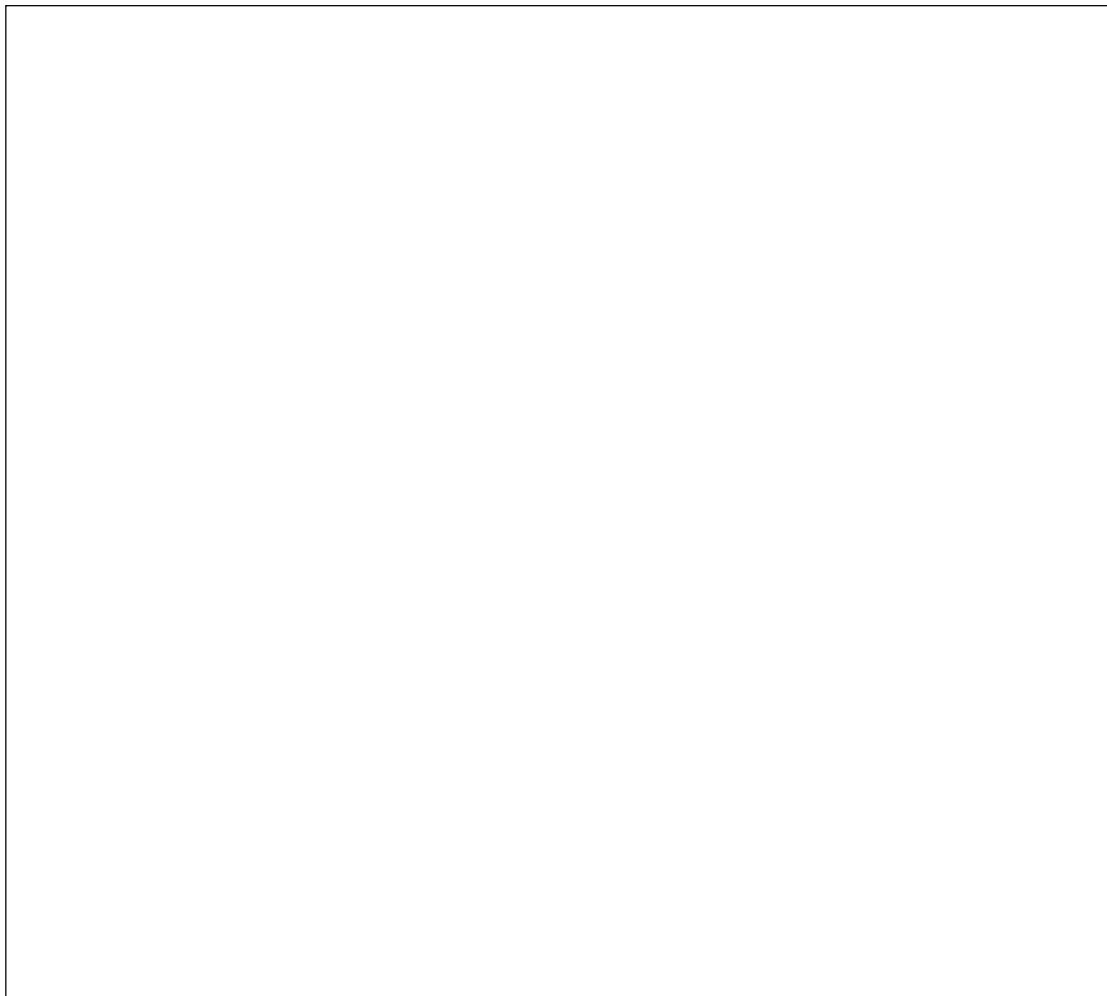
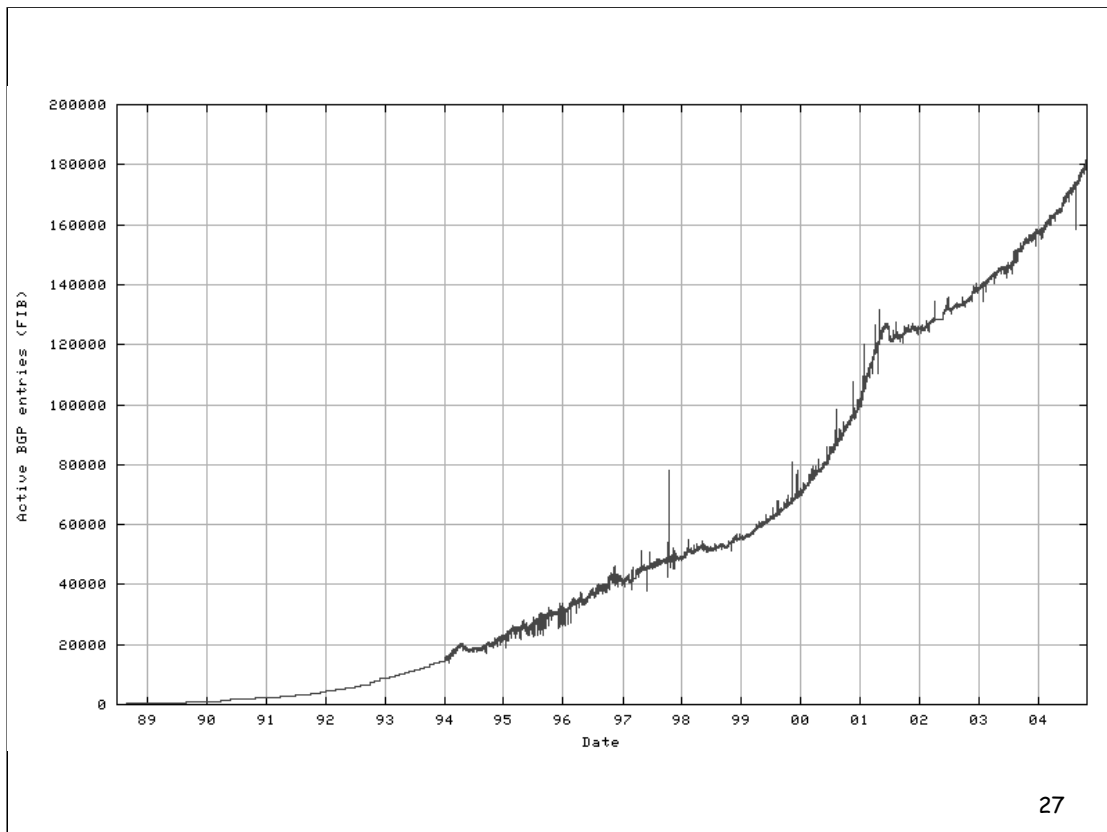
Address allocation

- World coverage
 - Europe and the Middle East (RIPE NCC)
 - Africa (ARIN & RIPE NCC)
 - North America (ARIN)
 - Latin America including the Caribbean (ARIN)
 - Asia-Pacific (APNIC)
- Current allocations of Class C
 - 193-195/8, 212-213/8, 217/8 for RIPE
 - 199-201/8, 204-209/8, 216/8 for ARIN
 - 202-203/8, 210-211/8, 218/8 for APNIC
- Simplifies routing
 - short prefix aggregates many subnetworks
 - routing decision is taken based on the short prefix

Number of hosts

Internet Domain Survey Host Count





IP Addresses and subnet mask

- subnet mask at ETHZ = 255.255.255.192
- CIDR **129.132/26**
- question: subnet prefix and host parts of spr13.tik.ee.ethz.ch = 129.132.119.77 ?

129.132.119.77 : 10000001.10000100.01110111.01001101

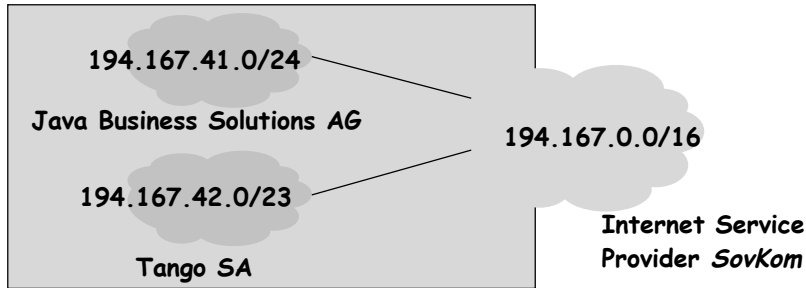
255.255.255.192: 11111111.11111111.11111111.11000000

answer:

subnet prefix = 129.132.119.64 (64=01000000)

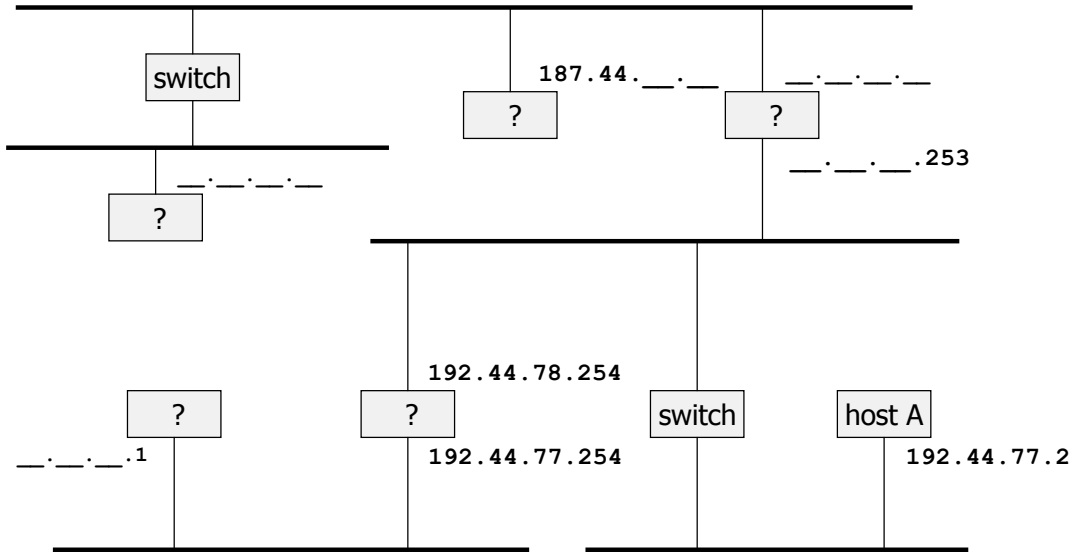
host = 13=001101 (6 bits)

IP Addresses



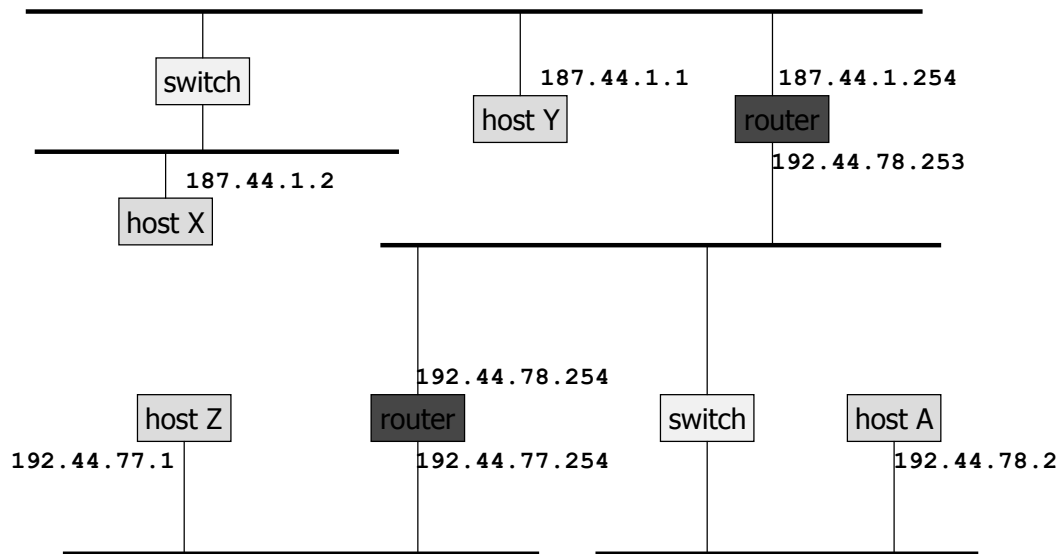
- **Sovkom** has received IP addresses 194.167.0.0 to 194.167.255.255 total: 2^{16} addr., but .0 and .255 are not usable
- **Java Business Solutions AG** has received IP addresses 194.167.41.0 to 194.167.41.255 total: $2^8 - 2$ addresses
- **Tango SA** has received IP addresses 194.167.42.0 to 194.167.43.255 total: $2^9 - 2$ addresses

Example



■ Can host A have this address?

Example



- Host A is on subnetwork 192.44.78

IP Principles

Homogeneous addressing

- an IP address is unique across the whole network (= the world in general)
- IP address is the address of the interface
- communication between IP hosts requires knowledge of IP addresses

Routing:

- inside a subnetwork: hosts communicate directly without routers
- between subnetworks: one or several routers are used
- a subnetwork = a collection of systems with a common prefix

32

We have seen the main principles of IP addressing: an IP address identifies an interface of an host (rather than the host itself); this address is unique in the Internet. In order to communicate, IP hosts need to know the IP addresses. We can distinguish between routing inside a subnetwork and routing between subnetworks, where a subnetwork is a collection of hosts that can communicate directly without routers.

IP packet forwarding algorithm

- Rule for sending packets (hosts, routers)
 - if the destination IP address has the same prefix as one of my interfaces, send directly to that interface
 - otherwise send to a router as given by the IP routing table

At lrcsuns: Next Hop Table

| destination@ | subnetMask | nextHop |
|--------------|------------|---------------|
| DEFAULT | | 128.178.156.1 |

Physical Interface Tables

| IP | subnetMask |
|----------------|---------------|
| 128.178.156.24 | 255.255.255.0 |

At in-inj: Next Hop Table

| destination@ | subnetMask | nextHop |
|---------------|---------------|---------------|
| 128.178.156.0 | 255.255.255.0 | 128.178.182.5 |
| DEFAULT | | 128.178.182.1 |

Physical Interface Tables

| IP | subnetMask |
|---------------|---------------|
| 128.178.79.1 | 255.255.255.0 |
| 128.178.182.3 | 255.255.255.0 |

33

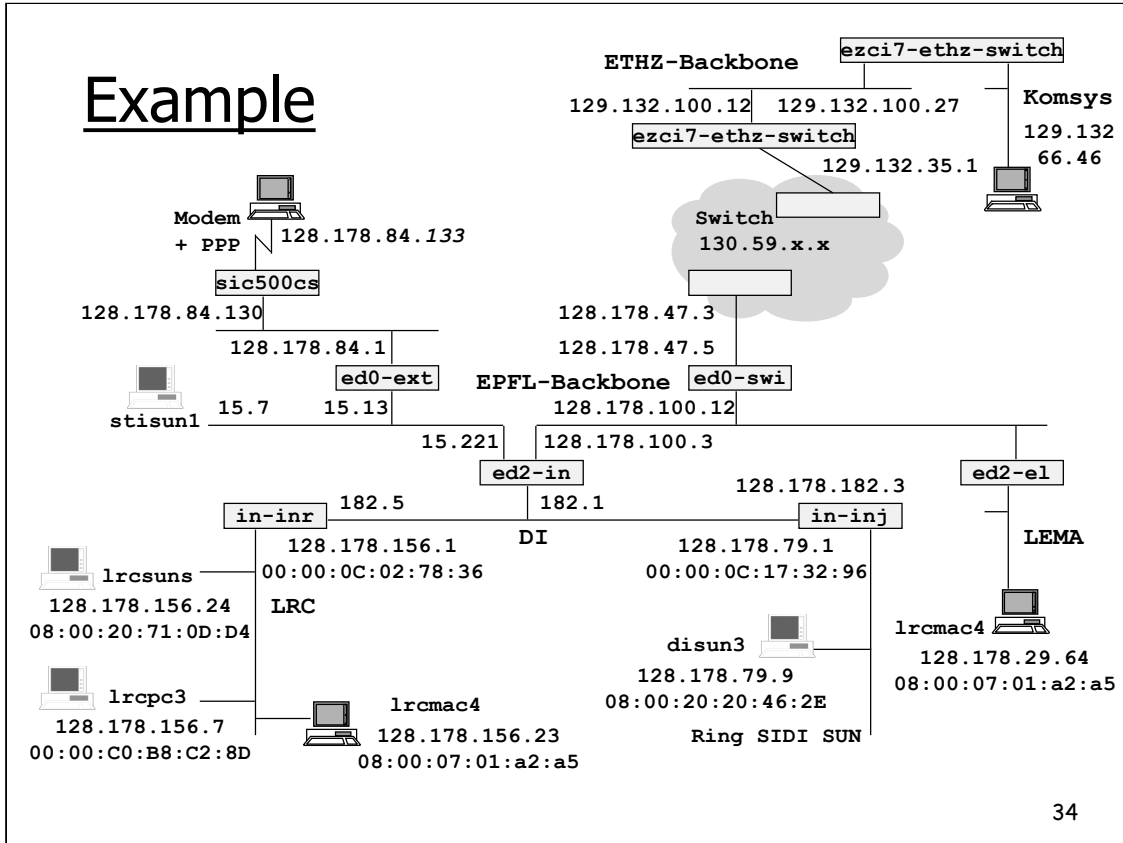
The IP packet forwarding algorithm is the core of the TCP/IP architecture. It defines what a system should do with a packet it has to send or forward. The rule is simple:

- if the destination IP address has the same prefix as one of my interfaces, send directly to that interface,
- otherwise send to a router as given by the IP routing table.

It uses the IP routing table; the table can be checked with a command such as “netstat” with Unix or “Route” with Windows NT.

The physical interface table can be checked with the command "ifconfig".

Example



IP packet forwarding algorithm

destAddr = packet dest. address, destinationAddr = address in routing table

Case 1: a **host route** exists for destAddr

for every entry in routing table
 if (destinationAddr = destAddr)
 then send to nextHop IPAddr; leave

Case 2: destAddr is on a **directly connected network** (= on-link):

for every physical interface IP address A and subnet mask SM
 if(A & SM = destAddr & SM)
 then send directly to destAddr; leave

Case 3: a **network route** exists for destAddr

for every entry in routing table and subnet mask SM
 if (destinationAddr & SM = destAddr & SM)
 then send to nextHop IP addr; leave

Case 4: use **default route**

for every entry in routing table
 if (destinationAddr=DEFAULT) then send to nextHop IPAddr; leave 35

The complete algorithm is as above; the cases should be tested in that order (it is a nested **if then else** statement).

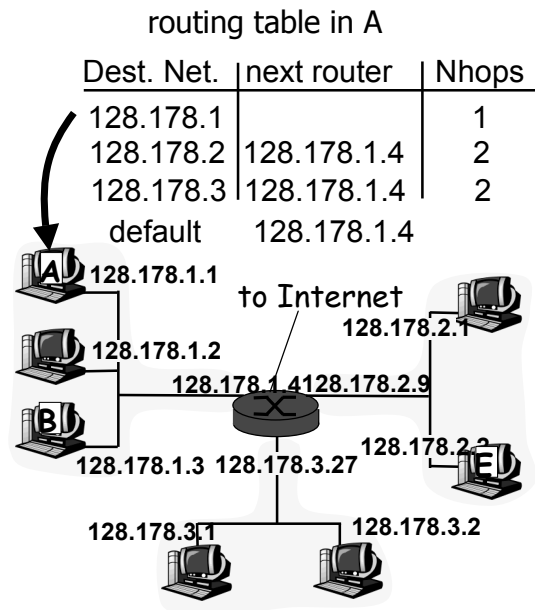
Remember that the above is the packet forwarding algorithm. The routing table is updated by a control method (a routing protocol).

Getting a datagram from source to dest.

IP datagram:

| misc fields | source IP addr | dest IP addr | data |
|-------------|----------------|--------------|------|
|-------------|----------------|--------------|------|

- datagram remains unchanged, as it travels source to destination
- addr fields of interest here



36

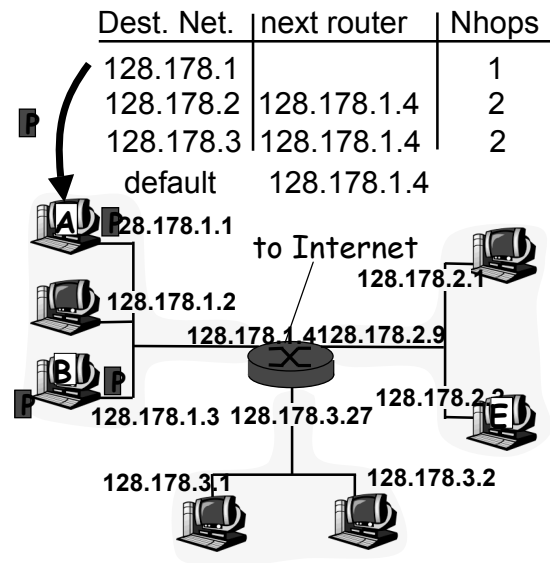
Every IP datagram has a source address field and a destination address field. The source host fills a datagram's source address field with its own 32-bit IP address. It fills the destination address field with the 32-bit IP address of the final destination host to which the datagram is being sent. The data field of the datagram is typically filled with a TCP or UDP segment. The IP datagram travels inside the network remaining unchanged. For routing purpose, the fields of main interest (e.g. the fields that are read and used) are the two addresses: source and destination. The way the network transports the datagram from the source to the destination depends on whether the source and destination reside on the same subnetwork.

Getting a datagram from source to dest.: same subnetwork

| | | | |
|--------|-------------|-------------|------|
| misc | 128.178.1.1 | 128.178.1.3 | data |
| fields | | | |

Starting at A, given IP datagram addressed to B:

- look up net. address of B
- find B is on same net. as A
- link layer will send datagram directly to B inside link-layer frame
 - B and A are directly connected



37

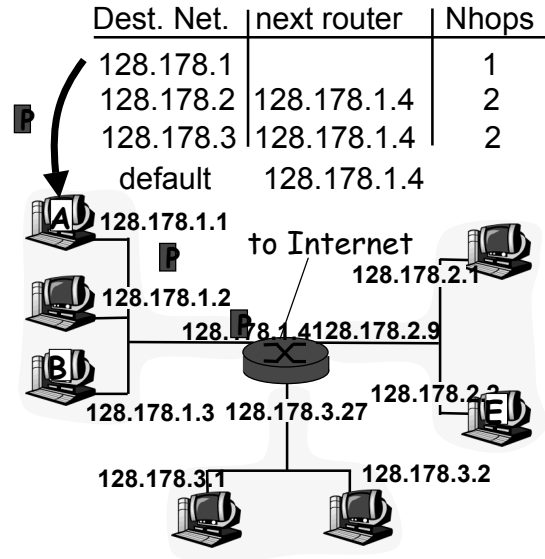
Host *A* wants to send an IP datagram to host *B*, which resides on the same network, 128.178.1.0/24, as *A*. This is accomplished as follows. IP in host *A* first consults its internal routing table, shown in Figure 4.22, and finds an entry, 128.178.1.0/24, whose network address matches the leading bits in the IP address of host *B*. The routing table shows that the destination host is on the same subnet as host *A* ($\text{Address-of-A} \ \& \ \text{SM} = \text{destAddr} \ \& \ \text{SM}$). Host *A* thus knows that destination host *B* can be reached directly via *A*'s outgoing interface, without the need for any intervening routers. Host *A* then passes the IP datagram to the link-layer protocol for the interface, which then has the responsibility of transporting the datagram to host *B*.

Getting a datagram from source to dest.: different subnetworks

| | | | |
|--------|-------------|-------------|------|
| misc | 128.178.1.1 | 128.178.2.3 | data |
| fields | | | |

Starting at A, dest. E:

- look up network address of E
- E on *different* network
 - A, E not directly attached
- routing table: next hop router to E is 128.178.1.4
- link layer sends datagram to router 128.178.1.4 inside link-layer frame
- datagram arrives at 128.178.1.4
- continued.....



38

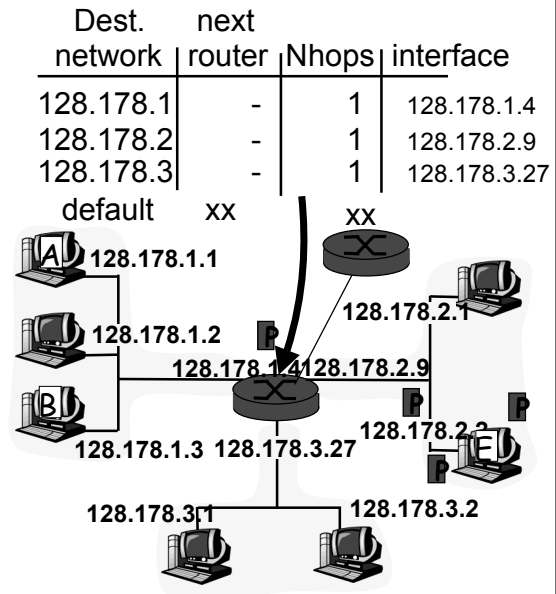
Host *A* wants to send a datagram to another host, say *E*, that is on a different network. Host *A* again consults its routing table and finds an entry, 128.178.2.0/24, whose network address matches the leading bits in the IP address of host *E*. The routing table tells host *A* that in order to get the datagram to host *E*, host *A* should first send the datagram to IP address 128.178.1.4, the router interface to which *A*'s own interface is directly connected. IP in host *A* then passes the datagram down to the link layer and indicates to the link layer that it should send the datagram to IP address 128.178.1.4. It's important to note here that although the datagram is being sent (via the link layer) to the router's interface, the destination address of the datagram remains that of the ultimate destination (host *E*), *not* that of the intermediate router interface.

Getting a datagram from source to dest.: different subnetworks

| | | | |
|--------|-------------|-------------|------|
| misc | 128.178.1.1 | 128.178.2.3 | data |
| fields | | | |

Arriving at 128.178.1.4,
destined for 128.178.2.2

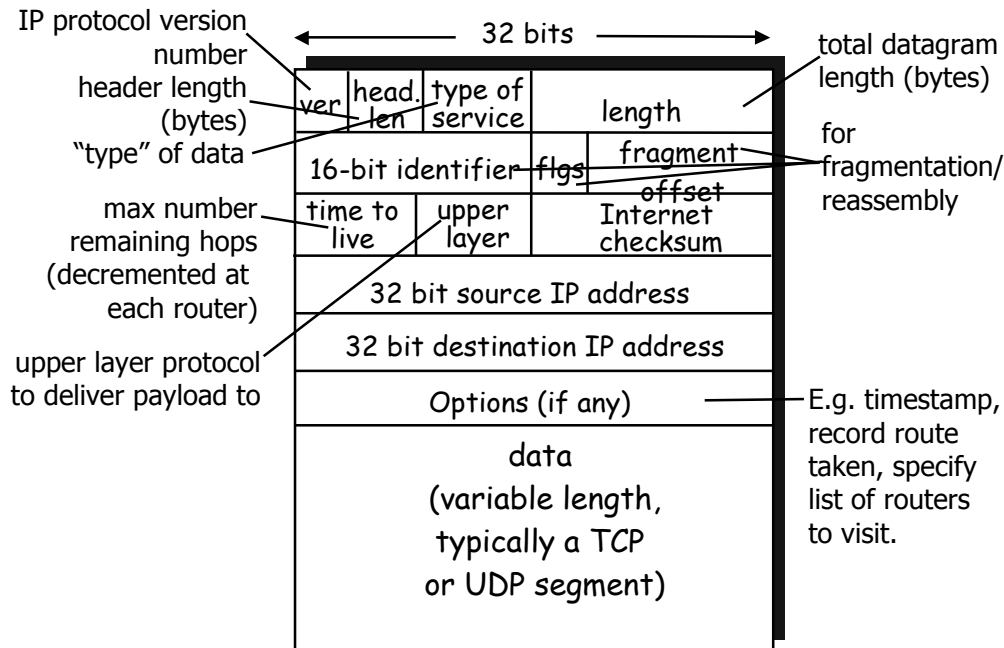
- look up network address of E
- E on *same* network as router's interface 128.178.2.9
 - router, E directly attached
- link layer sends datagram to 128.178.2.2 inside link-layer frame via interface 128.178.2.9
- datagram arrives at 128.178.2.2!!! (hooray!)



39

The datagram is now in the router, and it is the job of the router to move the datagram toward its ultimate destination. The router consults its own routing table and finds an entry, 128.178.2.0/24, whose network address matches the leading bits in the IP address of host *E*. The routing table indicates that the datagram should be forwarded on router interface 128.178.2.9 and that destination host *E* is on the same network as its own interface, 128.178.2.9. The router thus moves the datagram to this interface, which then transmits the datagram to host *E*.

IP datagram format



40

The key fields in the IPv4 datagram are the following:

- **Version Number.** These four bits specify the IP protocol version of the datagram. By looking at the version number, the router can then determine how to interpret the remainder of the IP datagram. Different versions of IP use different datagram formats.
- **Header Length.** Because an IPv4 datagram can contain a variable number of options (that are included in the IPv4 datagram header) these four bits are needed to determine where in the IP datagram the data actually begins. Most IP datagrams do not contain options so the typical IP datagram has a 20-byte header.
- **TOS.** The type of service (TOS) bits were included in the IPv4 header to allow different "types" of IP datagrams to be distinguished from each other, presumably so that they could be handled differently in times of overload. When the network is overloaded, for example, it would be useful to be able to distinguish network-control datagrams from datagrams carrying data. It would also be useful to distinguish real-time datagrams from non-real-time traffic. Now, the TOS field is redefined in *DiffServ* (Differentiated Services): 1 byte codepoint determining QoS class: Expedited Forwarding (EF) - minimize delay and jitter; Assured Forwarding (AF) - four classes and three drop-precedences (12 codepoints).
- **Datagram Length.** This is the total length of the IP datagram (header plus data) measured in bytes. Since this field is 16 bits long, the theoretical maximum size of the IP datagram is 65,535 bytes. However, datagrams are rarely greater than 1,500 bytes and are often limited in size to 576 bytes (every subnet should forward packets of $576 = 512 + 64$ bytes).
- **Identifier, Flags, Fragmentation Offset.** These three fields have to do with so-called IP fragmentation, a topic we will consider in depth shortly. Interestingly, the new version of IP, IPv6, does not allow for fragmentation at routers.
- **Time-to-live.** The time-to-live (TTL) field is included to ensure that datagrams do not circulate forever (due to, for example, a long-lived router loop) in the network. This field is decremented by one each time the datagram is processed by a router. If the TTL field reaches 0, the datagram must be dropped.
- **Protocol.** This field is used only when an IP datagram reaches its final destination. The value of this field indicates the transport-layer protocol at the destination to which the data portion of this IP datagram will be passed. For example, a value of 6 indicates that the data portion is passed to TCP, while a value of 17 indicates that the data is passed to UDP. For a listing of all possible numbers, see RFC 1700.
- **Header Checksum.** The header checksum aids a router in detecting bit errors in a received IP datagram. The header checksum is computed by treating each two bytes in the header as a number and summing these numbers using 1's complement arithmetic.
- **Source and Destination IP Address.** These fields carry the 32-bit IP address of the source and final destination for this IP datagram. The use and importance of the destination address is clear.
- **Options.** The options fields allow an IP header to be extended: *strict source routing* (gives the route to the destination: all routers), *loose source routing* (gives the route to the destination: some routers), record route, timestamp route, router alert (used by IGMP or RSVP for processing a packet)
- **Data (payload).** The data field of the IP datagram contains the transport-layer segment (TCP or UDP) to be delivered to the destination. However, the data field can carry other types of data, such as ICMP messages

IP header

- Version
 - IPv4, futur IPv6
- Header size
 - options - variable size
 - in 32 bit words
- Type of service
 - priority : 0 - normal, 7 - control packets
 - short delay (telnet), high throughput (ftp), high reliability (SNMP), low cost (NNTP)
- Redefined in *DiffServ* (Differentiated Services)
 - 1 byte codepoint determining QoS class
 - Expedited Forwarding (EF) - minimize delay and jitter
 - Assured Forwarding (AF) - four classes and three drop-precedences (12 codepoints)

IP header

- Packet size
 - in bytes including header
 - in bytes including header
 - ≤ 64 Kbytes; limited in practice by link-level MTU (*Maximum Transmission Unit*)
 - every subnet should forward packets of $576 = 512 + 64$ bytes
- Id
 - unique identifier for re-assembling
- Flags
 - M : *more* ; set in fragments
 - F : prohibits fragmentation

IP header

- Offset
 - position of a fragment in multiples of 8 bytes
- TTL (*Time-to-live*)
 - in secondes
 - now: number of hops
 - router : --, if 0, drop (send ICMP packet to source)
- Protocol
 - identifier of protocol (1 - ICMP, 6 - TCP, 17 - UDP)
- Checksum
 - only on the header

IP header

- Options
 - *strict source routing*
 - all routers
 - *loose source routing*
 - some routers
 - record route
 - timestamp route
 - router alert
 - used by IGMP or RSVP for processing a packet

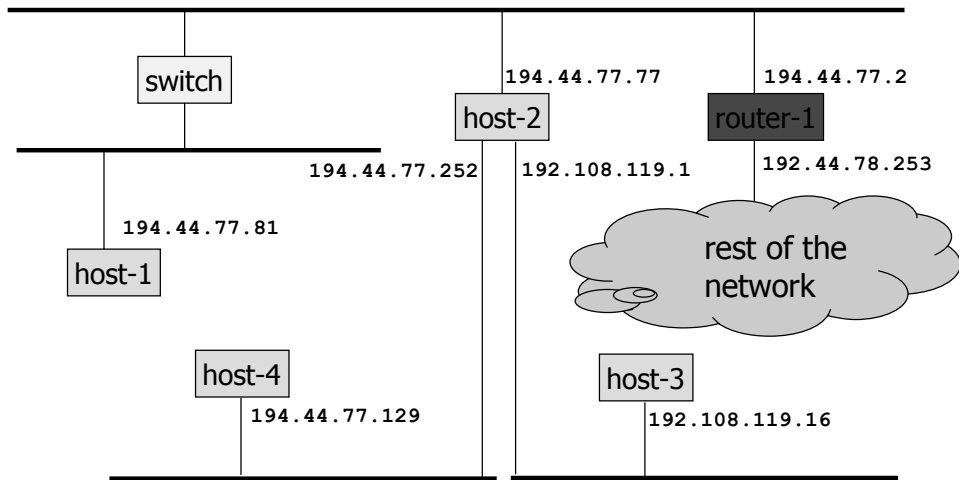
Configuration of a Unix host

```
/usr/etc/ifconfig interface [ address_family ]
    [ address [ dest_address ] ] [ netmask mask ]
    [ broadcast address ] [ up ] [ down ] [ trailers
]
    [ -trailers ] [ arp ] [ -arp ] [ private ]
    [ -private ] [ metric n ] [ auto-revarp ]
```

```
host-1# ifconfig le0 host-1 netmask +
Setting netmask of le0 to 255.255.255.128
# + means netmask from /etc/netmasks
host-1# ifconfig -a
le0: flags=863<UP,BROADCAST,NOTRAILERS,RUNNING>
    inet 192.44.77.81 netmask ffffffff broadcast 192.44.77.0
    ether 8:0:20:1c:74:84
lo0: flags=849<UP,LOOPBACK,RUNNING>
    inet 127.0.0.1 netmask ff000000
```

45

Example interconnection



Routing tables

host-1 (192.44.77.81) :

>netstat -n -r

Routing tables

| Destination | Gateway | Flags | Refcnt | Use | Interface |
|----------------|--------------|-------|--------|-------|-----------|
| 192.108.119.16 | 192.44.77.77 | UGHD | 1 | 1683 | 1e0 |
| 127.0.0.1 | 127.0.0.1 | UH | 2 | 12971 | 1o0 |
| default | 192.44.77.2 | UG | 3 | 16977 | 1e0 |
| 192.44.77.0 | 192.44.77.81 | U | 13 | 5780 | 1e0 |

U - up

G - gateway (next router)

H - host route

D - route from ICMP Redirect

Routing tables

```
host-2 (192.44.77.77) :  
>rsh host-2 netstat -n -r  
Routing tables  
Destination      Gateway          Flags  Refcnt  Use      Interface  
127.0.0.1        127.0.0.1       UH     3        351344   lo0  
default          192.44.77.2     UG     3        17388997 le0  
192.44.77.128    192.44.77.252  U      26       504768   le2  
192.44.77.0      192.44.77.77   U      24      10702069 le0  
192.108.119.0    192.108.119.1  U       2        249777   le1
```

48

How many network interfaces does host-2 has?

Which route is taken by a packet sent by host-1 to 192.108.119.16?

Which route is taken by a packet sent by host-1 to 192.44.77.129?

Modifying routing tables

```
/usr/etc/route [ -fn ] add|delete [ host|net ]
  destination [gateway [ metric ] ]
host-1# netstat -r
Routing tables
Destination      Gateway          Flags           Refcnt  Use
Interface
localhost        localhost       UH              2        13569  lo0
192.44.77.0      host-1          U               18        13272  le0
host-1# ping 133.11.11.11
sendto: Network is unreachable
host-1# route add 0.0.0.0 router-1 1
add net 0.0.0.0 gateway router-1
```

Modifying routing tables

```
host-1# netstat -r
```

```
Routing tables
```

| Destination | Gateway | Flags | Refcnt | Use | |
|-------------|-----------|-------|--------|-------|-----|
| Interface | | | | | |
| localhost | localhost | UH | 2 | 13591 | lo0 |
| default | router-1 | UG | 0 | 0 | le0 |
| 192.44.77.0 | host-1 | U | 16 | 13566 | le0 |

```
host-1# ping 133.11.11.11
```

```
133.11.11.11 is alive
```

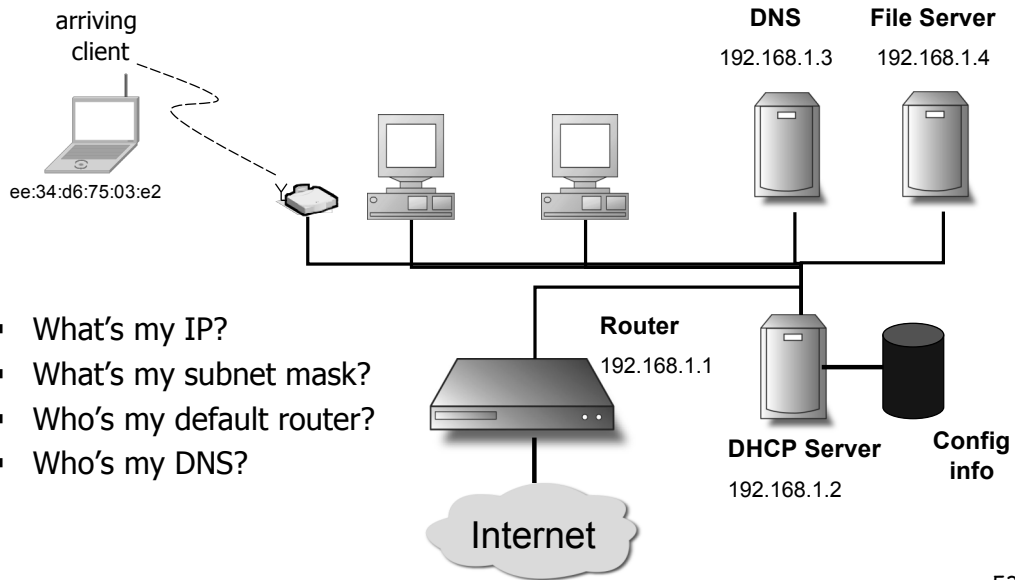
DHCP

- DHCP
 - Dynamic Host Configuration Protocol (RFC 2131)
- Goal: allow host to dynamically obtain its IP address from network server when it joins network
 - Support for mobile users who want to join network
 - Allows reuse of addresses (hold address only while connected)
- Uses UDP port 67 (to server) and 68 (to client)
 - IP source address 0, broadcast 255.255.255.255

DHCP

- Dynamic addresses
 - 2 databases
 - Static DB - Matches IP's and Physical Addresses
 - Dynamic DB - Pool of IP's leased out
- Temporary addresses
 - Addresses leased from Dynamic DB are temporary
 - Each lease has an expiration which the client must obey
 - Can renew its lease on address in use

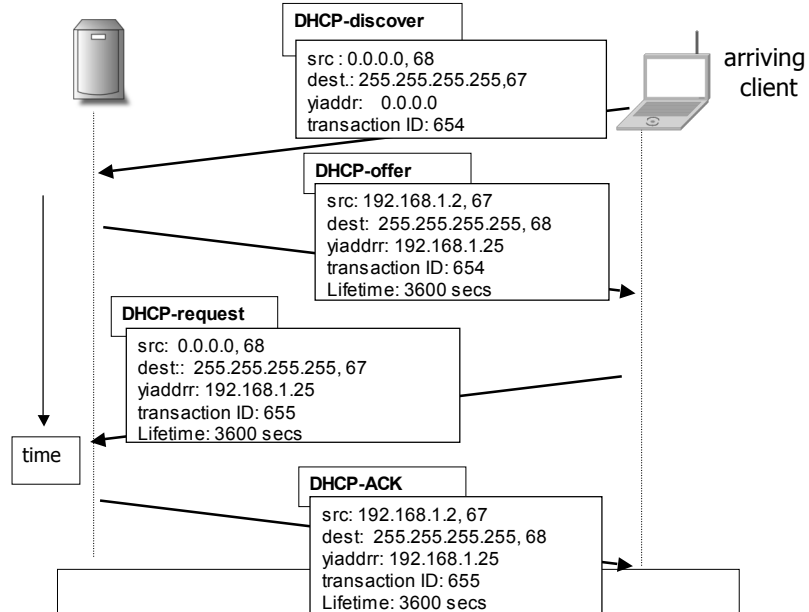
DHCP



53

DHCP client-server scenario

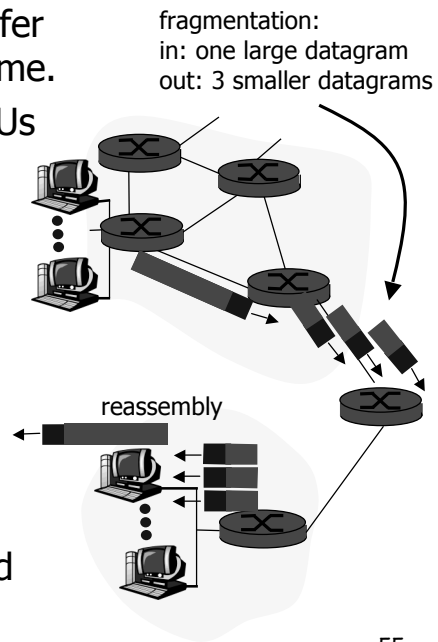
DHCP server: 192.168.1.2



54

IP Fragmentation & Reassembly

- network links have MTU (max.transfer size) - largest possible link-level frame.
 - different link types, different MTUs
- large IP datagram divided ("fragmented") within net
 - one datagram becomes several datagrams
 - "reassembled" only at final destination
 - IP header bits used to identify, order related fragments
- fragmentation is in principle avoided with TCP and UDP using small segments



Suppose you receive an IP datagram from one link, you check your routing table to determine the outgoing link, and this outgoing link has an MTU that is smaller than the length of the IP datagram. Time to panic--how are you going to squeeze this oversized IP packet into the payload field of the link-layer packet? The solution to this problem is to "fragment" the data in the IP datagram among two or more smaller IP datagrams, and then send these smaller datagrams over the outgoing link. Each of these smaller datagrams is referred to as a **fragment**.

Fragments need to be reassembled before they reach the transport layer at the destination. Indeed, both TCP and UDP are expecting to receive complete, unfragmented segments from the network layer. However, Fragmentation and reassembly puts an additional burden on Internet routers and on the destination hosts. For this reason it is desirable to keep fragmentation to a minimum. This is often done by limiting the TCP and UDP segments to a relatively small size, so that fragmentation of the corresponding datagrams is unlikely.

MTU: Maximum Transfer Unit

Data links have different maximum packet length

- MTU (maximum transmission unit) = maximum packet size usable for an IP packet
- value of short MTU ? of long MTU ?

| Link technology | MTU |
|---------------------|-------------|
| Ethernet | 1500 |
| 802.3 with LLC/SNAP | 1492 |
| FDDI | 4352 |
| X.25 | 576 |
| Frame Relay | 1600 |
| ATM with AAL5 | 9180 |
| Hyperchannel | 65535 |
| PPP | 296 to 1500 |

```
lrcsuns$ ifconfig -a
lo0: flags=849<UP,LOOPBACK,RUNNING,MULTICAST> mtu 8232
    inet 127.0.0.1 netmask ff000000
le0: flags=863<UP,BROADCAST,NOTRAILERS,RUNNING,MULTICAST> mtu 1500
    inet 128.178.156.24 netmask fffffff0 broadcast 128.178.156.255
    ether 8:0:20:71:d:d4
```

56

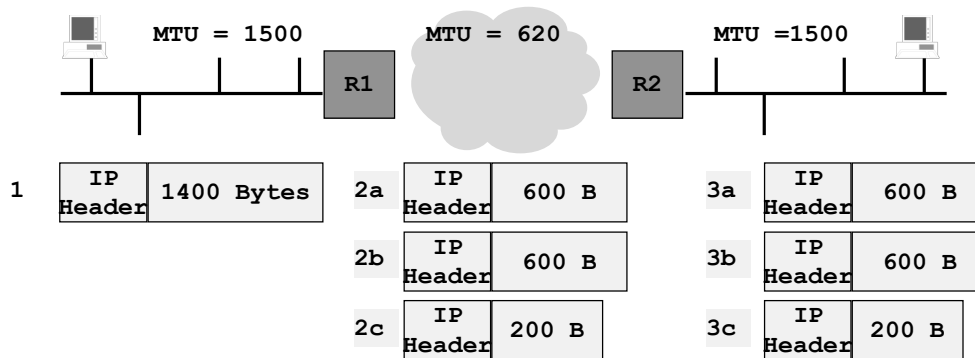
The maximum amount of data that a link-layer packet can carry is called the **MTU (maximum transfer unit)**. Because each IP datagram is encapsulated within the link-layer packet for transport from one router to the next router, the MTU of the link-layer protocol places a hard limit on the length of an IP datagram. Having a hard limit on the size of an IP datagram is not much of a problem. What is a problem is that each of the links along the route between sender and destination can use different link-layer protocols, and each of these protocols can have different MTUs.

- Modem link: short MTU 1000 B at 9600 b/s = 530 ms too large for interactive traffic
- large MTU = higher throughput less overhead(TCP + IP = 40 bytes header overhead)

IP fragmentation

IP hosts or routers may have IP datagrams larger than MTU

- fragmentation is performed when IP datagram too large
- re-assembly is only at destination
- fragmentation is in principle avoided with TCP



IP fragmentation

- IP datagram is *fragmented* if
 $\text{MTU of interface} < \text{datagram total length}$
- all fragments are self-contained IP packets
- fragmentation controlled by fields: Identification, Flag and Fragment Offset
- IP *datagram* = original ; IP *packet* = fragments or complete datagram

| | 1 | 2a | 2b | 2c |
|--------------------|------|-----|-----|------|
| Length | 1420 | 620 | 620 | 220 |
| Identification | 567 | 567 | 567 | 567 |
| More Fragment flag | 0 | 1 | 1 | 0 |
| Offset | 0 | 0 | 75 | 150 |
| 8 * Offset | 0 | 0 | 600 | 1200 |

Fragment data size (here 600) is always a multiple of 8
 Identification given by source

58

Value of fields at point 3?

TCP, UDP and fragmentation

- The UDP service interface accepts a datagram up to 64 KB
 - UDP datagram passed to the IP service interface as one SDU
 - is fragmented at the source if resulting IP datagram is too large
- The TCP service interface is stream oriented
 - packetization is done by TCP
 - several calls to the TCP service interface may be grouped into one TCP segment (many small pieces)
 - or: one call may cause several segments to be created (one large piece)
 - TCP always creates a segment that fits in one IP packet: no fragmentation at source
 - fragmentation may occur in a router, if IPv4 is used, and if PMTU discovery is not implemented

LAN Addresses and ARP

32-bit IP address:

- *network-layer* address
- used to get datagram to destination network (recall IP network definition)

LAN (or MAC or physical) address:

- used to get datagram from one interface to another physically-connected interface (same network)
- 48 bit MAC address (for most LANs) burned in the adapter ROM

Why different addresses at IP and MAC?

- LANs not only for IP (LAN addresses are neutral)
- if IP addresses used, they should be stored in a RAM and reconfigured when host moves
- independency of layers

60

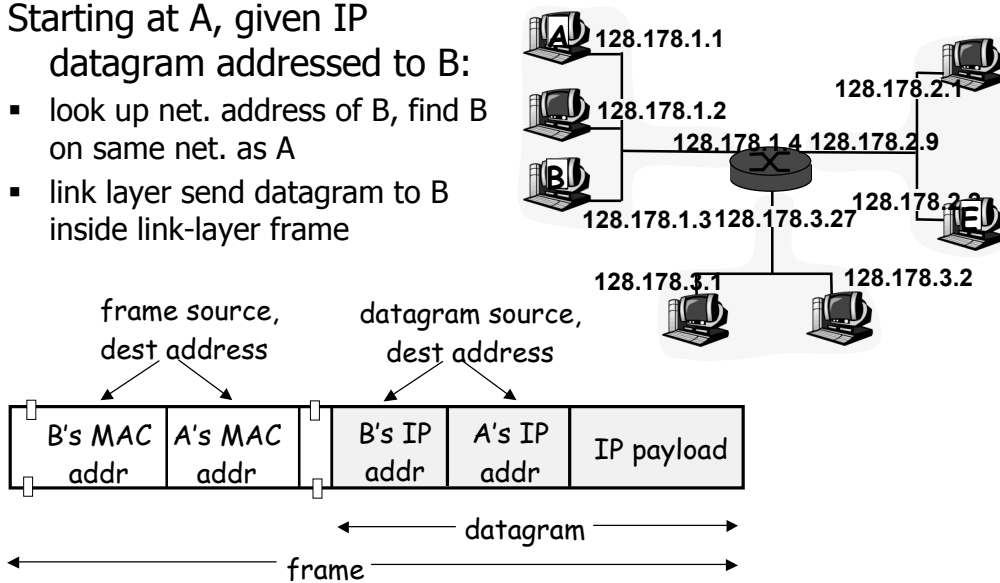
A **LAN address** is also variously called a **physical address**, an **Ethernet address**, or a **MAC** (media access control) **address**. For most LANs (including Ethernet and token-passing LANs), the LAN address is six-bytes long, giving 2^{48} possible LAN addresses. These six-byte addresses are typically expressed in hexadecimal notation, with each byte of the address expressed as a pair of hexadecimal numbers.

MAC Address resolution

Starting at A, given IP

datagram addressed to B:

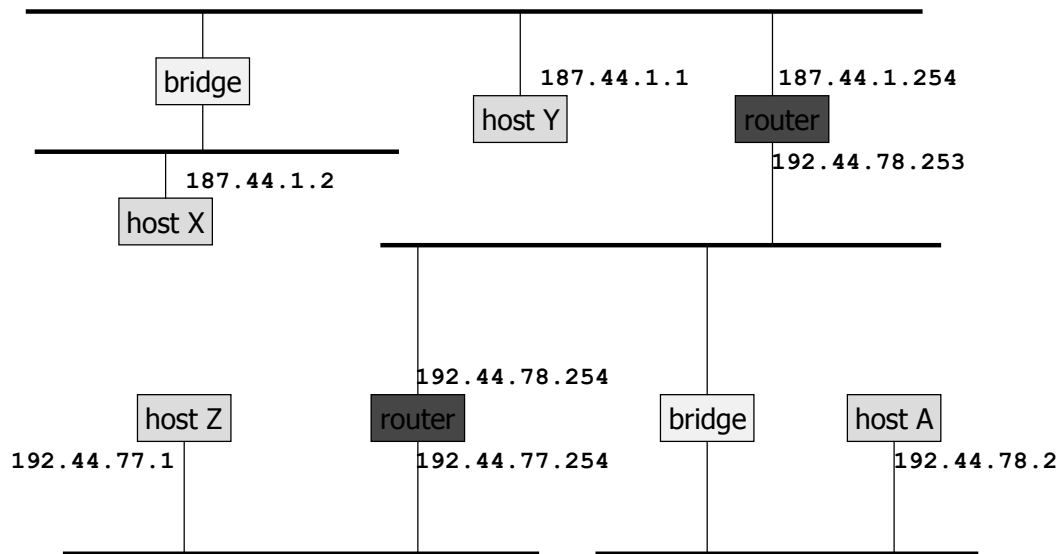
- look up net. address of B, find B on same net. as A
- link layer send datagram to B inside link-layer frame



61

Because there are both network-layer addresses (for example, Internet IP addresses) and link-layer addresses (that is, LAN addresses), there is a need to translate between them. The IP datagram contains the IP addresses of source and destination. Once the datagram is passed to the link-layer, it is necessary to specify the MAC address of the destination, in order to transmit the link-layer frame.

Example



■ Host A is on subnetwork 192.44.78

Packet delivery

Packet sent by 187.44.1.2 to
187.44.1.1

| | | | | |
|------------|------------|------------|------------|---------|
| MAC-host-Y | MAC-host-X | 187.44.1.1 | 187.44.1.2 | payload |
|------------|------------|------------|------------|---------|

Ethernet header IP header
X needs to know MAC address of Y
(ARP)

Packet sent by 187.44.1.2 to 192.44.78.2

| | | | | |
|------------|------------|-------------|------------|---------|
| MAC-router | MAC-host-X | 192.44.78.2 | 187.44.1.2 | payload |
|------------|------------|-------------|------------|---------|

Ethernet header IP header

| | | | | |
|------------|------------|-------------|------------|---------|
| MAC-host-A | MAC-router | 192.44.78.2 | 187.44.1.2 | payload |
|------------|------------|-------------|------------|---------|

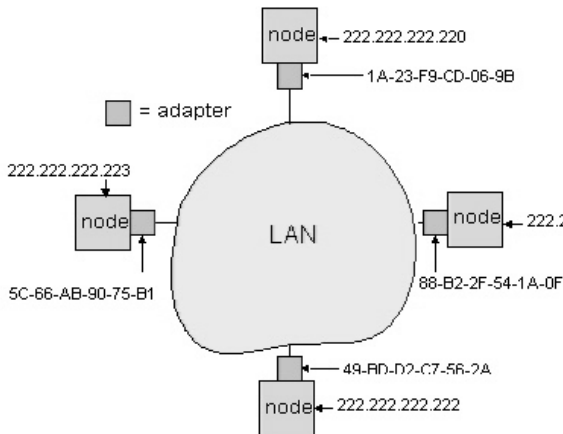
Ethernet header IP header

X needs to know MAC address of router (X knows the IP
address of router - configuration)

Router needs to know MAC address of A

ARP: Address Resolution Protocol

ARP is used to determine the MAC address of B given B's IP address



- Each IP node (Host, Router) on LAN implements ARP protocol and has ARP table

- ARP Table: IP/MAC address mappings for some LAN nodes

< IP address; MAC address >

- ARP table is a cache: after an interval (typically 20 min) the address mapping will be forgotten

64

For the Internet, this is the job of the address resolution protocol (ARP) [RFC 826]. Every Internet host and router on a LAN has an **ARP module**. ARP resolves an IP address to a LAN address ***only* for nodes on the same LAN**. The ARP module in each node has a table in its RAM called an **ARP table**. This table contains the mappings of IP addresses to LAN addresses.

An ARP entry is deleted from the table after an interval (typically 20 minutes).

ARP protocol

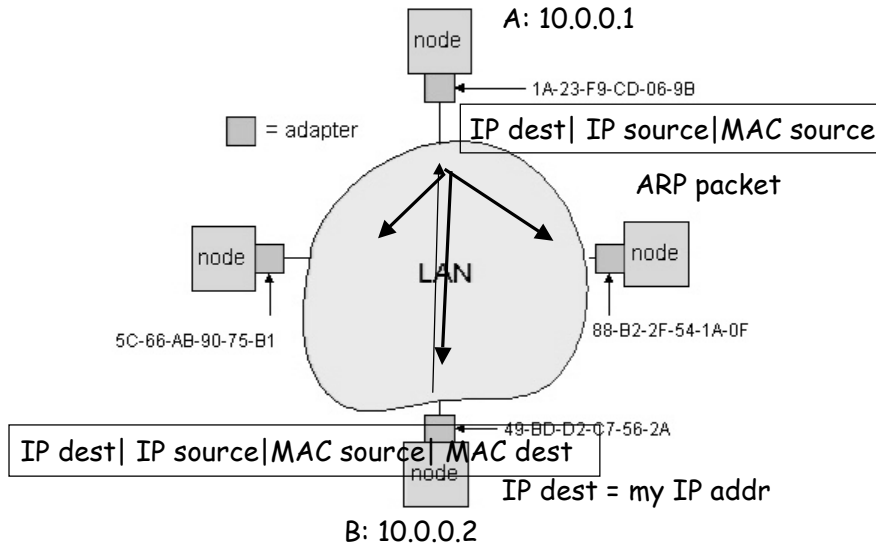
- A knows B's IP address, wants to learn physical address of B
- A broadcasts ARP query pkt, containing B's IP address
 - all machines on LAN receive ARP query
- B receives ARP packet, replies to A with its (B's) physical layer address
- A caches (saves) IP-to-physical address pairs until information becomes old (times out)
 - soft state: information that times out (goes away) unless refreshed

65

If the table does not contain the MAC address of the destination, the source constructs a special packet called an **ARP packet**. An ARP packet has several fields, including the sending and receiving IP and LAN addresses. Both ARP query and response packets have the same format. The purpose of the ARP query packet is to query all the other nodes on the LAN to determine the LAN address corresponding to the IP address that is being resolved. If the MAC address is returned, the querying node can then update its ARP table and send its IP datagram.

ARP protocol

| IP address | MAC address | TTL |
|------------|-------------------|---------|
| 10.0.0.2 | 49:BD:D2:07:56:2A | 6:00:00 |

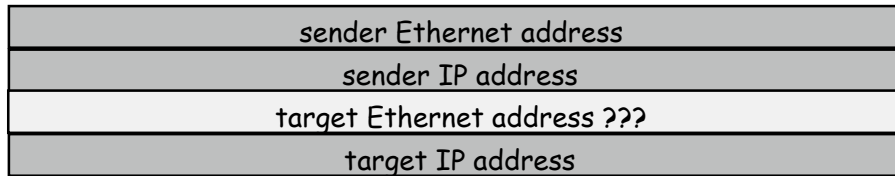


66

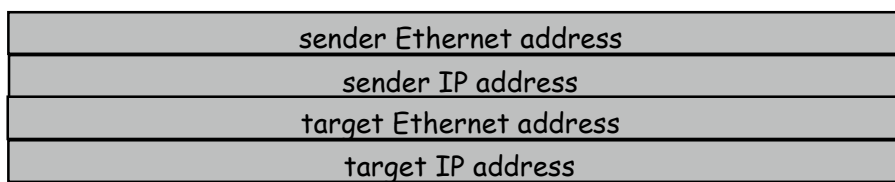
If the table does not contain the MAC address of the destination, the source constructs a special packet called an **ARP packet**. An ARP packet has several fields, including the sending and receiving IP and LAN addresses. Both ARP query and response packets have the same format. The purpose of the ARP query packet is to query all the other nodes on the LAN to determine the LAN address corresponding to the IP address that is being resolved. If the MAC address is returned, the querying node can then update its ARP table and send its IP datagram.

ARP frame

- Request (broadcast)



- Reply (unicast)



67

ARP Request sent

Wait for reply

- if received, put target information into the cache
- if not, repeat on timeout, increase timeout on each failure

ARP cache

- entry maintained for 20 minutes (on BSD), or less (Linux)
- incomplete entry (no reply), 3 minutes

ARP Paquet received

If IP address = our address

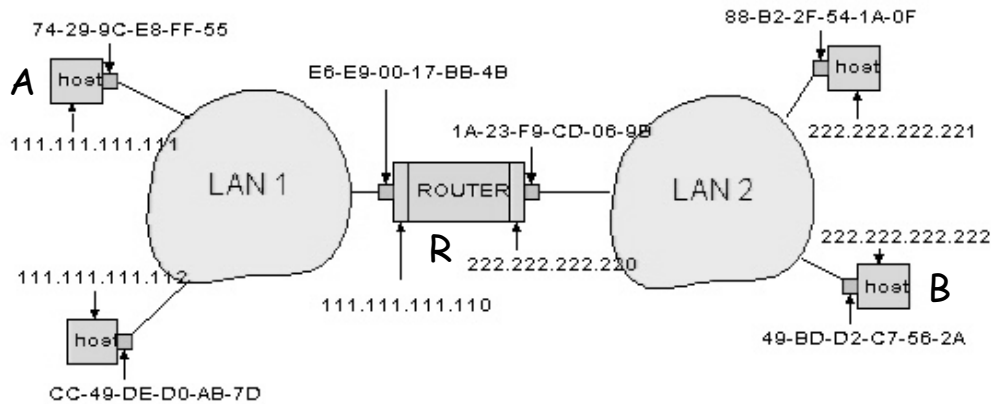
- put the sender information into the cache
- reply with our Ethernet address as target

Otherwise

- if IP address in the cache
 - update entry
- otherwise
 - do not update

Routing to another LAN

walkthrough: routing from A to B via R



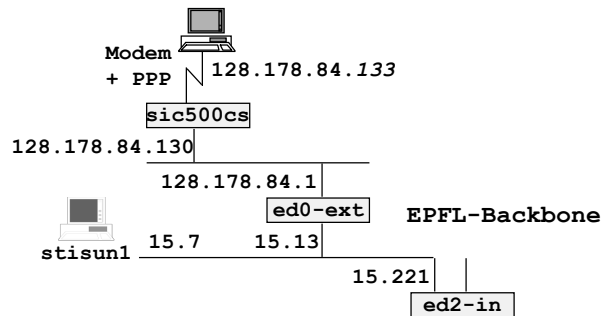
- In routing table at source Host, find router 111.111.111.110
- In ARP table at source, find MAC address E6-E9-00-17-BB-4B, etc

68

ARP operates when a node wants to send a datagram to another node *on the same LAN*. The situation is more complex when a node on a LAN wants to send a network-layer datagram to a node *off the LAN*. All of the interfaces connected to LAN 1 have addresses of the form 111.111.111.xxx and all of the interfaces connected to LAN 2 have the form 222.222.222.xxx. Now suppose that host 111.111.111.111 wants to send an IP datagram to host 222.222.222.222. The sending host passes the datagram to its adapter, as usual. However, it is not able to indicate an appropriate destination LAN address. Even if known, the MAC address of the destination cannot be used in this case: none of the adapters on LAN 1 would bother to pass the IP datagram up to its network layer, since the frame's destination address would not match the LAN address of any adapter on LAN 1. And the datagram would die. Indeed, the route of the datagram is decided at network layer. It has to pass through the router R, that will forward it to the LAN2. Therefore, the MAC address that has to be used is the one of the next step, that is the one of the interface on LAN1 of R. In R the packet is passed up to the network layer, where the next routing step is considered. When in the LAN2 (e.g. at the interface of R on LAN2) R uses ARP to get the destination physical layer address. Finally, R creates the frame containing source-to-destination IP datagram sends to destination.

Proxy ARP

- Proxy ARP: a host answers ARP requests on behalf of others
 - example: `sic500cs` for PPP connected computers
 - manual configuration of `sic500cs`



69

Proxy ARP is a trick used in special situations, typically:

- on modem lines
- when you want to interconnect 2 subnets while using only one subnet prefix.

It requires a manual configuration and causes a single point of failure.

Reverse ARP has nothing to do with ARP; the purpose of Reverse ARP is to find the IP address corresponding to a MAC address (eg. when booting a diskless station). It is now superseded by protocols like DHCP.

ICMP: Internet Control Message Protocol

| | | | | |
|--|-------------|-------------|--------------------|---|
| <ul style="list-style-type: none"> ▪ Used by hosts, routers, gateways to communication network-level information <ul style="list-style-type: none"> ▪ error reporting: unreachable host, network, port, protocol ▪ echo request/reply (used by ping) ▪ Network-layer "above" IP: <ul style="list-style-type: none"> ▪ ICMP msgs carried in IP datagrams ▪ ICMP message: type, code plus first 8 bytes of IP datagram causing error | <u>Type</u> | <u>Code</u> | <u>description</u> | |
| | | 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 | router advertisement |
| | | 10 | 0 | router discovery |
| | | 11 | 0 | TTL expired |
| | 12 | 0 | bad IP header | |

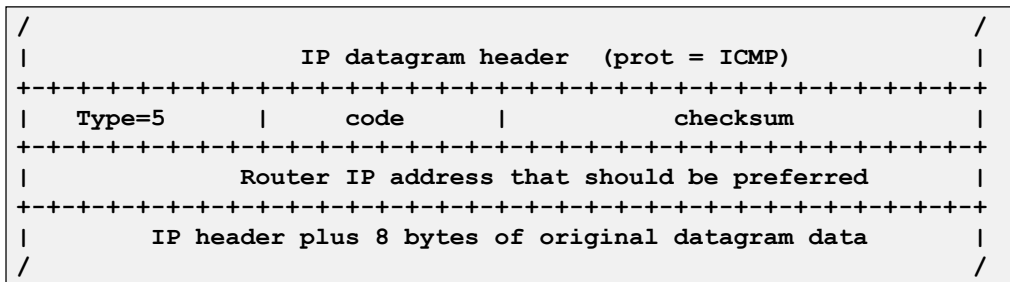
70

The most typical use of ICMP is for error reporting. ICMP is often considered part of IP, but architecturally lies just above IP, as ICMP messages are carried inside IP packets. That is, ICMP messages are carried as IP payload, just as TCP or UDP segments are carried as IP payload. ICMP messages have a type and a code field, and also contain the first eight bytes of the IP datagram that caused the ICMP message to be generated in the first place (so that the sender can determine the packet that caused the error). The well-known `ping` program sends an ICMP type 8 code 0 message to the specified host. The destination host, seeing the echo request, sends back a type 0 code 0 ICMP echo reply. Also `Traceroute` also uses ICMP messages.

ICMP Redirect

- Sent by router to source host to inform source that destination is directly connected
 - host updates the routing table
 - ICMP redirect can be used to update the router table (eg. `in-inj` route to LRC?)

ICMP Redirect Format



71

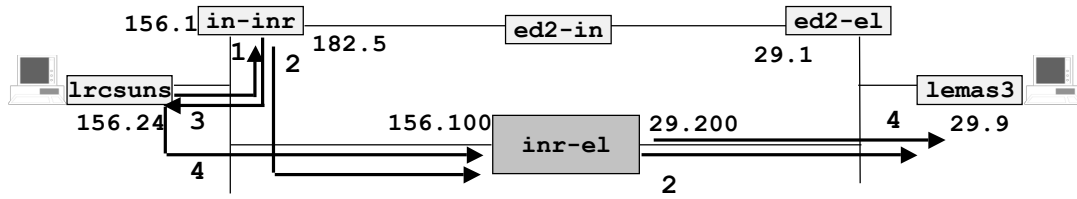
General routing principle of the TCP/IP architecture:

host have minimal routing information

learn host routes from ICMP redirects

routers have extensive knowledge of routes

ICMP Redirect example



| | dest IP addr | srce IP addr | prot | data part |
|----|----------------|----------------|------|---|
| 1: | 128.178.29.9 | 128.178.156.24 | udp | xxxxxxx |
| 2: | 128.178.29.9 | 128.178.156.24 | udp | xxxxxxx |
| 3: | 128.178.156.24 | 128.178.156.1 | icmp | type=redir code=host cksum 128.178.156.100 xxxxxxx (28 bytes) |
| of | | | | 1) |
| 4: | 128.178.29.9 | 128.178.156.24 | udp | |

ICMP Redirect example (cont'd)

After 4

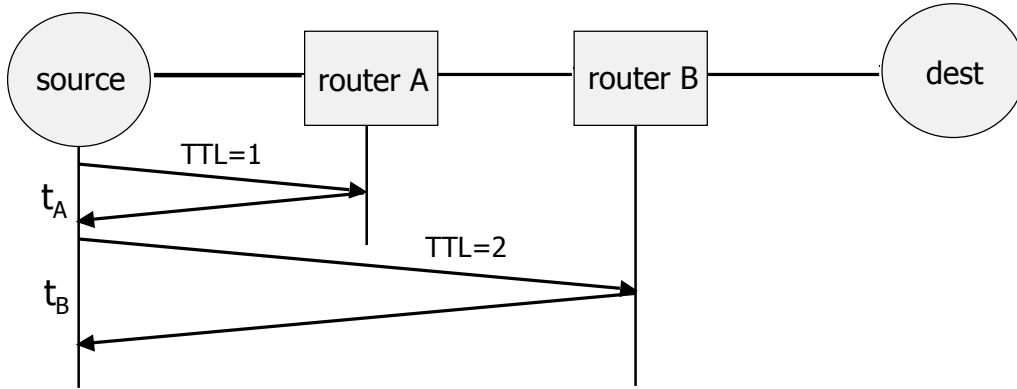
```
lrcsuns$ netstat -nr
Routing Table:
  Destination          Gateway                Flags    Ref    Use  Interface
-----
127.0.0.1              127.0.0.1             UH       0  11239  lo0
128.178.29.9           128.178.156.100      UGHD     0     19
128.178.156.0          128.178.156.24       U        3  38896  1e0
224.0.0.0              128.178.156.24       U        3     0  1e0
default                128.178.156.1        UG       0  85883
```

73

Tools that use ICMP

- *ping*
 - ICMP *Echo request*
 - wait for *Echo reply*
 - measure RTT
- *traceroute*
 - IP packet with TTL = 1
 - wait for ICMP *TTL expired*
 - IP packet with TTL = 2
 - wait for ICMP *TTL expired*
 - ...

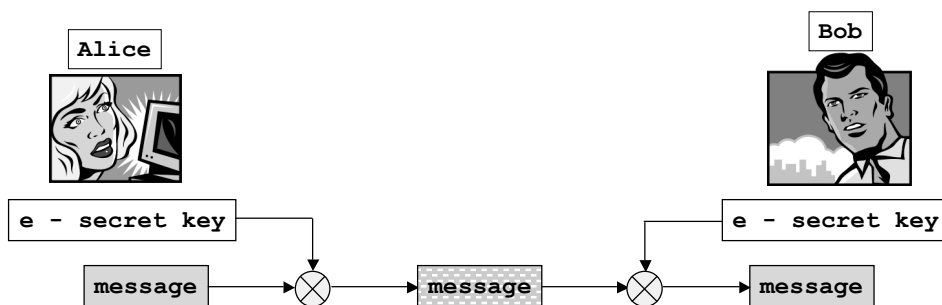
Traceroute



IPsec - secure IP communication

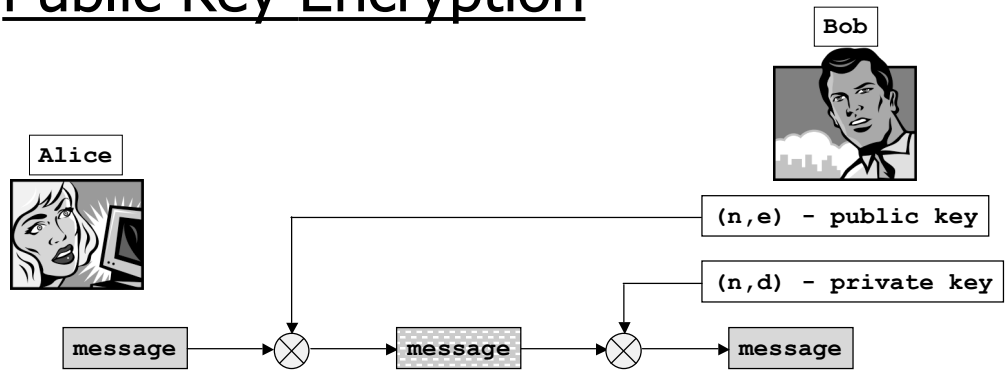
- Key exchange
 - based on Diffie-Hellman (form a shared secret using public keys)
 - secret symmetrical keys used for authentication and encryption
- Authentication
 - AH (Authentication Header): encrypted hash (MD5)
- Encryption
 - ESP (Encapsulating Security Payload): 3DES
- Similar to ssh tunnel, but all upper protocols may benefit from secure communication

Secret Key Encryption



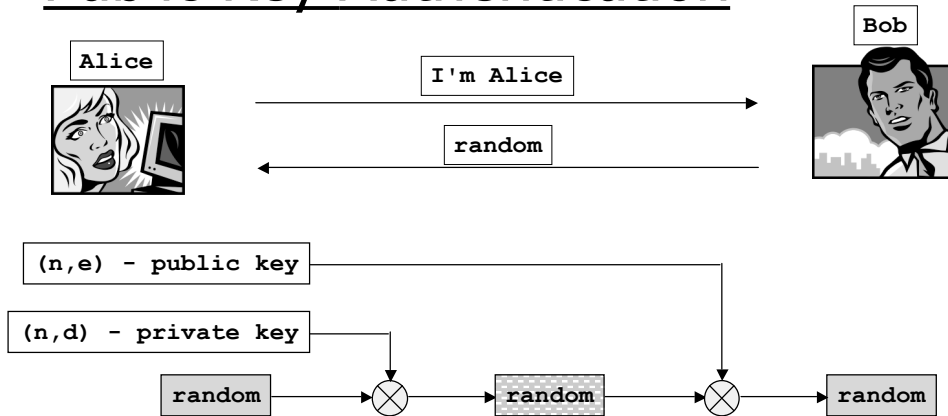
- Secret key encryption (DES, 3DES,...)
 - encrypted message $c = f(e, m)$
 - decrypted message $m = f^{-1}(e, c)$
- Must exchange the key
- Efficient encryption

Public Key Encryption



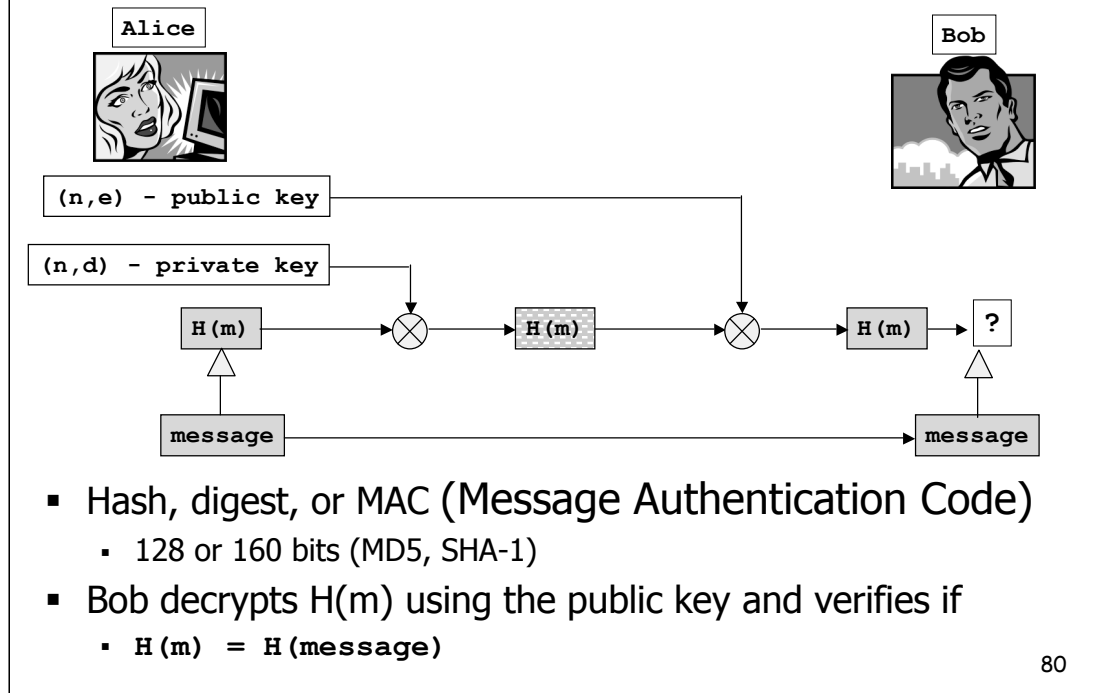
- RSA encryption
 - encrypted message $c = (m^e \bmod n)$
 - decrypted message $m = (c^d \bmod n)$
- Key property
 - $(m^e)^d \bmod n = m$
- Slow

Public Key Authentication

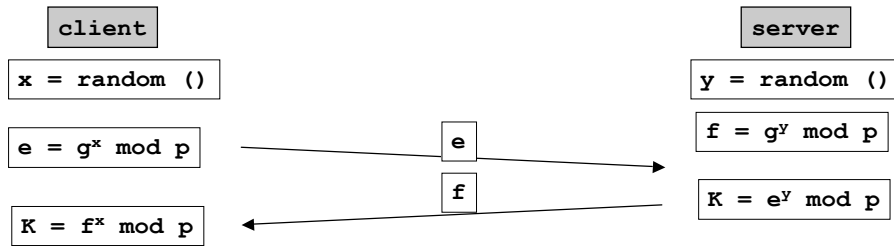


- Authentication
 - random challenge (nonce), used only once
- Bob verifies
 - $(r^d)^e \bmod n = r$

Integrity - digital signature



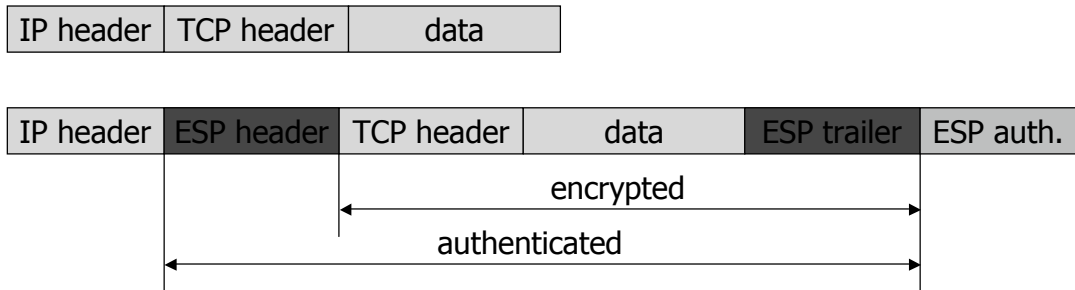
Diffie-Hellman key exchange



- Known parameters

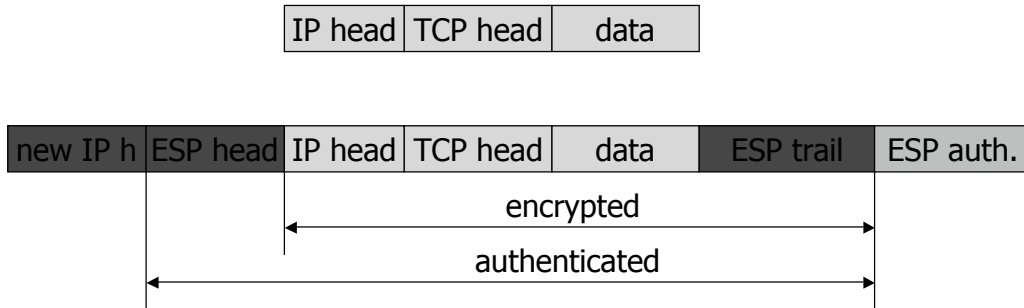
- **g - generator** (e.g. $g = 2$)
- **p - large prime**
 - e.g. $2^{1024} - 2^{960} - 1 + 2^{64} \lfloor \text{floor}(2^{894} \pi + 129093) \rfloor$
- $1 < x, y < (p - 1) / 2$
- $K = (g^x \text{ mod } p)^y \text{ mod } p = (g^y \text{ mod } p)^x \text{ mod } p$

IPsec



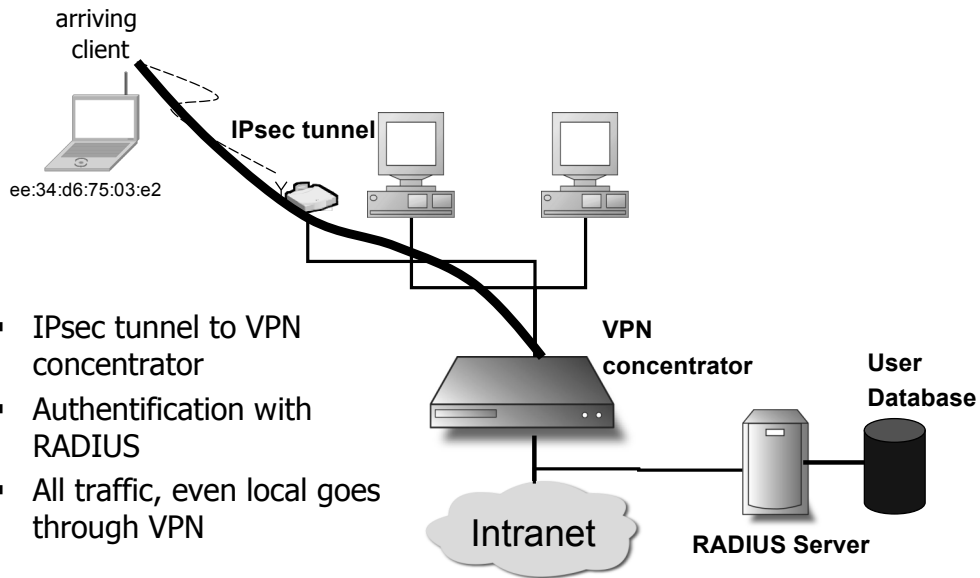
- Transport mode
 - only IP payload is encrypted

IPsec



- Tunnel mode
 - new IP header

VPN - Virtual Private Network



84

Summary

- The network layer transports packets from a sending host to the receiver host.
- Main components:
 - addressing
 - packet forwarding
 - routing protocols and routers (or how a router works)
- Routing protocols will be seen later in the advanced course
- Internet network layer
 - connectionless
 - best-effort

Question

- Ensimag has given the range of IP addresses 128.178.197/24 to the student association. The student network manager has to assign addresses to other students: they need 12 subnetworks with 12 hosts each.
 - What is the subnet mask she has to define?
 - How many hosts there can be at most on each subnetwork?
 - What is the first and the last address of subnetwork 12?
 - What is the broadcast address on subnetwork 12?

Question

- Ensimag has given the range of IP addresses 128.178.197/24 to the student association. The student network manager has to assign addresses to other students: they need 12 subnetworks with 12 hosts each.
 - What is the subnet mask she has to define?
 - How many hosts there can be at most on each subnetwork?
 - What is the first and the last address of subnetwork 12?
 - What is the broadcast address on subnetwork 12?

and the winners are:

- 1 point
 - Fabien Devauchelle
 - Mathieu Chondroyannis
 - Vincent Maingot
 - Gaël Bréard
 - Aissam Dannoun
- 0.75
 - Reda Bouallou
 - Antoine Markarian
 - Yann Le Mounier
 - Atman Choujaa
 - Julien Barthes
 - Cyril Chambeyron

88

Answers

- What is the subnet mask she has to define?
 - 11111111.11111111.11111111.11110000
 - 255.255.255.240
- How many hosts there can be at most on each subnetwork?
 - $14 = 16 - 2$
- What is the first and the last address of subnetwork 12?
 - 10000000.10110010.11000101.11100001
 - $192+1=193$, 128.178.197.193
 - 10000000.10110010.11000101.11101110
 - $192+14=206$, 128.178.197.206
- What is the broadcast address on subnetwork 12?
 - 10000000.10110010.11000101.11101111
 - 128.178.197.207

89

Answers

- What is the subnet mask she has to define?
 - 11111111.11111111.11111111.11110000
 - 255.255.255.240
- How many hosts there can be at most on each subnetwork?
 - $14 = 16 - 2$
- What is the first and the last address of subnetwork 12?
 - 10000000.10110010.11000101.10110001
 - $176+1=177$, 128.178.197.177
 - 10000000.10110010.11000101.10111110
 - $176+14=190$, 128.178.197.190
- What is the broadcast address on subnetwork 12?
 - 10000000.10110010.11000101.10111111
 - 128.178.197.191

90