Advanced TCP/IP

Table of contents

Chapter 1

Overview of the TCP/IP Protocol Suite

Chapter 1. Overview of TCP/IP Protocol Suite

- **After completing this chapter the student will be able to identify and describe in detail the following protocols:**
  - **Internet Protocol (IP)**
  - **Address Resolution Protocol (ARP)**
  - **Internet Control Message Protocol (ICMP)**
  - **Transmission Control Protocol (TCP)**
  - **User Datagram Protocol (UDP)**

ERICSSON

History of the Internet

1969	ARPANET	1989	No. of Hosts > 100,000
1972	Telnet	1990	ARPANET replaced
1973	FTP	1991	Gopher
1983	TCP/IP exclusively used by ARPANET ARPANET split in 2	1992	CERN – WWW Upgrade to T3 No. of Hosts > 1,000,000
1985/1986	NSFNET backbone formed	1993	Mosaic
1987	Upgrade to T1 No. of Hosts > 10,000	1994	Communities
		2000	… … … … … … …

Rev. B

Ericsson Systems Expertise

1969 - An experimental network called ARPANET was created by the US Department of Defence Advanced Research Projects Agency (ARPA). ARPANET originally connected four universities. It enabled scientists to share information and resources across long distances while providing a test-bed for emerging network technologies.

1972 - An application called Telnet was developed by the National Centre for Supercomputing Applications (NCSA).This application enabled a user to login to a remote computer.

1973 - File Transfer Protocol (FTP) was introduced. This application standardised the transfer of files between networked computers.

First international connections to ARPANET : England and Norway.

1983 - TCP/IP suite of networking protocols became the only set of protocols to be used on ARPANET. This set a standard for other networks.

ARPANET split into 2 networks ARPANET and MILNET - a military network.

Desktop workstations came into being , many running a Berkley Systems UNIX operating system which included IP networking software.

1985/1986 - The National Science Foundation (NSF) of America connected the nation's six supercomputing centres together. This network was called NSFNET or the NSFNET backbone. This backbone had a capacity of 56kbps.

1987 - NSFNET network links were upgraded to T1 speeds (1.544Mbps).

Merit Network Inc.- a non-profit Internet research and development corporation - signed an agreement with NSF to manage it's network.

The number of internet hosts exceeded 10,000.

1989 - The number of internet hosts exceeded 100,000.

1990 - ARPANET was replaced .

Merit Network Inc., IBM and MCI formed an organisation called Advanced Networks and Services (ANS) which was responsible for operating backbone routers and a Network Operation Centre (NOC).

1991 - Gopher, a hierarchical, menu-based method for locating and providing information on the internet, was released by the University of Minnesota.

1992 - The World Wide Web (WWW), an internet-based communication system, was released by the European Laboratory for Particle Physics (CERN). This changed the way in which information could be organised, presented and accessed on the Internet through the use of HyperText Transfer Protocol (HTTP) and hyperlinks.

The NSFNET backbone links were upgraded to T3 speeds (44.736 Mbps)

The number of internet hosts exceeded 1,000,000.

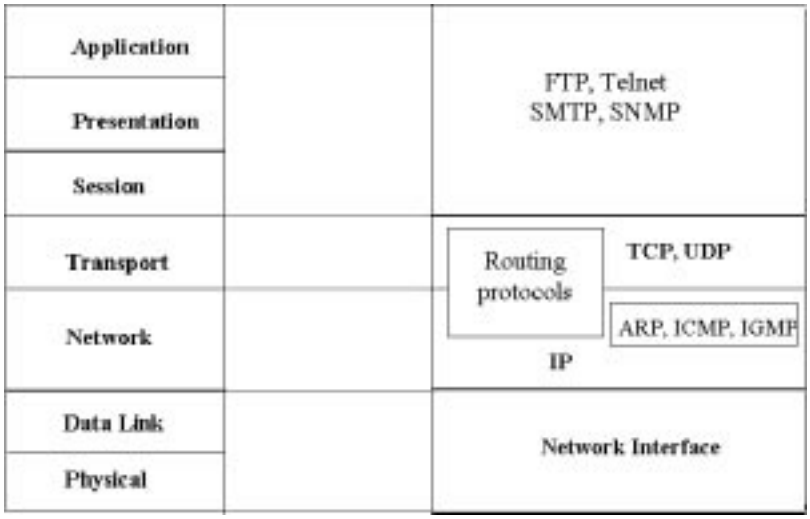
1993 - Mosaic Internet browser released. WWW grew at a staggering rate.

1994 - Communities began to get wired up to the Internet directly e.g. US White House.

2000 - ……..

ERICSSON

Internet Protocol Suite & the OSI Reference Model



1/038 13 LZUBB 108 101/3 Rev. B Ericsson Systems Expertise

Transmission Control Protocol/Internet Protocol (TCP/IP) is not a single protocol; it refers to a family or suite of protocols. The suite consists of a four-layer model.

Network Interface Layer

The Network Interface Layer is equivalent to the combination of the Physical and Data Link Layers in the OSI model. It is responsible for formatting packets and placing them onto the underlying network. All common Data Link protocols support TCP/IP.

Internet Layer

The Internet Layer is equivalent to the Network Layer in the OSI model. It is responsible for network addressing. The main protocols at this layer are: Internet Protocol (IP), Address Resolution Protocol (ARP), Reverse Address Resolution Protocol (RARP), Internet Control Message Protocol (ICMP), and Internet Group Management Protocol (IGMP).

The Transport Layer

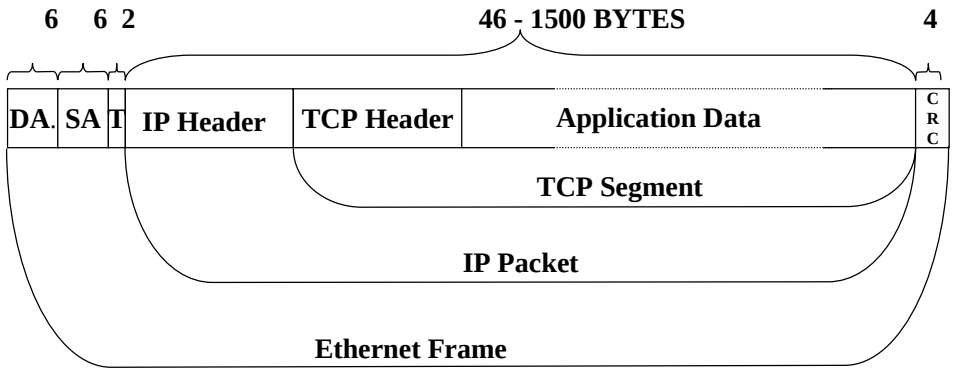
The Transport Layer is equivalent to the Transport Layer in the OSI model. The Internet Transport layer is implemented by TCP and the User Datagram Protocol (UDP). TCP provides connection-oriented data transport, while UDP operation is connectionless.

The Application Layer

The Application Layer is equivalent to the top three layers, (Application, Presentation and Session Layers), in the OSI model. The Application Layer is responsible for interfacing between user applications and the Transport Layer. Applications commonly used are File Transfer Protocol (FTP), Telnet, Simple Network Management Protocol (SNMP), and Domain Name system (DNS).

ERICSSON

Encapsulation in an Ethernet Frame



DA = Destination Address, e.g. 00-80-37-12-34-56
SA = Source Address
T = Type, e.g. 0800 = IP, 6003 = DECnet
CRC = Cyclic Redundancy Check
Ethernet Frame Size - Minimum 64 Bytes Maximum 1518 Bytes

If an IP packet is to be transmitted across a physical network that does not understand its format, the packet must be encapsulated. The entire IP packet is placed in the data portion of a data link frame.

An Ethernet frame consists of a header, a trailer and a data portion. In the example above, the data portion contains an IP packet. The IP packet is said to be encapsulated in an Ethernet frame.

The IP packet itself consists of a header and a data portion. The data portion of the IP packet contains a TCP segment.

The TCP segment consists of a TCP header and the actual application data.

An Ethernet frame is always between 64 and 1,518 bytes in size. Eighteen bytes are required for the Ethernet header and trailer. The data portion is therefore between 46 and 1,500 bytes.

Note: when the Ethernet frame is sent over the cable, every device connected to the cable receives the frame and checks if the destination address (called the physical address or [MAC - Medium Access Control] - address) matches its own address. If so, the frame is processed, the checksum control, header and trailer are dropped and data is passed to the higher layers. If not, the frame is destroyed.

ERICSSON

Internet Protocol (IP)

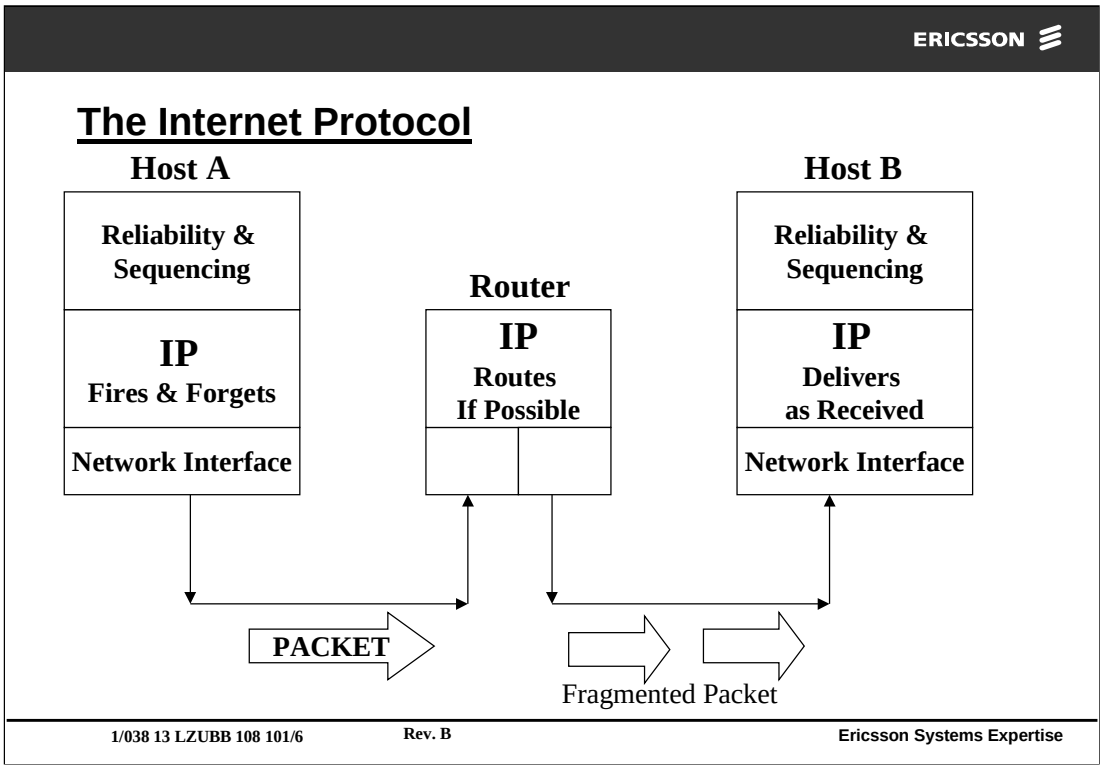
- **Provides logical 32-bit network addresses**
- **Routes data packets**
- **Connectionless protocol**
  - **No session is established**
- **“Best effort” delivery**
- **Reliability is responsibility of higher-layer protocols and applications**
- **Fragments and reassembles packets**

IP is a connectionless protocol primarily responsible for addressing and routing packets between network devices.

Connectionless means that a session is not established before exchanging data. IP is unreliable in that delivery is not guaranteed. It makes a “best effort” attempt to deliver a packet. Along the way a packet might be lost, delivered out of sequence, duplicated or delayed.

An acknowledgement is not required when data is received. The sender or receiver is not informed when a packet is lost or out of sequence. The acknowledgement of packets is the responsibility of a higher-layer transport protocol, such as TCP.

IP is also responsible for fragmenting and reassembling packets. A large packet must be divided into smaller pieces when the packet has to traverse a network that supports a smaller packet size. For example, an IP packet on a FDDI network could be up to 4,040 bytes long. If this packet then needs to traverse an Ethernet network, it must be split up into IP packets which are a maximum of 1,500 bytes long.



IP delivers its packets in a connectionless mode. It does not check to see if the receiving host can accept data and it does not keep a copy in case of errors. IP is therefore said to “fire and forget”.

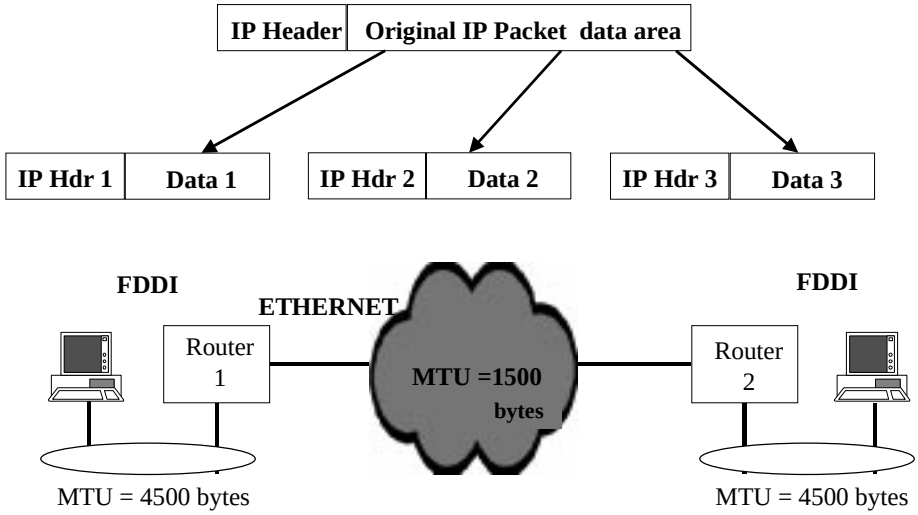
When a packet arrives at a router, it forwards the packet only if it knows a route to the destination. If the router does not know the destination it drops the packet. The router does not send any acknowledgements to the sending device.

A router checks the checksum, if it is not correct the packet is dropped. It also decreases the Time-To-Live (TTL), if this value is zero then the packet is dropped. If necessary it fragments larger packets into smaller ones and sets Flags and Fragment Offset fields accordingly.

Finally, a new checksum is generated due to possible changes in TTL, flags and Fragment Offset and then the packet is forwarded.

ERICSSON

Fragmentation



1/038 13 LZUBB 108 101/7 Rev. B Ericsson Systems Expertise

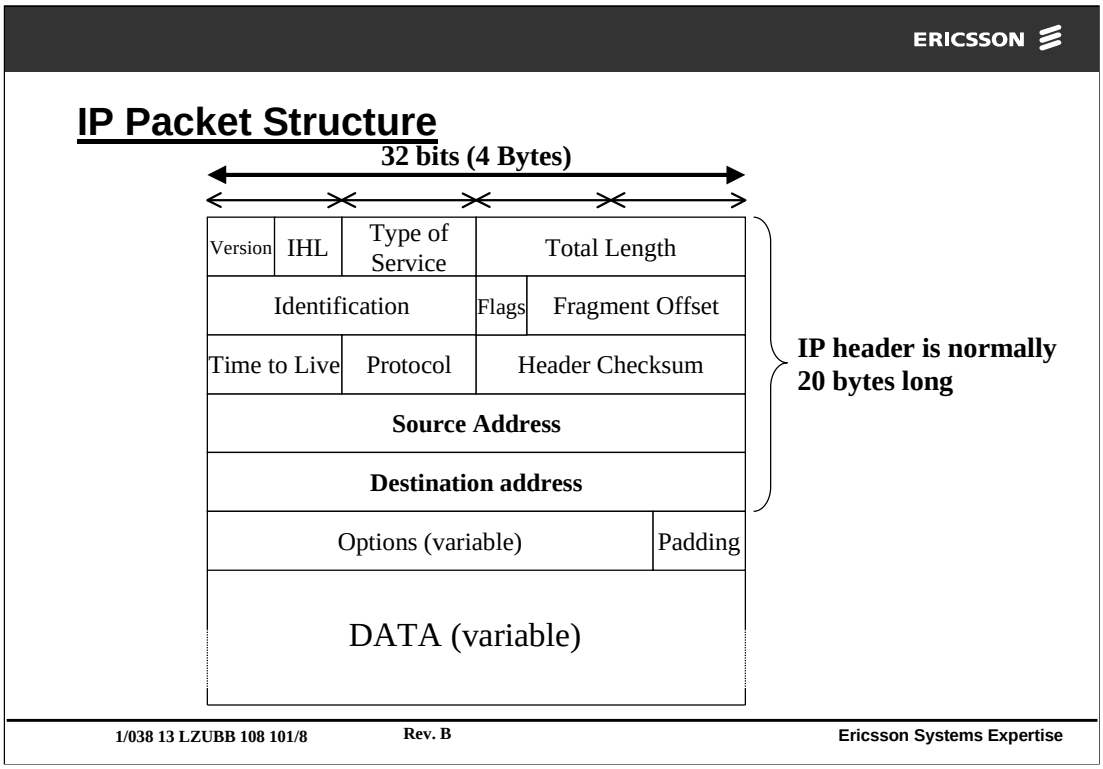
Each physical network imposes some maximum transmission (the Maximum Transfer Unit) size on the packets that may be sent over it. When the size of the packet exceeds the MTU of the network on the outgoing interface, it must be broken into smaller packets, each of which carries a portion of the original data. This process is called Fragmentation.

The fragmented IP packets have data copied from the original packet into their data area. Each fragment contains an IP header that duplicates the original header except for the information in the flags and offset fields. They are treated as normal IP packets while being transported to their destination. Therefore the fragment packets may take different routes to their final destination.

When the fragment packets arrive at their destination, the destination host must join the fragments together again before processing the original packet in the normal way.

However, if one of the fragments gets lost, the complete IP packet is considered lost. This is because IP does not provide any acknowledgement mechanism. The remaining fragments will simply be discarded by the destination host.

Note: if a packet has a flag set to “don’t fragment” and the router decides to send this packet over a medium which does not support the size of the packet, then the packet is dropped.



Version (4 bits): This specifies the version of the IP protocol and hence the format of the IP header being used. The current protocol version is 4 (IPv4); the new version is 6 (IPv6).

IHL, Internet Header Length (4 bits): This is the length of the header in 32-bit words. The minimum value is five, which is the most common header. Thus the header must be at least 20 bytes long.

Type of Service (8 bits): This is an indication of the quality of service requested for the IP packet. It specifies reliability, precedence, delay and throughput parameters.

Total length (16 bits): This is the total packet length, including header and data, in bytes.

Identification (16 bits): This is a unique number assigned by the sending device to aid in reassembling a fragmented packet. Its primary purpose is to allow the destination device to collect all fragments from a packet, since they will all have the same identification number.

Flags (3 bits): These provide the fragmentation control fields.The first bit is not used and is always 0. If the second bit is 0, it means “May fragment”. If the second bit is 1, it means “Don’t fragment”. If the third bit is 0, it means “Last fragment”. If the third bit is 1, it means “More fragments”.

Fragment Offset (13 bits): This is used with fragmented packets to aid in reassembling the full packet. The value is the number of 8-byte pieces (header bytes are not counted) that are contained in earlier fragments. In the first fragment or in a unique fragment, this value is always zero.

Time to Live (8 bits): This contains the time, in seconds, that the packet is allowed to remain on an internetwork. Each IP device that the packet passes through will decreases the value by the time it takes it to process the IP header. All routers must decrease this value by a minimum of one. If the value drops to zero the packet is discarded. This guarantees that packets cannot travel around an IP network in a loop, even if routing tables become corrupt.

Protocol (8 bits): This indicates the higher level protocol to which IP should deliver the data in the packet, for example, UDP is 17 and TCP is 6.

Header Checksum (16 bits): This is a checksum on the header only, which ensures integrity of header values. The sending IP device performs a calculation on the bits in the IP header, excluding the header checksum field, and places the result in the header checksum field. The receiving device performs the same calculation and compares the result with the value in the header checksum field. If they are different then an error has occurred and the IP packet is discarded.

Source Address (32 bits): This is the 32-bit IP address of the sending device.

Destination Address (32 bits): This is the 32-bit IP address of the receiving device.

Options (variable): These are not required in every packet. They are mainly used for network testing or debugging.

Data (variable): The total length of the data field plus header is a maximum of 65,535 bytes.

ERICSSON

Address Resolution Protocol (ARP)

- **A source must know a destination's hardware address before it can send an IP packet directly to it**
- **ARP is the mechanism that maps IP to hardware addresses.**
- **ARP uses a local broadcast to obtain a hardware address**
- **ARP stores mappings in cache for future use**

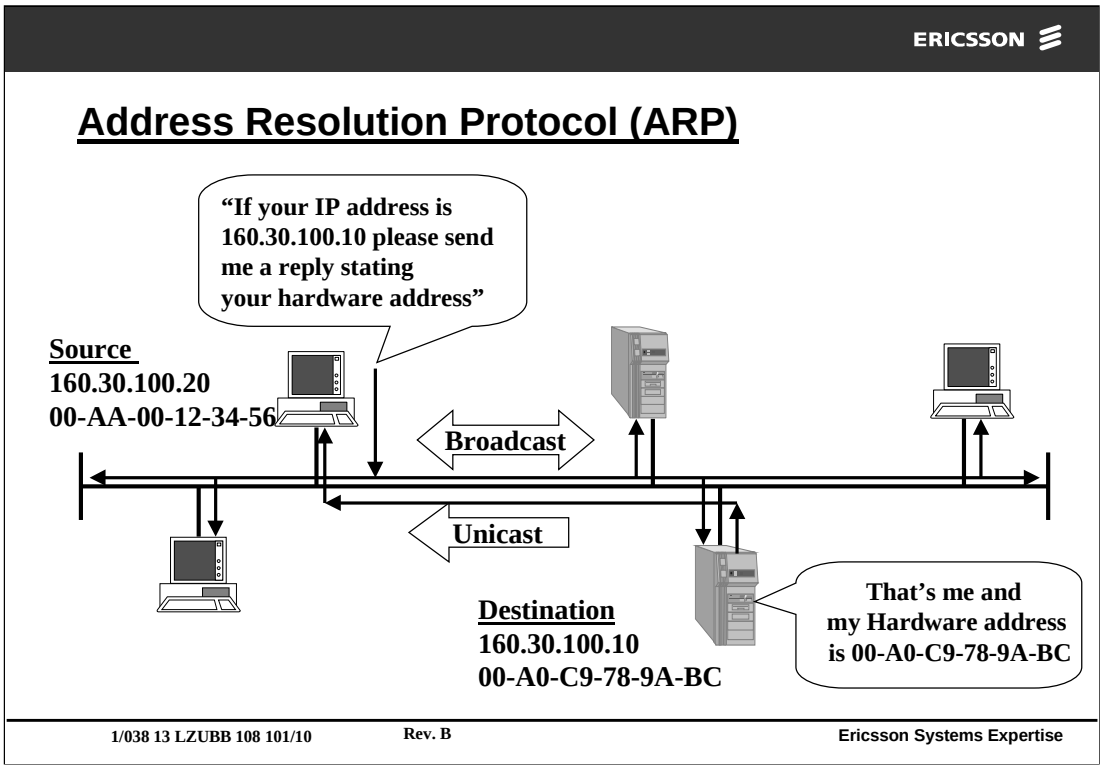
Network devices must know each other's hardware address to communicate on a network. Address resolution is the process of mapping a host's IP address to its hardware address.

The Address Resolution Protocol (ARP) is responsible for obtaining hardware addresses of TCP/IP devices on broadcast-based networks.

ARP uses a local broadcast of the destination IP address to acquire the hardware address of the destination device.

Once the hardware address is obtained, both the IP address and the hardware address are stored as one entry in the ARP cache.

The ARP cache is always checked for an IP address/hardware address mapping before initiating an ARP request broadcast.



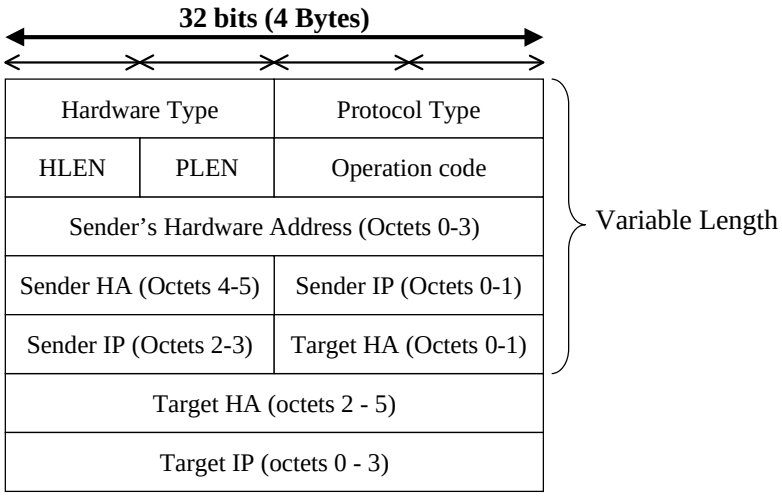
The source device knows its own IP and hardware address and the IP address of the device it wants to send the information to.

It checks its existing ARP cache for the hardware address of the destination host. If no mapping is found, the source builds an ARP request packet, looking for the hardware address to match the IP address. The ARP request is a broadcast so all local devices receive and process it. Each device checks for a match with its own IP address. The destination device determines that there is a match and sends an ARP reply directly to the source device with its hardware address. Both devices update their ARP cache with the IP address/hardware address mapping of the other device. From then on the devices can communicate directly with each other.

If devices do not communicate with each other after a period of time they will clear the entry from their ARP caches.

ERICSSON

ARP Packet Structure



Hardware Type (16 bits): This specifies the hardware interface type, for example, Ethernet has a value of 1.

Protocol type (16 bits): This specifies the higher-level protocol whose address needs to be mapped onto the hardware, for example, IP - 0800.

HLEN, Hardware address length (8 bits): This specifies the length in bytes of the hardware address in this packet, for example, Ethernet - 6.

PLEN, Protocol address length (8 bits): This specifies the length in bytes of the protocol address in this packet. For IP this is four.

Operation code (16 bits): This specifies whether this is an ARP request (1) or an ARP reply (2).

Sender's hardware address (48 bits): This contains the hardware address of the sender (the ARP requester).

Sender's IP address (32 bits): This contains the protocol address of the sender (the ARP requester).

Target's hardware address (48 bits): This contains the hardware address of the target (the ARP responder).

Target's IP address (32 bits): This contains the protocol address of the sender (the ARP responder).

ERICSSON

Reverse Address Resolution Protocol

- **Reverse ARP is the mechanism that maps hardware addresses to the IP address**
- **RARP protocol allows a newly booted machine to broadcast its Ethernet address**
- **The RARP server sees this request and sends back the corresponding IP address**

ARP solves the problem of mapping a host's IP address to its hardware address, but sometimes the reverse problem has to be solved.

Reverse ARP (RARP) is used when we are given the hardware address, for example an Ethernet address, but not its corresponding IP address.

The RARP protocol allows a newly booted device to broadcast its Ethernet address and say: "My 48-bit Ethernet address is 00-A0-C9-78-9A-BC. Does anyone know my IP address?". The RARP protocol uses the same message format as ARP.

The server sees this request, looks up the Ethernet address in its configuration files and sends back the corresponding IP address. This type of server is known as a RARP server.

To prevent multiple servers from sending a reply simultaneously, causing collisions, a primary server may be designated for each host wishing to use RARP. This server replies immediately and all non-primary servers simply listen and note the time of the request

If the primary server is unavailable, the originating node will timeout and re-broadcast the RARP request. The non-primary servers respond when they hear a copy of the request within a short time after the original broadcast. This prevents unnecessary transmissions

Example :Printers use RARP to get an IP address.

Note: RARP requests stay within the local LAN, so the servers must reside there also.

ERICSSON

Internet Control Message Protocol (ICMP)

- **Reports errors and sends control messages on behalf of IP**
- **ICMP messages are encapsulated within an IP packet**
- **One of the most frequently used debugging tools uses ICMP**
  - **It tests whether a destination is reachable and responding, by sending ICMP echo requests and receiving back ICMP echo replies**
  - **It carries out this test by using the "PING" command**
- **Echo Request and Reply message format**

IP Header.....		
Type	Code	Checksum
Identifier		Sequence Number
Optional Data		

ICMP reports errors and sends control messages on behalf of IP. ICMP does not attempt to make IP a reliable protocol. It merely attempts to report errors and provide feedback on specific conditions. ICMP messages are carried as IP packets and are therefore unreliable.

If an IP device is sending packets to another IP host at a rate that is too fast for the routers to handle, the router can send an ICMP "source quench" message asking it to slow down.

Type (8 bits): This specifies the type of ICMP message, for example,

0 is an echo reply, 3 is destination unreachable, 4 is a source quench, 5 is a redirect, 8 is an echo request.

Code (8 bits): This contains an error code that further describes the problem. For example, in a destination unreachable message: 0 means network unreachable, 1 means host unreachable, 2 means protocol unreachable.

Checksum (16 bits): This is a checksum carried out on the ICMP header only.

Identifier and **Sequence number** (16 bits each): These are used by the sender to match replies to requests.

Optional Data: This field contains information to be returned to the sender. An Echo reply always returns exactly the same data as was received in the Echo request.

ERICSSON

Transmission Control Protocol (TCP)

- **Connection-oriented**
- **Provides logical connections between pair of processes:**
  - **These are uniquely identified using sockets**
  - **Socket = IP address & port number, e.g. FTP is port 21**
- **End-to-End reliable delivery**
- **Implements Flow Control**

TCP is a reliable, connection-oriented delivery service. Connection-oriented means that a session must be established before devices can exchange data.

TCP processes or applications communicate with each other by having both the sending and receiving device create end points, called sockets. An application creates a socket by specifying three items: the IP address of the device, the transport protocol (TCP or UDP) and the port the application is using. Each socket has a socket number (address) consisting of the IP address of the device and a 16-bit number called a port. A port is used by transport protocols to identify which application protocol or process they must deliver incoming messages to. A port can use any number between 0 and 65,536. All “well-known” port numbers are below 256, for example, FTP is port 21, Telnet is port 23 and DNS is port 53.

TCP views the data stream as a sequence of octets or bytes that is divided into segments for transmission. Each segment travels across the network in a single IP packet. Reliability is achieved by assigning a sequence number to each segment transmitted. If a TCP segment is broken into smaller pieces, the receiving device knows whether all pieces have been received. An acknowledgement is used to verify that the data was received. For each segment sent, the receiving device must return an acknowledgement (ACK) within a specified period. If an ACK is not received the data is retransmitted.

End-to-end flow control is implemented as follows: if the sending device is transmitting data faster than the receiving device is processing it, the receiver will not send back an acknowledgement until it has sufficient buffer space to accommodate more data. This prevents the sender sending any new data until the receiver is ready.

ERICSSON

Transmission Control Protocol (TCP)

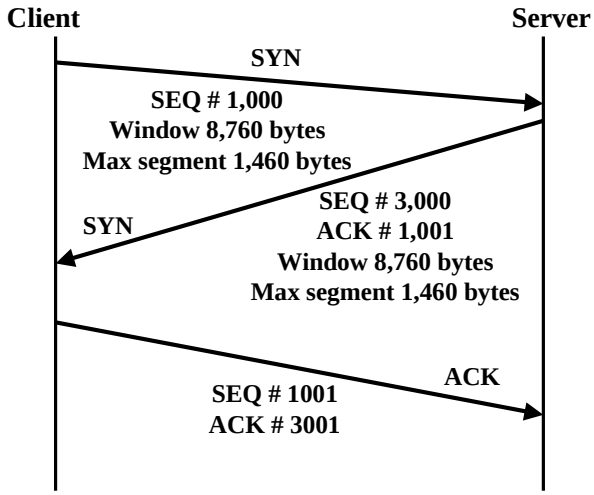
- **Units of data transferred between two devices running TCP software are called “segments”**
- **Segments are exchanged to do the following:**
  - **Establish a connection**
  - **Agree window size**
  - **Transfer data**
  - **Send acknowledgements**
  - **Close connection**

A TCP session is initialised through a three-way handshake. During this process the two communicating devices synchronise the sending and receiving of segments, inform each other of the amount of data they are able to receive at once (window size and segment size), and establish a virtual connection.

TCP uses a similar handshake process to end a connection.

ERICSSON

Establishing a TCP Connection



1/038 13 LZUBB 108 101/16 Rev. B Ericsson Systems Expertise

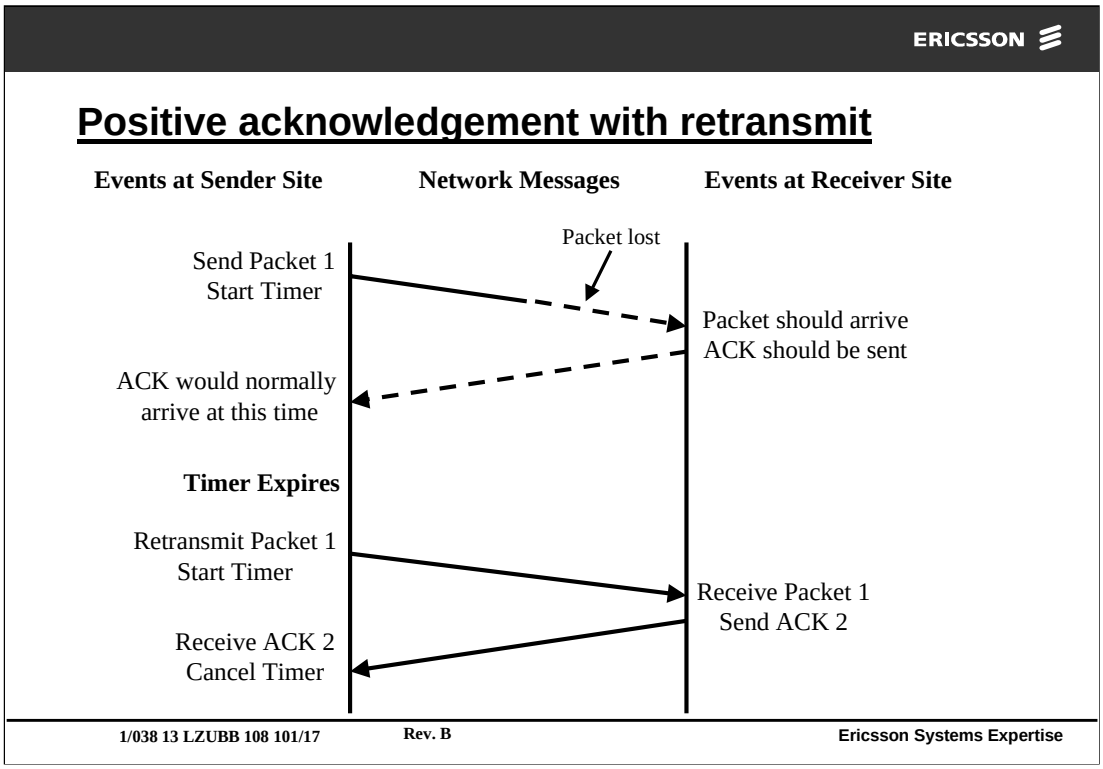
Establishing a TCP Connection

To establish a connection TCP uses a three-way handshake.

The client's TCP software generates a sequence number (1,000 in this example). The client requests a session by sending out a segment with the synchronisation (SYN) flag set to on. The segment header also includes the sequence number, the size of its receive buffer (window size) and the size of the biggest data segment it can handle.

The server acknowledges (ACK) the request by sending back a segment with the synchronisation (SYN) flag set to on. The segment header contains the server's own start-up sequence number and acknowledgement as well as the number of the next segment it expects to receive from the client. The segment header also includes the size of the server's receive buffer (window size) and the size of the biggest data segment it can handle.

The client sends back an acknowledgement of the server's start-up sequence segment. It does this by sending the sequence number of the next segment it expects to receive. TCP uses a similar handshake to end a connection.



Computers do not all operate at the same speed. Data overruns can occur when a computer sends data across a network faster than the destination can absorb data. Consequently data can be lost.

Several techniques are available to provide reliable delivery, and these techniques are known as flow control mechanisms.

A simple form of flow control is positive acknowledgement with retransmission. This technique requires a recipient to communicate with the source, and send back an acknowledgement message when it receives data.

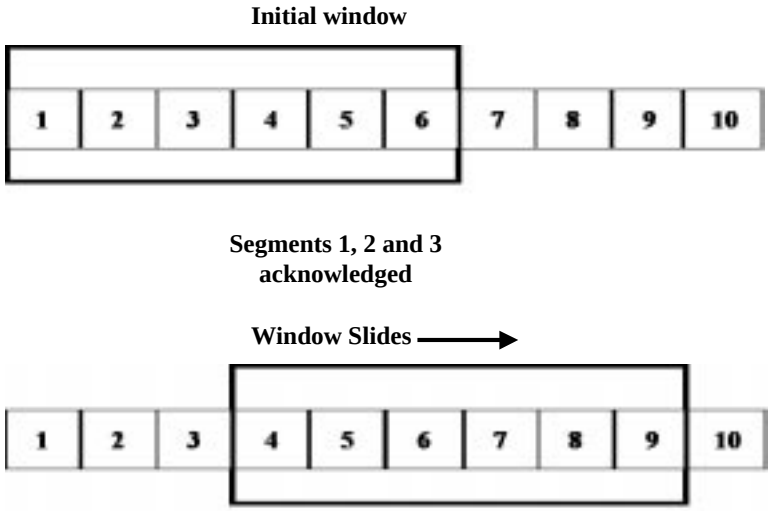
The sender keeps a copy of each packet it sends and waits for an acknowledgement before sending the next packet. The sender also starts a timer when it sends a packet and retransmits the packet if the timer expires before an acknowledgement arrives. The acknowledgement will contain the sequence number that the receiver expects to receive next.

The diagram above shows the events happening when a packet is lost or corrupted. The sender starts a timer after transmitting a packet. When the timer expires, the sender assumes the packet was lost and retransmits it.

Problems can arise when duplicate packets are received. Duplicates can arise when networks experience long delays that cause premature retransmission. Both packets and acknowledgements can be duplicated. To avoid the problem of duplication, positive acknowledgement protocols send sequence numbers back in acknowledgements, so the receiver can correctly associate acknowledgements with packets

ERICSSON

Sliding Window Protocol



1/038 13 LZUBB 108 101/18 Rev. B Ericsson Systems Expertise

With positive acknowledgement with retransmission, the sender transmits a packet and waits for an acknowledgement before transmitting another. So data flows in one direction at any one time. The network is completely idle during times that machines delay responses. As a result, the positive acknowledgement protocol wastes a substantial amount of network bandwidth because it must delay sending a new packet until it receives an acknowledgement for the previous packet.

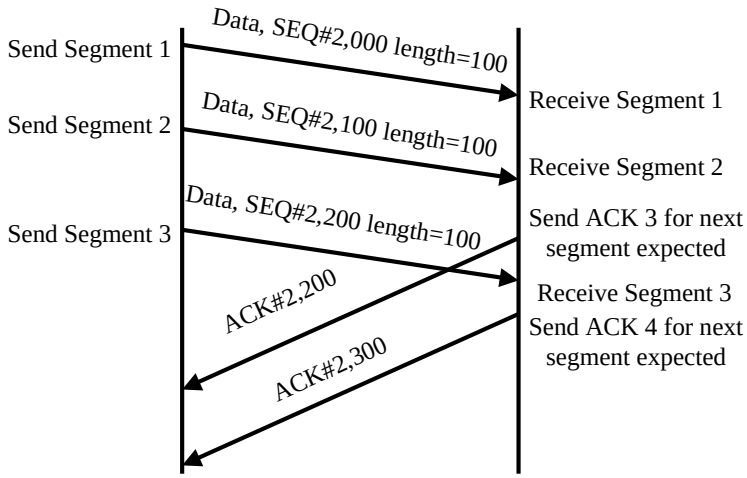
The Sliding Window Protocol (SWP) uses network bandwidth more efficiently. It allows the sender to transmit multiple packets before waiting for an acknowledgement (ACK). The protocol places a small window on the sequence and transmits all packets that lie inside the window. Technically the number of packets that can be unacknowledged at any given time is constrained by the window size and is limited to a small, fixed number.

For example, in an SWP protocol with window size 6, the sender is permitted to transmit 6 packets before it receives an ACK. As the diagram above shows, once the sender receives an acknowledgement for the first three packets inside the window, it slides the window along and sends the next packet. The window continues to slide as long as ACKs are received.

Note: the TCP sliding window mechanism operates at byte level. For example, on an Ethernet network the window size might be defined as 11,680. This means that 11,680 bytes can be transmitted by the sender before it receives any acknowledgement. On an Ethernet network this is the equivalent of eight TCP segments filled to their maximum size, assuming the TCP and IP headers are twenty bytes each.

ERICSSON

Sliding Window Protocol



1/038 13 LZUBB 108 101/20 Rev. B Ericsson Systems Expertise

The performance of the sliding window protocol depends on the window size and the speed at which the network accepts packets. The receiver can choose how much to acknowledge, thus throttling the sender to match its capacity.

The diagram above shows an example of the operation of the sliding window protocol when sending three segments.

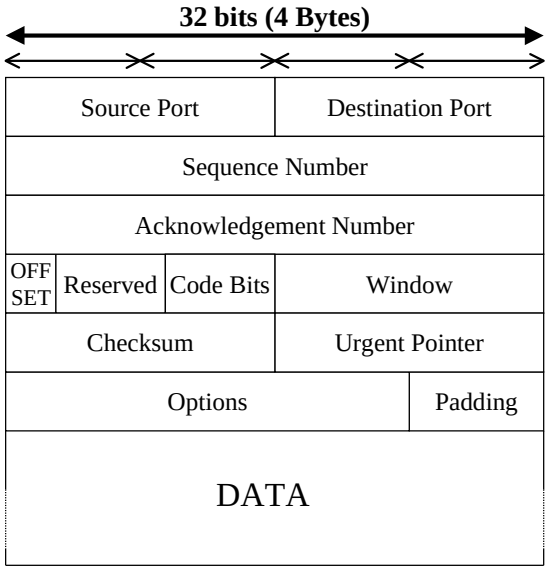
A sliding window protocol keeps a separate timer for each unacknowledged segment. If a segment is lost, the timer expires and the sender retransmits that segment. When the sender slides its window, it moves past all acknowledged segments. At the receiving end, the protocol software keeps an analogous window, accepting and acknowledging segments as they arrive.

Note: in TCP the acknowledgement number sent is the sequence number of the next data byte (not segment or packet) that the receiver is expecting. It is the sum of the last sequence number it received and the length of the data in bytes.

For example, if a device receives a segment with sequence number 2,000 and length 1,000 bytes, it will send back an acknowledgement number of 3,000.

ERICSSON

TCP Packet Structure



1/038 13 LZUBB 108 101/21 Rev. B Ericsson Systems Expertise

Source port (16 bits): This is the TCP port number of the sending device.

Destination port (16 bits): This is the TCP port number of the receiving device.

Sequence number (32 bits): This is the sequence number of the data byte stream in the segment.

Acknowledgement number (32 bits): This is the sequence number that the receiver expects to receive next.

Offset (4 bits): This is the number of 32-bit words in the TCP header. It is needed because the Options field length is variable.

Reserved (6 bits): Reserved for future use. It must be zero.

Code Bits (6 bits): These are six flags that control the behaviour of a TCP packet. They are: 1. Urgent 2. Acknowledgement 3. Push 4. Reset connection 5. Synchronous 6. Finish.

Window (16 bits): This is used in acknowledgement segments to implement flow control. It specifies the number of data bytes which the receiver is willing to accept.

Checksum (16 bits): This is used to verify the integrity of the TCP header. The checksum is performed on a pseudo header consisting of information obtained from the IP as well as the TCP header

Urgent Pointer (16 bits): When urgent data is being sent (as specified in the code bits), this points to the end of the urgent data in the segment.

Options: This is used to specify maximum segment size during the establishment of a connection.

ERICSSON

User Datagram Protocol

- **Connectionless**
  - **No session is established**
- **Does not guarantee delivery**
  - **No sequence numbers**
  - **No acknowledgements**
- **Reliability is the responsibility of the application**
- **Uses port numbers as end points to communicate**
- **UDP Packet Format:**

Source Port	Destination Port
Length	UDP Checksum

UDP provides a connectionless packet service that offers unreliable “best effort” delivery. This means that the arrival of packets is not guaranteed, nor is the correct sequencing of delivered packets.

UDP is used by applications that do not require an acknowledgement of receipt of data and that typically transmit small amounts of data at one time, for example, the Simple Network Management Protocol (SNMP).

To use UDP, the application must supply the IP address and port number of the destination application. UDP ports are separate and distinct from TCP ports even though some of them use the same port numbers.

The UDP header is divided into four 16-bit fields:

Source port: This is the UDP protocol port number of the process on the sending device.

Destination port: This is the UDP protocol port number of the process on the destination device.

Length: This is the size in bytes of the UDP packet, including the header and data. The minimum length is 8 bytes, the length of the header alone.

UDP Checksum: This is used to verify the integrity of the UDP header. The checksum is performed on a pseudo header consisting of information obtained from the IP as well as the UDP header.

```
DLC:  ----- DLC Header -----
DLC:
DLC:  Frame 5 arrived at 03:36:02.55769 ; frame size is 60 (003C hex) bytes
DLC:  Destination      = Station cisco 07AC0A
DLC:  Source           = Station 00A0C943074A
DLC:  Ethertype        = 0800 (IP)
DLC:
IP:   ----- IP Header -----
IP:
IP:   Version = 4, header length = 20 bytes
IP:   Type of service = 00
IP:         000. .... = routine
IP:         ...0 .... = normal delay
IP:         .... 0... = normal throughput
IP:         .... .0.. = normal reliability
IP:   Total length     = 40 bytes
IP:   Identification   = 39327
IP:   Flags            = 4X
IP:         .1.. .... = don't fragment
IP:         ..0. .... = last fragment
IP:   Fragment offset  = 0 bytes
IP:   Time to live     = 128 seconds/hops
IP:   Protocol         = 6 (TCP)
IP:   Header checksum= 8FE7 (correct)
IP:   Source address   = [163.33.232.166]
IP:   Destination address = [172.28.153.100]
IP:   No options
IP:
TCP:  ---- TCP Header -----
TCP:
TCP:  Source port      = 1279
TCP:  Destination port  = 139 (NetBIOS-ssn)
TCP:  Sequence number= 265535486
TCP:  Acknowledgement number = 2067413552
TCP:  Data offset      = 20 bytes
TCP:  Flags            = 10
TCP:        ..0. .... = (No urgent pointer)
TCP:        ...1 .... = Acknowledgement
TCP:        .... 0... = (No push)
TCP:        .... .0.. = (No reset)
TCP:        .... ..0. = (No SYN)
TCP:        .... ...0 = (No FIN)
TCP:  Window           = 8760
TCP:  Checksum         = 318D (correct)
TCP:  No TCP options
TCP:


DETAIL:
0000   00 00 0C 07 AC 0A 00 A0      C9 43 07 4A 08 00 45 00
0010   00 28 99 9F 40 00 80 06      8F E7 A3 21 E8 A6 AC 1C
0020   99 64 04 FF 00 8B 0F D3      BF FE 7B 3A 3A 30 50 10
0030   22 38 31 8D 00 00 00 00      00 00 00 00
```

Summary

•In chapter 1 we examined the four layers in the TCP/IP protocol and described the functions of each layer.

•We discussed encapsulation and looked at an example of an IP packet encapsulated in an Ethernet packet.

•We looked at Internet Protocol (IP) and discussed its characteristics and how it transports data across a network. We also examined the IP packet structure.

•We discussed the ARP mechanism and looked at an example of how it works. We also examined the ARP packet structure.We discussed RARP and why and how it is used.We described Internal Control Message Protocol (ICMP) and discussed its functionality.

•We looked at Transmission Control Protocol (TCP) and discussed its characteristics. We examined how it transmits data across a network and its packet structure.

•We looked at User Datagram Protocol (UDP) and its characteristics.

Chapter 2

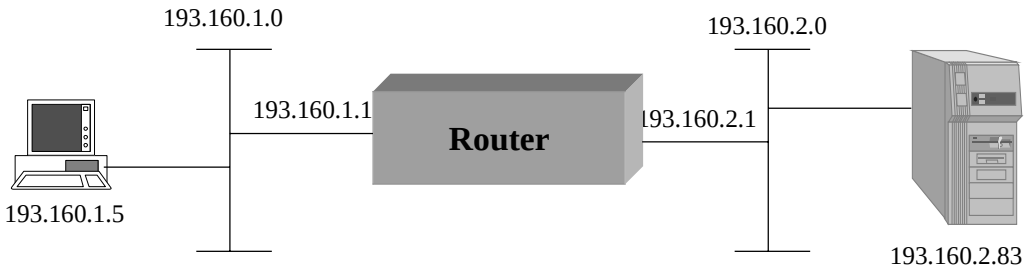
IP Addressing

Chapter 2. IP Addressing

- **After completing this chapter the student will be able to comprehend IP addressing and DHCP. Topics include:**
  - **The format of an IP Address**
  - **The different IP address classes**
  - **Default subnet masks**
  - **Dynamic Host Configuration Protocol (DHCP)**

ERICSSON

The IP Address



Binary Format	11000001 10100000 00000001 00000101
Dotted Decimal Notation	193.160.1.5

Every network interface on a TCP/IP device is identified by a globally unique IP address. Host devices, for example, PCs, typically have a single IP address. Routers typically have two or more IP addresses, depending on the number of interfaces they have.

Each IP address is 32 bits long and is composed of four 8-bit fields called octets. This address is normally represented in “dotted decimal notation” by grouping the four octets and representing each octet in decimal form. Each octet represents a decimal number in the range 0-255.

For example, 11000001 10100000 00000001 00000101, is known as 193.160.1.5.

Each IP address defines the network ID and host ID of the device.

The network ID part of the IP address is centrally administered by the Internet Network Information Centre (InterNIC) and is unique throughout the Internet. The host ID is assigned by the authority which controls the network.

The network ID identifies the systems that are located on the same network or subnet. The network ID must be unique to the internetwork.

The host ID identifies a TCP/IP network device (or host) within a network. The address for each host must be unique to the network ID.

In the example above, the PC is connected to network “193.160.1.0” and has a unique host ID of “.5”.

ERICSSON

Converting from Binary to Decimal

Binary Value	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**
	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Decimal Value	**128**	**64**	**32**	**16**	**8**	**4**	**2**	**1**

If all bits are set to 1 then the decimal value is 255 i.e. 1+2+4+8+16+32+64+128=255

Each bit position in an octet has an assigned decimal value. A bit set to zero always has a zero value. The lowest order bit has a decimal value of 1. The highest order bit has a decimal value of 128.

The highest decimal value of an octet is 255, that is, when all bits are set to one.

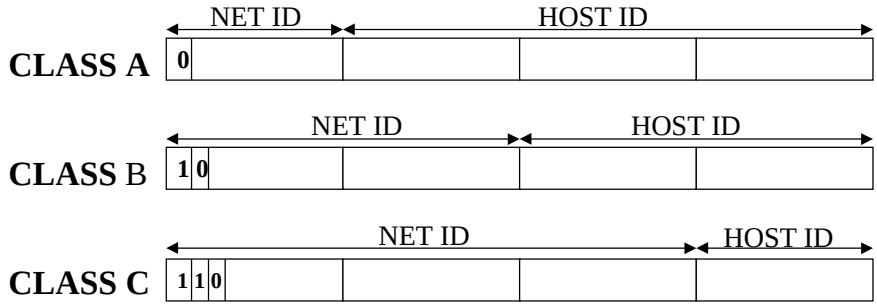
In the example below, the binary value 10011000 is converted to a decimal value of 152.

Binary Value	**1**	**0**	**0**	**1**	**1**	**0**	**0**	**0**
	2^7	**0**	**0**	2^4	2^3	**0**	**0**	**0**
Decimal Value	**128**			**16**	**8**			

The binary value 10011000 is 152, this is 128+16+8=152 .

ERICSSON

Different IP Address Classes



	Number of Networks	Hosts per Network	1st Octet
Class A	126	16,777,214	1 – 126
Class B	16,384	65,534	128 – 191
Class C	2,097,152	254	192 - 223

An IP address is 32 bits in length, divided into two or three parts. The first part makes up the network address, the second part makes up the subnet address (if used) and the third part makes up the host address.

IP address = <network number><host number>

There are five different address classes supported by IP addressing. The class of an IP address can be determined from the high-order (left-most) bits.

Class A

Class A addresses are assigned to networks with a very large number of hosts. The high-order bit in a class A address is always set to zero. The next seven bits (completing the first octet) represent the network ID and provide 126 possible networks. The remaining 24 bits (the last three octets) represent the host ID; each network can have up to 16,777,214 hosts.

Class B

Class B addresses are assigned to medium-sized to large-sized networks. The two high-order bits in a class B address are always set to binary 1 0. The next 14 bits (completing the first two octets) represent the network ID. The remaining 16 bits (last two octets) represent the host ID. Therefore, there can be 16,382 networks and up to 65,534 hosts per network.

Class C

Class C addresses are used for small networks. The three high-order bits in a class C address are always set to binary 1 1 0. The next 21 bits (completing the first three octets) represent the network ID. The remaining 8 bits (last octet) represent the host ID. Therefore, there can be 2,097,150 networks and 254 hosts per network.

ERICSSON

Different IP Address Classes

- **Class D**
  - **Used for multicast group usage - first 4 high-order bits are 1110**
- **Class E**
  - **Reserved for future use - first 5 high-order bits are 11110**

Class D

Class D addresses are used for multicast group usage. A multicast group may contain one or more hosts, or none at all. The four high-order bits in a class D address are always set to binary 1 1 1 0. The remaining bits designate the specific group in which the client participates. There are no network or host bits in the multicast operations. Packets are passed to a selected subset of hosts on a network. Only those hosts registered for the multicast operation accept the packet.

Class E

Class E is an experimental address not available for general use; it is reserved for future use. The high-order bits in a class E address are set to 1 1 1 1 0.

Note: All Internet addresses are assigned by a central authority, the Network Information Centre (NIC). They only assign the network ID portion of the address. Definition of host IDs on a network is the responsibility of the network managers. It is possible for stand-alone networks to have arbitrarily assigned network IDs, but if they ever need to connect to the Internet or other company networks, then the network ID must be assigned by the NIC to prevent an address clash.

ERICSSON

Addressing Guidelines

- **Network ID cannot be 127**
  - **127 is reserved for loop-back function**
- **Network ID and host ID cannot be 255 (all bits set to 1)**
  - **255 is a broadcast address**
- **Network ID and host ID cannot be 0 (all bits set to 0)**
  - **O means "this network only"**
- **Host ID must be unique to the network**

The following rules must be adhered to when assigning network IDs and host IDs:

• The network ID cannot be 127. The class A network address 127.0.0.0 is reserved for loop-back and is designed for testing and inter-process communication on the local device. When any device uses the loop-back address to send data, the protocol software in the device returns the data without sending traffic across any network.

• The network ID and host ID bits cannot be all 1s. If all bits are set to 1, the address is interpreted as a broadcast rather than a host ID.

  • If a destination address contains all 1s in the network ID and the host ID (i.e. 255.255.255.255) this is a limited broadcast, that is, a broadcast on the source's local network.

  • If a destination address contains all 1s in the host ID but a proper network ID, for example, 160.30.255.255, this is a directed broadcast, that is, a broadcast on a specified network (in this example network 160.30.0.0)

• The network ID and host ID bits cannot all be 0s. If all bits are set to 0, the address is interpreted to mean "this network only".

• The host ID must be unique to the local network.

ERICSSON

Private IP address space

10.0.0.0	**- 10.255.255.255 (10/8 prefix)**	**1 Class A network**
172.16.0.0	**- 172.31.255.255 (172.16/12 prefix)**	**16 Class B network**
192.168.0.0	**- 192.168.255.255 (192.168/16 prefix)**	**256 Class C network**

The Network Information Centre (NIC) has reserved the addresses listed above for use in private networks. These addresses can be used by companies that do not require to connect to the Internet or to the IP networks of other companies.

ERICSSON

Subnet Mask

- **Blocks out a portion of the IP address to distinguish the Network ID from the host ID**
- **Specifies whether the destination's host IP address is located on a local network or on a remote network**
- **Default subnets are used on IP networks that are not divided into subnets:**
  - **Class A Default Mask 255.0.0.0**
  - **Class B Default Mask 255.255.0.0**
  - **Class C Default Mask 255.255.255.0**
- **Subnetting allows an organisation to split up its network into smaller, more efficient subnetworks.**

Due to the immense growth of the Internet, assigned IP addresses became too inflexible to allow easy changes to local network configuration. Changes to local network configuration may be necessary when:

•A new physical network is installed at a location.

•Growth of the number of hosts requires splitting the local network into two or more separate networks.

To avoid having to request additional IP network addresses in these cases, the concept of subnets was introduced.

The host number part of the IP address is further subdivided into a network number and a host number. This second network is known as a subnetwork or subnet. The IP address is constructed as follows:

IP address = <network number><subnet number><host number>

The subnet number and host number are collectively known as the "local address". A host within a network which has subnets is aware of subnetting but a host in a different network is not; it still regards the local part of the IP address as a host number.

A subnet mask is a 32-bit address used to:

• Block out a portion of the IP address to distinguish the network ID from the host ID.

• Specify whether the destination's host IP address is located on a local network or on a remote network.

ERICSSON

Determining the destination of a packet

- **The source's IP address is ANDed with its subnet mask. The destination's IP address is ANDed with the same subnet mask. If the result of both ANDing operations match, the destination is local to the source, that is, it is on the same subnet.**
- **1 AND 1 = 1. Other combinations = 0.**
- **For example 160.30.20.10 is on the same subnet as 160.30.200.100 if the mask is 255.255.0.0**

IP Address	160.30.20.10	10100000 00011110 00010100 00001010
Subnet Mask	255.255.0.0	11111111 11111111 00000000 00000000
Result	160.30.0.0	10100000 00011110 00000000 00000000

IP Address	160.30.200.100	10100000 00011110 11001000 01100100
Subnet Mask	255.255.0.0	11111111 11111111 00000000 00000000
Result	160.30.0.0	10100000 00011110 00000000 00000000

ANDing is an internal process that TCP/IP uses to determine whether a packet is destined for a host on a local network, or a host on a remote network.

When TCP/IP is initialised, the host's IP address is ANDed with its subnet mask. Before a packet is sent, the destination IP address is ANDed with the same subnet mask. If both results match, IP knows that the packet belongs to a host on the local network. If the results don't match, the packet is sent to the IP address of an IP router.

To AND the IP address to a subnet mask, TCP/IP compares each bit in the IP address to the corresponding bit in the subnet mask. If both bits are 1's, the resulting bit is 1. If there is any other combination, the resulting bit is 0.

The four possible variations are as follows:

1 AND 1 = 1

1 AND 0 = 0

0 AND 0 = 0

0 AND 1 = 0

ERICSSON

Example: Network with default subnet mask

Allocated Class B IP network address 160.30.0.0

Default, -2-octet mask 255.255.0.0 i.e. no subnetting				
1 Network, 16 bits available for host				
	255	255	0	0
	1111 1111	1111 1111	0000 0000	0000 0000
No. of Hosts	**Network**		**Host**	
160.30.0.1	1010 0000	0001 1110	**0000 0000**	**0000 0001**
↓			↓	↓
160.30.255.254	1010 0000	0001 1110	**1111 1111**	**1111 1110**
Maximum of 65,534 hosts (2^{16} - 2)				

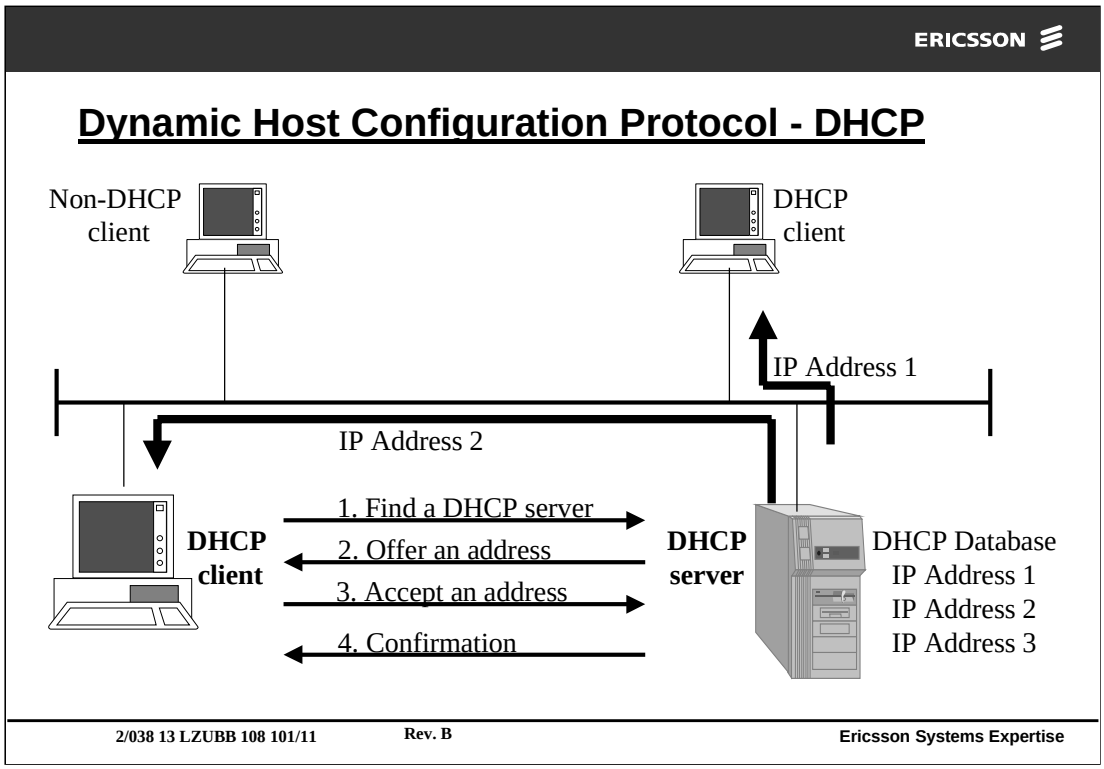
The example above calculates the number of hosts on a network when a default subnet mask is used.

A single Class B IP network address has been allocated to an organisation. The default subnet mask for a Class B network is 255.255.0.0.

Using the default mask the organisation has only one network (160.33.0.0). The host addresses 0.0 and 255.255 cannot be used. Therefore, the lowest possible host address within this network is 0.1 and the highest possible network address within this network is 255.254.

A shortcut method of working out the number of hosts in a subnet is {(2 to the power of the number of zeros in the mask) less two}.

In the example above this is 2^{16} -2 which gives a total of 65,534 hosts.



DHCP centralises and manages the allocation of TCP/IP configuration information by automatically assigning IP addresses to devices configured to use DHCP. Implementing DHCP eliminates some of the configuration problems associated with manually configuring TCP/IP. Typing in the IP address, subnet mask, or default gateway incorrectly can lead to problems including communication difficulties and network problems due to a duplicate IP address.

Each time a DHCP client starts, it requests IP an address from a DHCP server.

When a DHCP server receives a request, it selects IP addressing information from a pool of addresses defined in its database and offers it to the DHCP client. If the client accepts the offer, the IP addressing information is leased to the client for a specified period of time.

In addition, the DHCP server will supply a subnet mask and optional values such as default gateway address, Domain Name Server (DNS) address and WINS (Windows Internet Name Service) address.

Non-DHCP clients still need to be configured manually with static addresses.

If there is no available IP addressing information in the pool to lease to a client, the client cannot initialise TCP/IP.

ERICSSON

DHCP

- **DHCP supports three mechanisms for IP address allocation:**
  - **Manual allocation**
  - **Automatic allocation**
  - **Dynamic allocation**

DHCP supports three mechanisms for IP address allocation.

1. Manual Allocation

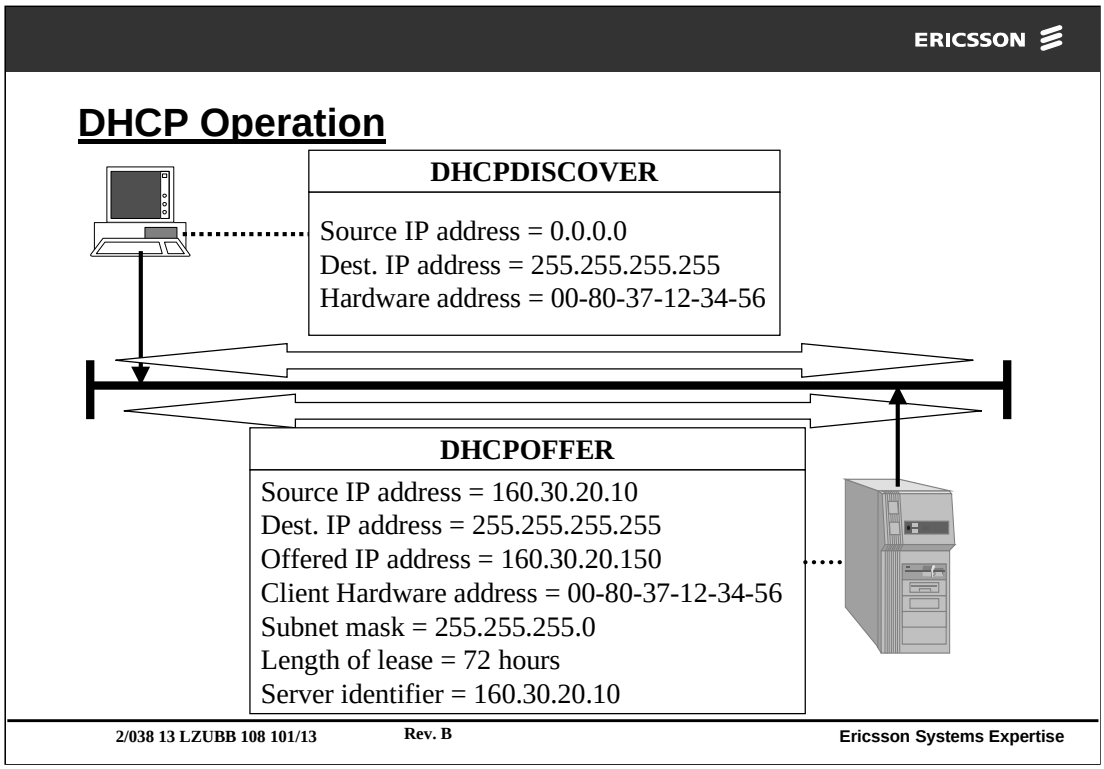
In this scheme, DHCP is simply used as a mechanism to deliver a predetermined network address and other configuration options to a host. There is a one-to-one mapping between the unique client identifier (generally the Ethernet address) offered by the client during DHCP initialisation and the IP address returned to the client by the DHCP server. It is necessary for a network administrator to provide the unique client ID/IP address mapping used by the DHCP server.

2. Automatic Allocation

This is similar to manual allocation in that a permanent mapping exists between a host's unique client identifier and its IP address. However, in automatic allocation this mapping is created during the initial allocation of an IP address. The IP addresses assigned during automatic allocation come from the same pool as dynamic addresses, but once assigned they cannot be returned to the free address pool without administrative intervention. Both automatic and manually assigned addresses are considered to have permanent leases.

3. Dynamic Allocation

DHCP assigns an IP address for a limited period of time. This IP address is known as a lease. This mechanism allows addresses that are no longer needed by their host to be automatically re-used.



Here we shall explain Dynamic Allocation.

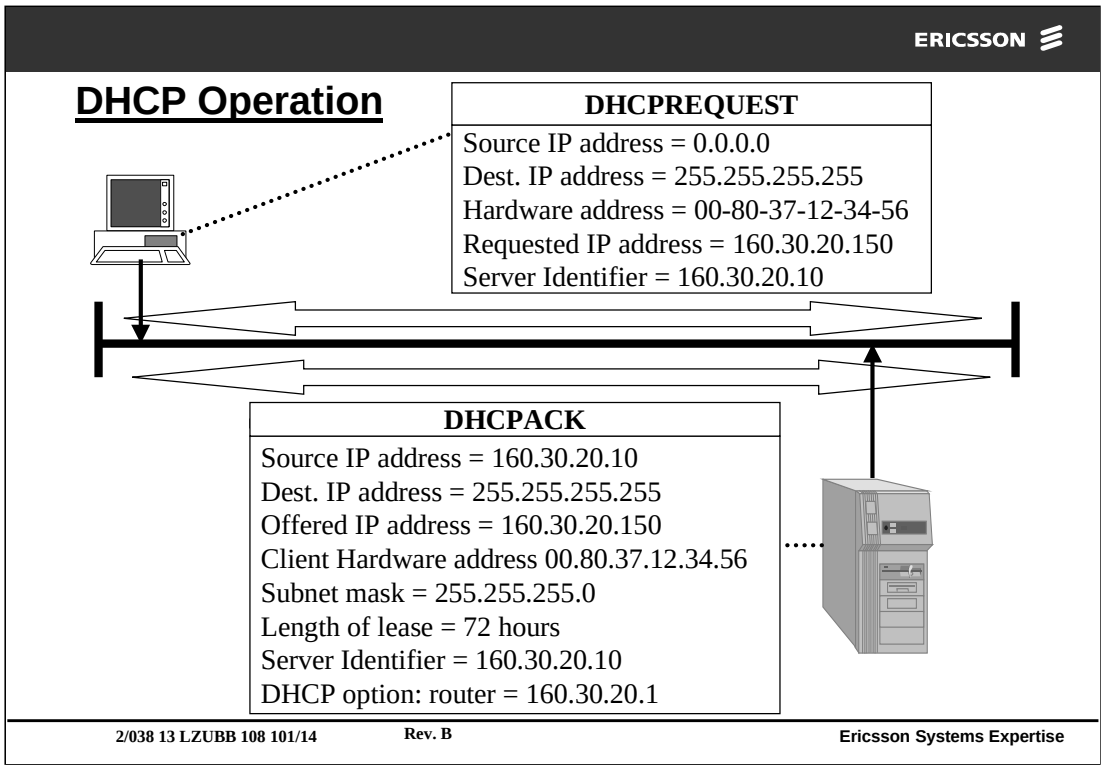
DHCP uses a four-phase process to configure a DHCP client. In the first two phases the client requests a lease from a DHCP server and a DHCP server offers an IP address to the client.

IP Lease Request

The first time a client is initialised, it requests an IP address lease by broadcasting a request to all DHCP servers.

Because the client does not have an IP address or know the IP address of a DCHP server, it uses 0.0.0.0 as the source address and 255.255.255.255 as the destination address.

The request for a lease is sent in a **DHCPDISCOVER** message. This message also contains the client's hardware address and computer name, so that DHCP servers know which client sent the request.



IP Lease Offer

All DHCP servers that receive the request, and have a valid configuration for the client, broadcast an offer with the following information: the client's hardware address, an offered IP address, a subnet mask, the length of the lease and a server identifier (the IP address of the offering DHCP server).

A broadcast is used because the client does not yet have an IP address. The offer is sent as a **DHCPOFFER** message.

The DHCP server reserves the IP address so that it will not be offered to another DHCP client. The DHCP client selects the IP address from the first offer it receives.

In the last two phases, the client selects an offer and the DHCP server acknowledges the lease.

IP Lease Selection

After the client receives an offer from at least one DHCP server, it broadcasts to all DHCP servers that it has made a selection by accepting an offer.

The broadcast is sent in a **DHCPREQUEST** message and includes the identifier (IP address) of the server whose offer was accepted. All other DHCP servers then retract their offer so that their IP addresses are available for the next IP lease request.

IP Lease Acknowledgement (Successful)

The DHCP server with the accepted offer broadcasts a successful acknowledgement to the client in the form of a DHCPACK message. This message contains a valid lease for an IP address and possibly other configuration information.

When the DHCP client receives the acknowledgement, TCP/IP is completely initialised and is considered a bound DHCP client. Once bound, the client can use TCP/IP to communicate on the internetwork.The client stores the IP address, subnet mask and other IP addressing information locally.

IP Lease Acknowledgement (Unsuccessful)

An unsuccessful acknowledgement (DHCPNACK) is broadcast if:

•The client is trying to lease its previous IP address and the IP address is no longer available, or

• The IP address is invalid because the client has been physically moved to a different subnet.

When the client receives an unsuccessful acknowledgement, it returns to the process of requesting an IP lease.

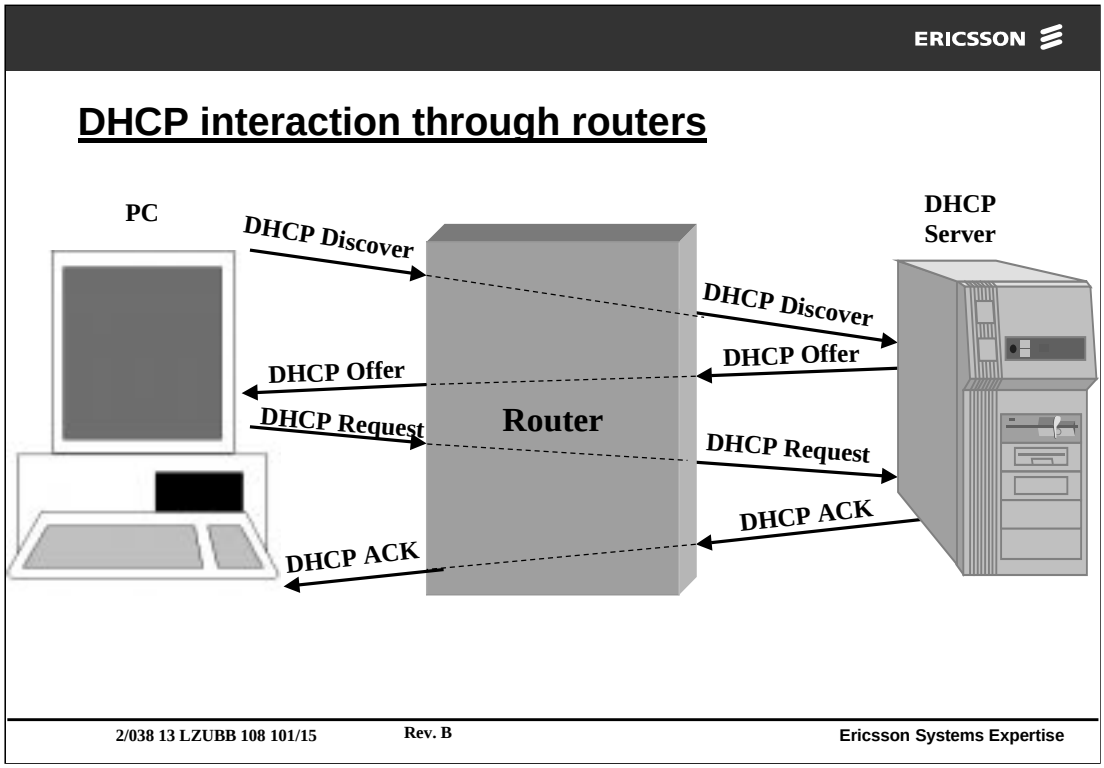
IP Lease Renewal

All DHCP clients attempt to renew their lease when 50 percent of the lease time has expired. To renew its lease, a DHCP client sends a DHCPREQUEST message directly to the DHCP server from which it obtained the lease.

If a lease cannot be renewed by the original DHCP server, the client still uses the address as 50 percent of the lease life is still available.

The client will attempt to contact any available DHCP server when 87.5 percent of the lease time has expired.

If this is unsuccessful and the lease expires, the DHCP client can no longer use the IP address and communication over TCP/IP stops until a new IP address can be assigned to the client.



Routers can be configured to act as “relay agents” to allow DHCP servers located on one IP network to serve configuration requests from remote networks.

A relay agent that conforms to RFC 1542 relays DHCP packets to a remote network even though they are broadcast packets. Before relaying a DHCP message from a DHCP client, the agent examines the gateway IP address field. If the field has an IP address of 0.0.0.0 the agent fills it with the routers IP address.

When the DHCP server receives the message it examines the relay IP address field to see if it has a DHCP scope (a pool of IP addresses) that can be used to supply an IP address lease. If the DHCP server has multiple scopes the address in the relay IP address field identifies the DHCP scope from which to offer an IP address lease. This process allows one DHCP server to manage different scopes for different networks.

When it receives the DHCP Discover message, the DHCP server sends a DHCP Offer directly to the relay agent identified in the gateway IP address field, and the agent relays the message to the client. The client’s IP address is unknown, thus it has to be broadcast on the local subnet.

Similarly a DHCP request message is relayed from client to server and a DHCP ACK message is relayed from server to client according to RFC 1542.

Summary

•In chapter 2 we looked at the IP address and discussed its characteristics. We discussed the differences between the different IP address classes, and we examined the rules for addressing.

•We defined subnet masking and highlighted the reason why an organisation might want to use it. We looked in detail at examples of default and customised masks.

•We discussed DHCP and looked at an example of its operation.

Chapter 3

Subnetting

ERICSSON

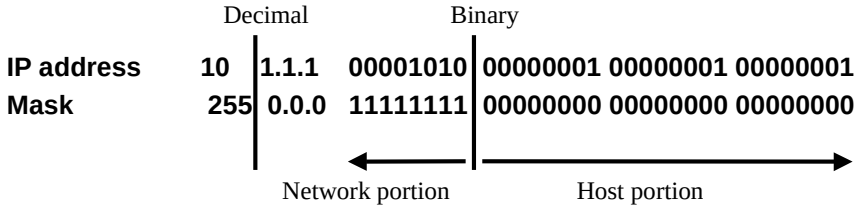
Chapter 3. Subnetting

- **After completing this chapter the student will be able to implement IP subnetting schemes. Topics include:**
  - **How to implement subnetting**
  - **Defining Subnet Mask and Subnet IDs**
  - **Variable Length Subnet Masks**
  - **Supernetting**
  - **Network design problem-solving exercise**

ERICSSON

How to implement subnetting

- **Subnetting is the technique used to allow a single network address to span multiple physical networks**
  - **Class A natural mask 255.0.0.0**
  - **Class B natural mask 255.255.0.0**
  - **Class C natural mask 255.255.255.0**



3/038 13 LZUBB 108 101/2 Rev. B Ericsson Systems Expertise

Subnetting is the technique used to allow a single network address to span multiple physical networks. A subnetwork or subnet is a subset of a class A, B or C network.

Let's take a closer look at IP addresses to help us understand subnetting. IP addresses are made up of a network portion and a host portion. A network mask is used to separate the network information from the host information.

In the diagram shown above, the network mask 255.0.0.0 is applied to the IP address 10.1.1.1. The mask in binary notation is a series of contiguous ones followed by a series of contiguous zeros. The ones portion represents the network ID, whereas the zeros portion represents the host ID. This splits the IP address 10.1.1.1 into a network portion of 10 and a host portion of 1.1.1. As such, classes A, B and C each have a natural mask, which is the mask created by the very definition of the network and host portions of each class.

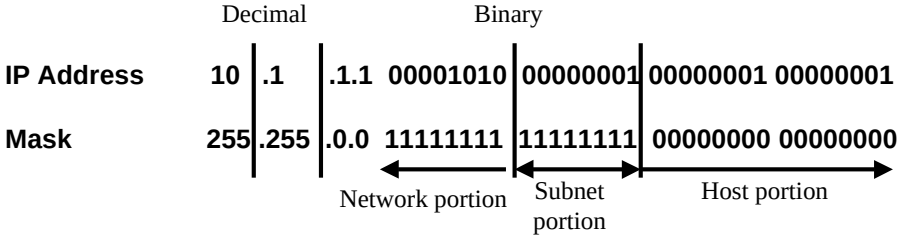
- Class A natural mask 255.0.0.0
- Class B natural mask 255.255.0.0
- Class C natural mask 255.255.255.0

By separating the network and host IDs of an IP address, masks facilitate the creation of subnets.

With the use of masks, networks can be divided into subnetworks by extending the network IDs of the address into the host ID. Subnetting increases the number of subnetworks and reduces the number of hosts.

ERICSSON

Subnetting



- **Before subnetting: 1 network with approx. 16 million hosts**
- **After subnetting: 254 networks with approx. 65 thousand hosts per subnet**

In the diagram shown above a mask of 255.255.0.0 is applied to the IP address 10.1.1.1. This divides the IP address 10.1.1.1 into a network portion of 10, a subnet portion of 1 and a host portion of 1.1. The 255.255.0.0 mask has borrowed a portion of the host space and applied it to the network space. As a result, the network space of class 10 has increased from a single network 10.0.0.0, to 254 (2^8-2) subnetworks ranging from 10.1.0.0 to 10.254.0.0. This decreases the number of hosts per subnet from 16,777,214 (2^{24}-2) to 65534 (2^{16}-2).

ERICSSON

Example: network with customised mask

Allocated Class B IP network address 160.30.0.0

3 octet mask 255.255.255.0				
8 bits available for subnets and 8 bits available for host				
	255	255	255	0
	1111 1111	1111 1111	1111 1111	0000 0000
No. of Subnets	**Network**			**Host**
160.30.1.x	1010 0000	0001 1110	**0000 0001**	xxxx xxxx
↓			↓	
160.30.254.x	1010 0000	0001 1110	**1111 1110**	xxxx xxxx
Maximum of 254 subnets (2^8 - 2)				

The example above calculates the number of subnets available when a customised mask is applied.

A single Class B IP network address has been allocated to an organisation. The default subnet mask for a Class B network is 255.255.0.0.

In the previous chapter we saw that using the default subnet mask on a Class B network gives one single network with a total of 65,534 hosts.

Using the customised mask 255.255.255.0 the organisation has up to 254 subnets, rather than just one single network.

The network addresses 160.30.0 and 160.30.255 cannot be used. Therefore the lowest possible network address within this internetwork is 160.30.1 and the highest possible network address within this network is 160.30.254.

A shortcut method of working out the number of subnets is {(2 to the power of the number of ones in the mask, excluding the default mask portion) less two}.

In the example above this is 2^8 -2 which gives a total of 254 subnets.

ERICSSON

Example: network with customised mask (continued)

Allocated Class B IP network address 160.30.0.0

3 octet mask 255.255.255.0				
8 bits available for subnets and 8 bits available for host				
	255	255	255	0
	1111 1111	1111 1111	1111 1111	0000 0000
No. of hosts	**Network**			**Host**
160.30.x.1	1010 0000	0001 1110	xxxx xxxx	**0000 0001**
↓				↓
160.30.x.254	1010 0000	0001 1110	xxxx xxxx	**1111 1110**
Maximum of 254 hosts (2^8 - 2)				

This is the same example as the previous page, but this time we want to calculate the number of hosts in any one of the 254 subnets.

The host addresses 0 and 255 cannot be used. Therefore the lowest possible host address on each subnet is 1 and the highest possible host address on each subnet is 254.

As in the previous example, a shortcut method of working out the number of hosts in a subnet is {(2 to the power of the number of zeros in the mask) less two}.

In the example above this is 2^8 -2 which gives a total of 254 hosts.

ERICSSON

Defining a subnet mask based on the number of subnets required

- Add two to the number of subnets required and convert to binary
- Count the number of bits required
- Convert the required number of bits to decimal in high order
- Example: Class C address, 5 subnets required
  - 7 converted to binary is 110 (3 bits)
  - Three bits are required so configure the first three bits of the host ID as the subnet ID
  - The decimal value for 1110 0000 is 224
  - The subnet mask is 255.255.255.224 for this class C address

If you are dividing your network into subnets, you need to define a subnet mask. Follow these steps:

1. Determine the number of subnets you require. Add two to the number of subnets required and convert to binary.

2. Count the number of bits required to represent the number of physical segments in binary. For example, if you need five subnets, the binary value of seven is 110. Representing seven in binary requires three bits.

3. Convert the required number of bits to decimal format in high order (from left to right). For example, if three bits are required, configure the first three bits of the host ID as the subnet ID. The decimal value for binary 11100000 is 224. The subnet mask is 255.255.225.224 (for a Class C address).

ERICSSON

Defining a subnet mask based on the number of hosts

- Add two to the number of hosts required and convert the sum to binary
- Count the number of bits required for the host portion
- Subtract this number from the total number of bits in the host ID
- Convert the required number of bits to decimal in high order
- Example:
  - Class B address, 2000 devices per subnet required
  - 2002 converted to binary is 11111010010 (11 bits)
  - Eleven bits are required for the host so configure the first five bits of the host ID as the subnet ID (16 - 11 = 5)
  - The decimal value for 1111 1000 is 248
  - The subnet mask is 255.255.248.0 for a class B address

If you do not want all your hosts to be on the same subnet, you need to define a subnet mask, assuming that you have been allocated a single network address.

Follow these steps:

1. Decide on the number of hosts you want to have on each subnet. Convert this number to binary format.

2. Count the number of bits required to represent the number of hosts in binary. For example, if you want up to 2,000 hosts per subnet, the binary value for 2002 is 11111010010. Representing 2,002 in binary requires 11 bits. To calculate the number of bits required for the mask, subtract the number of bits required for the host from the total number of bits in the host. In this example the result is five (16 - 11).

3. Convert the required number of bits to decimal format in high order (from left to right). In this example, five bits are required. Configure the first five bits of the host ID as the subnet ID. The decimal value for 11111000 is 248. The subnet mask is 255.255.248.0 (for a class B address).

ERICSSON

Subnet Conversion Tables - Class C Networks

Subnet bits	Subnet mask	4th octet of mask	Maximum number of subnets	Maximum number of hosts
0	255.255.255.0	0000 0000	0	254
1	255.255.255.128	1000 0000	N/A	N/A
2	255.255.255.192	1100 0000	2	62
3	255.255.255.224	1110 0000	6	30
4	255.255.255.240	1111 0000	14	14
5	255.255.255.248	1111 1000	30	6
6	255.255.255.252	1111 1100	62	2
7	255.255.255.254	1111 1110	N/A	N/A

The subnet conversion table above shows all the possible combinations of subnets and hosts for a Class C network address.

For example, if we want to implement five subnets, we would use a subnet mask of 255.255.255.224. This would allow up to a maximum of six subnets with 30 devices per subnet.

If there are zero bits in the subnet mask we are not using subnetting and are left with the default of one network with 254 hosts. We cannot just use one bit in the subnet mask because the only subnet IDs would be 0 and 1 neither of which are valid. Similarly we cannot use 7 bits in the subnet ID because the only host IDs would again be 0.

ERICSSON

Defining Subnet IDs

Example mask = 255.255.255.240

Mask 240 =	**1111**	**0000**	
	~~**0000**~~	~~**0000**~~	~~**= 0**~~
	0001	**0000**	**= 16**
	0010	**0000**	**= 32**
	0011	**0000**	**= 48**
	0100	**0000**	**= 64**
	1101	**0000**	**= 208**
	1110	**0000**	**= 224**
	~~**1111**~~	~~**0000**~~	~~**= 240**~~

Subnet IDs are 16, 32, 48, 64,.....208, 224.

When a portion of the address, blocked out by the subnet mask changes, the network devices know that these addresses are in different subnets. For example, for all addresses between 16 and 31 in the diagram above, the 4 bits blocked by the mask are 0001. These are on the same subnet. Therefore, for address 32 which is binary 0010 0000, we can see that the four bits blocked by the mask portion have changed. Therefore this must be a different subnet.

Note: in the example above, 16 is the subnet ID but it is not a valid host ID since 16 = 0001 0000 and we cannot have all zeros in the host portion. Similarly 31 is not a valid host ID since 31 = 0001 1111 which is the broadcast address for this subnet.

Subnet IDs comprised of all 0s or all 1s are called special case subnet addresses. A subnet ID of all 1s indicates a subnet broadcast while a subnet ID of all 0s indicates “this subnet”. When subnetting it is strongly recommended not to use these subnet IDs. However, it is possible to use these special case subnet addresses if they are supported by all routers and hardware on the network. Request For Comment (RFC) 950 details the limitations imposed when using special case addresses.

ERICSSON

Shortcut method for defining Subnet ID's using the Subnet Conversion Table

- **From the maximum number of hosts**
  - **Add 2 to the maximum number of hosts and this gives the first valid subnet ID. All subsequent IDs are multiples of the first valid subnet ID.**
  - **Example: maximum number of hosts = 14 14+2=16**
  - **subnet IDs = 16, 32, 48, 64,……..**
- **From the maximum number of subnets**
  - **add 2 to the maximum number of subnets. Divide 256 by this number and the result is the first valid subnet ID. All subsequent ID's are multiples of the first valid subnet ID.**
  - **Example: maximum number of subnets = 14 14+2=16 256/16 = 16**
  - **subnet Ids = 16, 32, 48, 64,……..**

Subnet IDs

There are two shortcut methods to define the subnet ID.

1. Based on the subnet conversion table. This is described in the overhead above.

2. Based on the number of bits in the host portion. This is described in the following text.

Shortcut method for defining subnet IDs from the number of bits in the host portion.

Count the number of bits in the host ID portion. Multiply this number by a power of two and this is the first valid subnet ID. All subsequent subnet IDs are multiples of the first valid subnet ID.

Mask = 255.255.255.192

192 = 1100 0000

Six bits in host portion $2^6 = 64$

Subnet IDs 0, 64, 128, 192

Mask = 255.255.255.224

224 = 1110 0000

Five bits in host portion $2^5 = 32$

Subnet IDs 0, 32, 64, 96, 128, 160, 192, 224

Mask = 255.255.255.240

240 = 1111 0000

Four bits in host portion $2^4 = 16$

Subnet IDs = 0,16, 32, 48, 64, 80, 96, 112, 128, 144, 160, 176, 192, 208, 224, 240

Mask = 255.255.255.248

248 = 1111 1000

Three bits in host portion $2^3 = 8$

Subnet IDs = 0, 8, 16, 24, 32, 40,……., 224, 232, 240, 248

Mask = 255.255.255.252

252 = 1111 1100

Two bits in host portion $2^2 = 4$

Subnet IDs = 0, 4, 8, 12, 16, 20,……., 240, 244, 248, 252

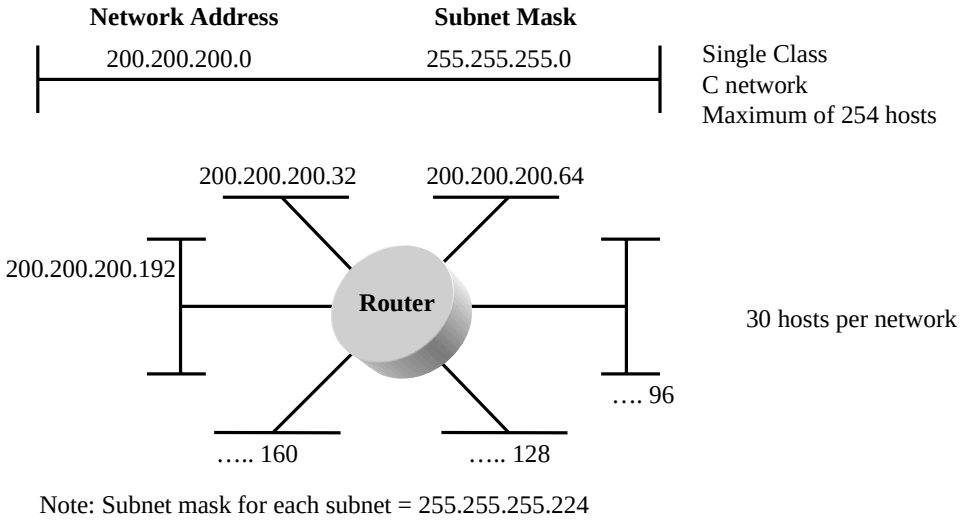
Note; in the last example there are only two valid host IDs on each subnet.

For example; in subnet ID = 4 address 5 and 6 are the only two valid source addresses.

4	0000 01 00	not valid
5	0000 01 01	
6	0000 01 10	
7	0000 01 11	not valid

ERICSSON

Subnetting Example



Note: Subnet mask for each subnet = 255.255.255.224

In the example above, a small company has been assigned a single Class C network. Without subnetting, up to a maximum of 254 hosts can share this network. In this configuration, if one device sends out an IP broadcast (e.g. DHCP Discover message) it will be received by every device on the network. To improve performance, the network administrator may reduce the number of devices that receive the broadcast by splitting the network into smaller subnets separated by a router.

In the example above, the network has been split into six smaller subnets with a maximum of 30 hosts on each subnet.

Note: the total maximum number of hosts on the network has been reduced from 254 to 180 hosts. Consult the subnet conversion table for all possible combinations of hosts and subnets.

ERICSSON

Subnet Conversion Tables - Class B Networks

Subnet Bits	Subnet Mask	Maximum # of Subnets	Maximum # of Hosts
0	255.255.0.0	0	65,534
1	255.255.128.0	N/A	N/A
2	255.255.192.0	2	16,382
3	255.255.224.0	6	8,190
4	255.255.240.0	14	4,094
5	255.255.248.0	30	2,046
6	255.255.252.0	62	1,022
7	255.255.254.0	126	510
8	**255.255.255.0**	**254**	**254**
9	255.255.255.128	510	126
10	255.255.255.192	1,022	62
11	255.255.255.224	2,046	30
12	255.255.255.240	4,094	14
13	255.255.255.248	8,190	6
14	255.255.255.252	16,382	2

The subnet conversion table above shows all the possible combinations of subnets and hosts on a class B network address.

For example, if we want to implement subnets with approximately 100 devices on each we would use a subnet mask of 255.255.255.128. This would allow up to a maximum of 510 subnets with 126 devices on each.

A commonly used subnet mask in class B networks is 255.255.255.0. This allows for 254 subnets with 254 devices each.

ERICSSON

Static Subnetting vs Variable Length Subnetting

- **Static subnetting means that all subnets in the subnetted network use the same subnet mask**
- **Simple to implement and easy to maintain, but results in wasted address space for small networks**
- **For example, a network of four hosts that uses a subnet mask of 255.255.255.0 wastes 250 IP addresses**
- **Variable Length subnetting implies that the subnetworks that make up the network may use different subnet masks**
- **A small subnet with only a few hosts needs a subnet mask that accommodates only these few hosts**

Each host on a TCP/IP network requires a subnet mask. A default subnet mask is used when a network is not divided into subnets. A customised subnet mask is used when a network is divided into subnets.

In a default subnet mask, all bits that correspond to the network ID are set to 1. The decimal value in each of these octets is 255. All bits that correspond to the host ID are set to 0.

For example, the class B address 160.30.100.10 has a network ID of 160.30.0.0 and a host ID 100.10. The default mask is therefore 255.255.0.0.

There are two types of subnetting: static and variable length.

•Static subnetting means that all subnets in the subnetted network use the same subnet mask. This is simple to implement and easy to maintain, but results in wasted address space for small networks. For example, a network of four hosts that uses a subnet mask of 255.255.255.0 wastes 250 IP addresses.

•Variable length subnetting implies that the subnetworks that make up the network may use different subnet masks. A small subnet with only a few hosts needs a subnet mask that accommodates only these few hosts.

ERICSSON

Variable Length Subnet Mask (VLSM)

- **Variable Length Subnet Mask (VLSM) refers to the fact that one network can be configured with different masks**

252 (1111 1100) - 62 subnets with 2 hosts each
248 (1111 1000) - 30 subnets with 6 hosts each
240 (1111 0000) - 14 subnets with 14 hosts each
224 (1110 0000) - 6 subnets with 30 hosts each
192 (1100 0000) - 2 subnets with 62 hosts each

- **How can the network be divided into 3 subnets, with 50 hosts in one subnet, and 25 hosts for each of the remaining subnets ?**

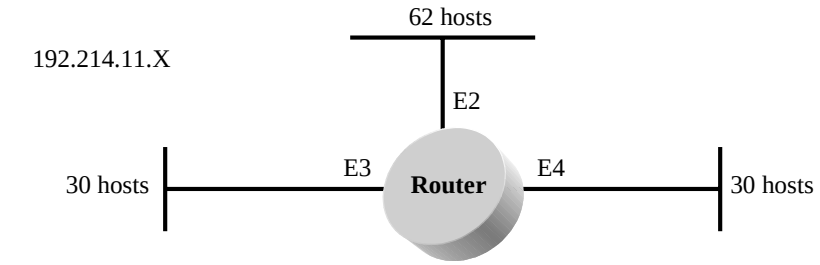
Variable Length Subnet Mask (VLSM) refers to the fact that one network can be configured with different masks. The idea behind Variable Length Subnet Masks is to offer more flexibility in dividing a network into multiple subnets while still maintaining an adequate number of hosts in each subnet. Without VLSM, one subnet mask only can be applied to a network. This restricts the number of hosts given the number of subnets required. If you pick the mask so that you have enough subnets, you might not be able to allocate enough hosts in each subnet. The same is true for the hosts; a mask that allows enough hosts might not provide enough subnet space.

Suppose for example, you were assigned a Class C network 192.214.11.0 and you need to divide that network into three subnets, with 50 hosts in one subnet and 25 hosts for each of the remaining subnets. Without subnetting you have 254 addressees available, 192.214.11.1 to 192.214.11.2. The desired subdivision cannot be done without VLSM, as we shall see.

There are a handful of subnet masks of the form 255.255.255.X that can be used to divide the class C network 192.214.11.0 into more subnets. Remember that a mask should have a contiguous number of ones starting from the left (network portion) and the rest of the bits should be zeros. The masks shown in the diagram above could be used to segment the 254 addresses available to you into more subnets.

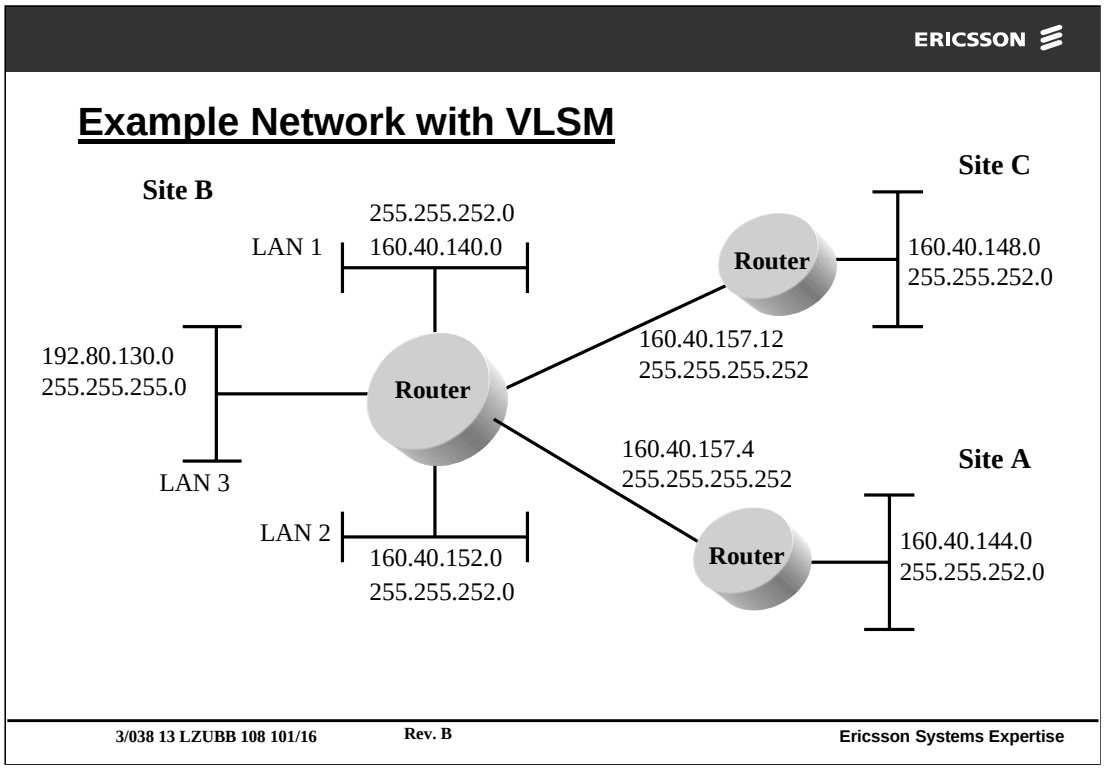
ERICSSON

VLSM



62 addresses (E2) (mask 255.255.255.192)	
30 addresses (E3) (mask 255.255.255.224)	30 addresses (E4) (mask 255.255.255.224)

Without VLSM, you have the choices to use either mask 255.255.255.192 and divide the addresses into two subnets with 62 hosts each or use 255.255.255.224 and divide the addresses into 6 subnets with 30 hosts each. This would not meet the requirement of having 50 hosts on one segment and 25 hosts on each of the two other segments. By using multiple masks, you can use mask 192 to divide the addresses into two subnets with 62 hosts each and then use the mask 224 to further subnet the second batch of addresses into two subnets with 30 hosts each. The diagram above shows how to divide the address space accordingly.



The diagram above is an example of a real network with VLSM.

The company owns the Class B network 160.40.0.0 and the Class C network 192.80.130.0. On Site A all devices are on the same subnet. There can be a maximum of 1,022 devices (2^{10} -2), since there are 10 bits available for host addresses. Valid network addresses are from 160.40.144.1 to 160.40.147.254.

Similarly, on Site C all devices are on the same subnet. There can be a maximum of 1,022 devices. Valid network addresses are from 160.40.148.1 to 160.40.151.254.

On Site B there are three subnets. Two of the subnets (LAN 1 & LAN 2) can have 1,022 devices.

Valid network addresses on LAN 1 are 160.40.140.1 to 160.40.143.254.

Valid network addresses on LAN 2 are 160.40.152.1 to 160.40.155.254.

Also on Site C there is one Class C network which can have a maximum of 254 devices. Valid network addresses are 192.80.130.1 to 192.180.130.254.

The WAN links both use the smallest possible subnets to support 2 network addresses by using a mask 255.255.255.252.

ERICSSON

Supernetting

- **Supernetting is the reverse of subnetting**
- **Supernetting uses bits from the network ID and masks the host ID**
- **Supernetting is the term used when multiple network addresses of the same class are combined into blocks.**
- **Prerequisites for Supernetting**
  - **networking addresses be consecutive and fall on the correct boundaries**
  - **you must either be running static routing everywhere or be using a classless routing protocol such as RIP2 or OSPF which includes subnet mask information and can pass supernetting information in order for this to work**

Supernetting is the reverse of subnetting, which allows the use of a single IP network on multiple interfaces.

Supernetting is the term used when multiple network addresses of the same class are combined into blocks. If the IP networks are contiguous, you may be able to use supernetting.

A prerequisite for supernetting is that the networking addresses be consecutive and that they fall on the correct boundaries. To combine two Class C networks, the first address's third octet must be evenly divisible by 2. If you need to supernet 8 networks, the mask would be 255.255.248.0 and the first address's third octet needs to be divisible by 8. For example, 198.41.15.0 and 198.41.16.0 could not be combined into a supernet, but 198.41.18.0 and 198.41.19.0 could.

Another prerequisite for supernetting is that you must either be running static routing everywhere or be using a classless routing protocol such as RIP2 or OSPF. These include subnet mask information and can pass supernetting information in order for this to work.

Supernetting is also used in routers to aggregate destination addresses instead of four entries like this:

192.192.192.0/255.255.255.0	next hop: xx.xx.xx.xx
192.192.193.0/255.255.255.0	next hop: xx.xx.xx.xx
192.192.194.0/255.255.255.0	next hop: xx.xx.xx.xx
192.192.195.0/255.255.255.0	next hop: xx.xx.xx.xx

we could have

192.192.192.0/255.255.252.0	next hop: xx.xx.xx.xx

ERICSSON

Supernetting: example

Class C address	Binary Equivalent
192.192.192.0	11000000.11000000.110000-00.00000000
192.192.193.0	11000000.11000000.110000-01.00000000
192.192.194.0	11000000.11000000.110000-10.00000000
192.192.195.0	11000000.11000000.110000-11.00000000

4 Class C addresses assigned (up to 254 hosts each)
Construct mask to get one network (with up to 1,022 hosts)

1st two octets all ones = 255.255
Last octets = 0
Third octet 6 bits for network ID
and 2 for the host ID = 1111 1100 = 252
Subnet mask = 255.255.252.0

Supernetting is most often used in Class C addresses. A single Class C IP network has 24 bits for the network portion of the IP address and 8 bits for the host portion of the IP address. This gives a possibility of 254 hosts within a Class C IP network (2*8=256 - 2 = 254).

Suppose we need 1000 IP addresses for our network. Our ISP assigns us four Class Cs:

192.192.192.0

192.192.193.0

192.192.194.0

192.192.195.0

Now let’s convert them to binary:

192.192.192.0 11000000.11000000.110000-00.00000000

192.192.193.0 11000000.11000000.110000-01.00000000

192.192.194.0 11000000.11000000.110000-10.00000000

192.192.195.0 11000000.11000000.110000-11.00000000

As you can see, the only part that changes is the last two bits in the third octet. Now let's treat everything to the left of the '-' as the network ID and everything to the right as a host ID. Using this scheme we have 10 bits available for host IDs which gives us 1022 possible hosts (2^{10}-2). This is we require. All we need to do now is to construct the appropriate subnet mask.

The first two octets will be all 1s (255) and last octet will be all 0s (0). The third octet has 6 bits for the network ID and 2 bits for the host ID. All we have to do is convert 1111 1100 to decimal. Therefore the mask of the third octet is 252.

Summary

•In chapter 3 we defined subnetting and highlighted the reason why an organisation might want to use it. We looked in detail at examples of default and customised masks.

•We discussed VLSM and supernetting and looked at an example of their operation.

Chapter 4

IP Routing

Chapter 4. IP Routing

- **Chapter Objectives - At the end of this chapter students will comprehend:**
  - **How IP routing works**
  - **The difference between using a default gateway and proxy ARP**
  - **The difference between static and dynamic routes**
  - **Routing protocols (RIP, OSPF and BGP)**

ERICSSON

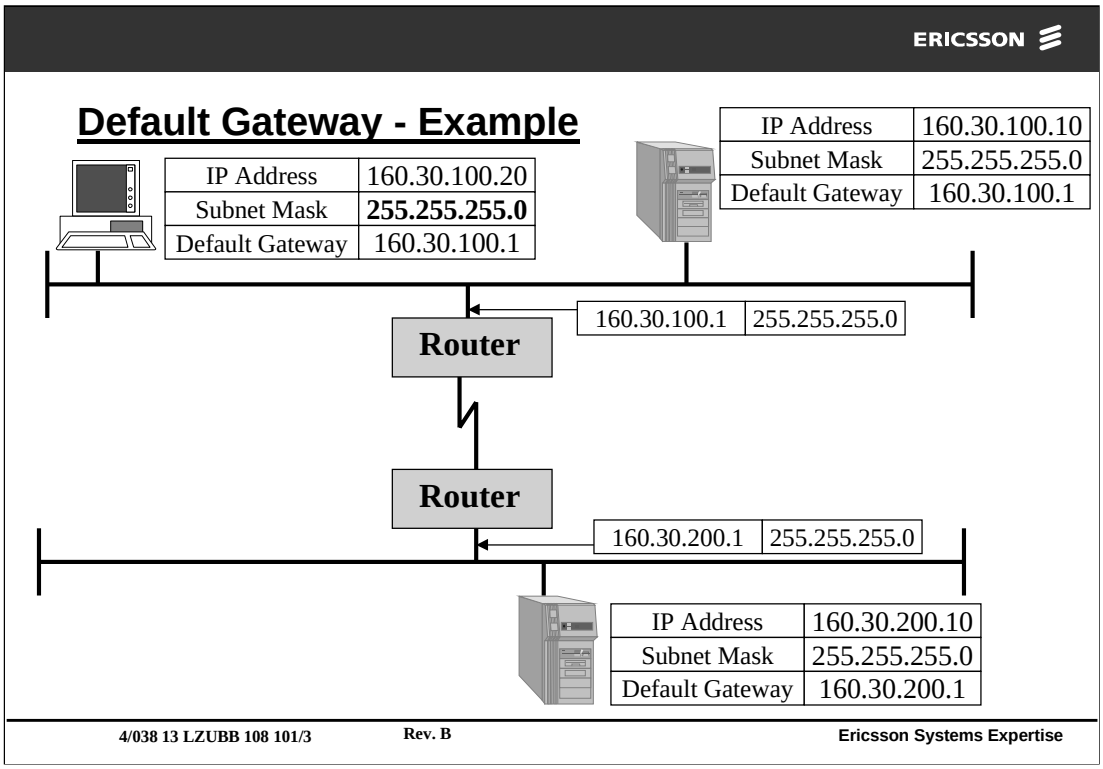
IP routing prerequisites

- **The sending device must send the information to an IP router**
  - **Achieve this by using a default gateway or proxy ARP**
- **The IP router must know how to transfer the information to the remote network, via other routers if necessary**
  - **Achieve this by using routing algorithms, for example, RIP or OSPF.**

When a source device wants to communicate with a destination device on a different network, it must pass the data to an IP router. The IP router must know how to transfer this information on to the remote network.

The sending device will either use a default gateway or rely on proxy ARP to get the information to the router. In this chapter we will examine both of these methods.

The router uses a routing algorithm to find the best route to a remote network. There are many different routing algorithms, in this chapter we will examine three of the most frequently used : RIP (Routing Information Protocol), OSPF (Open Shortest Path First) and BGP (Border Gateway Protocol).



Default Gateway

Most TCP/IP devices are configured with the IP address of one or more default gateways (sometimes referred to as a default router).

When a device wants to transmit a packet it compares the destination's network ID and subnet ID, with its own network ID and subnet ID. If the two match, the device sends the packet directly to the destination, using ARP if necessary.

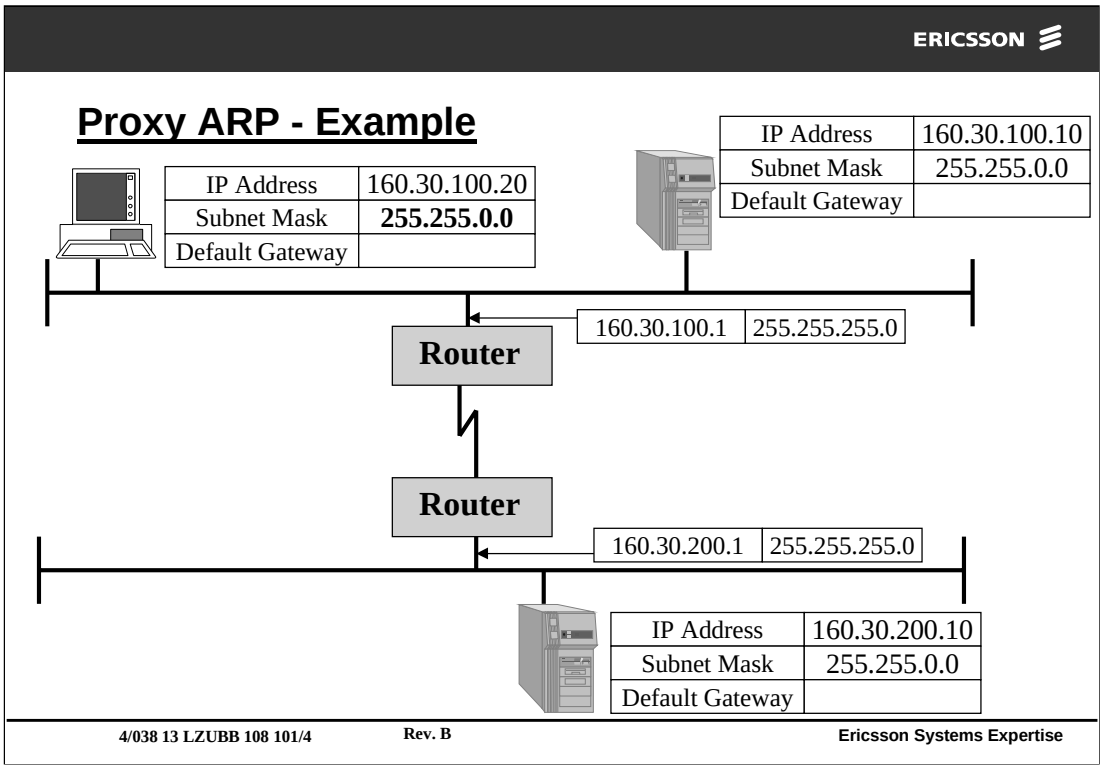
(Note: ARP is the mechanism that maps IP addresses to hardware addresses, as described in chapter 1)

If the destination's network ID and subnet ID, and the source's network ID and subnet ID, do not match, the source device sends the packet to the hardware address of its designated default gateway, again using ARP if necessary.

Instead of looking for the IP address of the destination in its ARP cache, it looks for the IP address of its designated default gateway. If it doesn't find it, it simply uses ARP to get it. Note that it will nearly always be in the cache because all other devices not on this subnet will correspond to this one address.

This is a simple and efficient mechanism. Broadcasts are rarely needed because any device the source contacts beyond its own subnet will cause this one default gateway entry to be refreshed in the source's ARP cache.

If the default gateway itself fails, there can be a problem, but in most cases a list of default gateways can be entered in the device's IP configuration. However, there is often just a single router accessible, so if it fails nothing can be done.



Proxy ARP

In the example above the PC and servers are configured with the Class B default subnet mask, (255.255.0.0). The routers are configured with the customised mask (255.255.255.0).

If the PC wants to send a packet to the server on the remote network it will compare the destination's network ID and subnet, with its own network ID and subnet. The result implies that they are on the same network, so that the PC will try to send the packet directly using ARP.

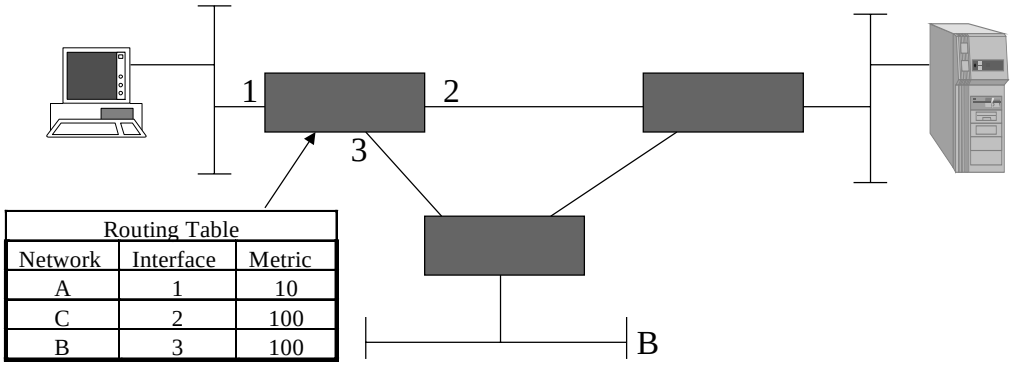
Routers do not normally propagate broadcasts so the actual ARP broadcast will not go beyond the sender's network.

However, routers can run a protocol called Proxy ARP. When a router running Proxy ARP receives an ARP request it reads the packet and applies the subnet mask for the sender's subnet to the requested destination IP address. This gives it the network ID which it compares with its routing tables looking for a match.

In the example above the router will determine that it knows a viable route to get the packet to the subnet of the destination. The router then replies to the ARP request exactly as if the router were itself the destination device. The only difference is that the hardware address returned in the ARP reply is the address of the router port connected to the source network. The source device and router now enter each other's IP/Hardware address pair in their ARP cache and the first data packet can be sent.

ERICSSON

Determination of optimal routing Path



Routing Table		
Network	Interface	Metric
A	1	10
C	2	100
B	3	100

Routing algorithms calculate route metrics to determine optimal routes to a destination

To aid the process of path determination, routing algorithms initialise and maintain routing tables, in routers. These tables typically contain three items:

• a network identifier - the network addresses of a remote network.

• an interface number - the interface that should be used to route traffic towards that particular remote network.

•a metric - a standard of measurement, for example, path length, that is used by routing algorithms to determine the optimal path to a destination.

In the routing table a network identifier may have several interface numbers with different metrics corresponding to each number. The routing algorithm compares the metrics to determine the optimal route.

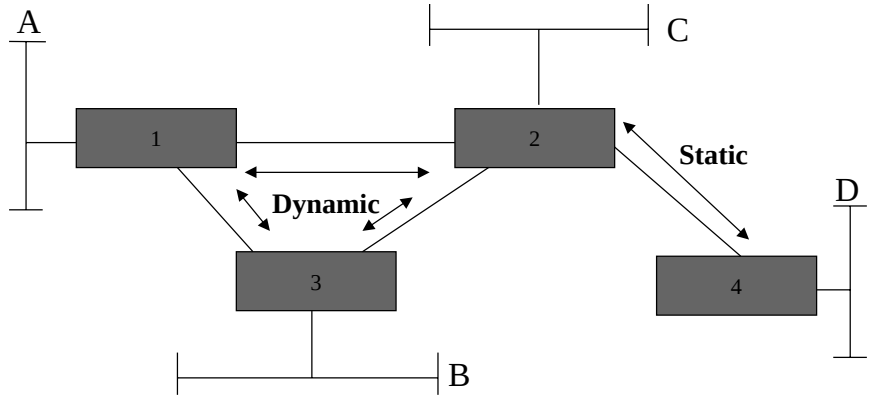
When a router receives an incoming packet, it checks the destination address, looks it up in the routing table and selects the interface (next hop on the optimal route).

Routers communicate with one another (and maintain their routing tables) through the transmission of routing update messages. By analysing routing updates from other routers a router can build up a picture of the network topology. Once the network topology is understood, routers can determine optimal routes to network destinations.

ERICSSON

Dynamic Vs Static routing

Dynamic routes - Adjust in real time to network changes by analysing routing update message.
Static Routes - Manually configured and changed by network administrator.



4/038 13 LZUBB 108 101/6 Rev. B Ericsson Systems Expertise

Static routing tables are configured by a network administrator and do not change unless the administrator changes them. They prohibit a router from offering alternative routes if a network link goes down. Static routes work well in environments where network traffic is relatively predictable and network design is relatively simple.

Dynamic routing algorithms adjust, in real time, to changing network circumstances. They do this by analysing incoming routing update messages. If the message indicates that a network change has occurred, the routing software recalculates routes and sends out new routing update messages. These messages permeate the network, causing routers to rerun their algorithms and change their routing tables accordingly.

Example: In the diagram above routers 1, 2 and 3 update each other dynamically. Router 2 has a static route configured to send all traffic destined for network D, via router 4. Similarly, Router 4 has a static route configured to send traffic destined for all networks except D, via router 2.

ERICSSON

Routing Metrics

- **Path Length / Hop count**
- **Reliability**
- **Delay**
- **Bandwidth**
- **Load**
- **Communications cost**

Metrics are used by routing algorithms to select the best route. Sophisticated routing algorithms can use a combination of the following metrics:

Path Length is the sum of the interface costs associated with each network link traversed. Hop count specifies the number of passes through internetworking products (such as routers) that a packet must take when going from source to destination.

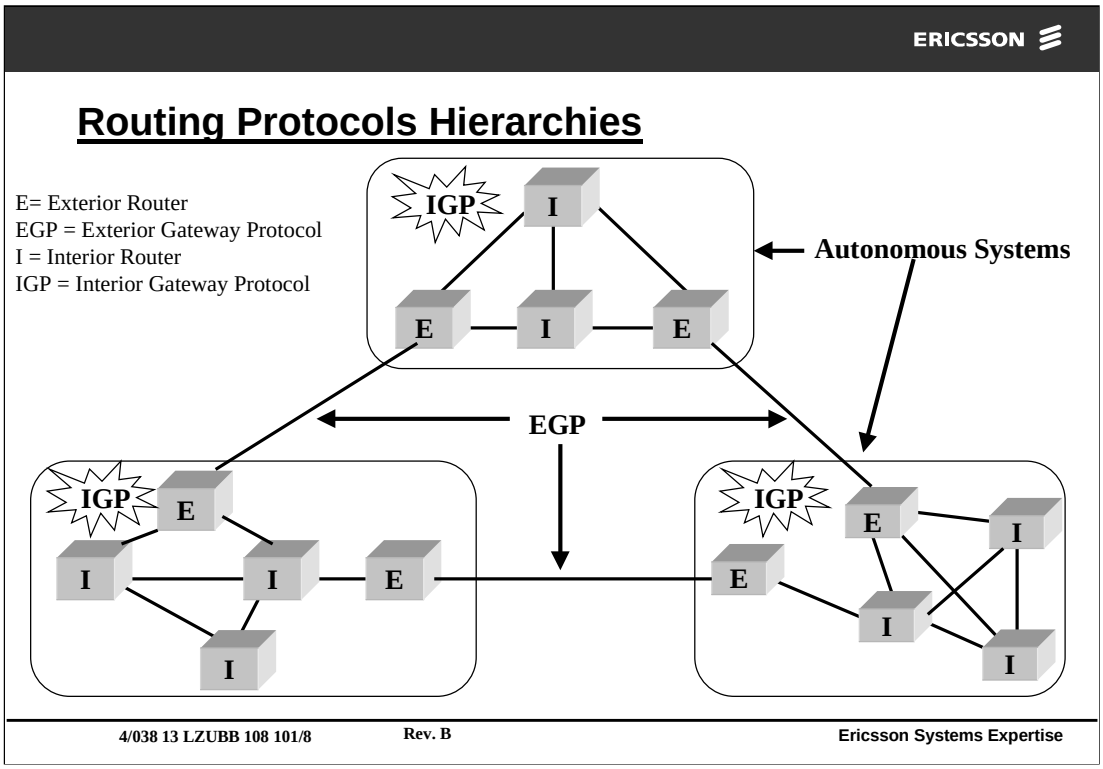
Reliability is usually assigned to network links by network administrators. The values assigned are based on how frequently the network link goes down and how long it typically takes to be repaired.

Delay refers to the length of time to move a packet from source to destination through an internetwork. It is dependent on many factors including the bandwidth of intermediate network links, the port queues at each router along the way, network congestion on all intermediate network links, and the physical distance to be travelled.

Bandwidth refers to the available traffic capacity of a link.

Load refers to the degree to which a network resource (such as a router) is busy, for example, its CPU utilisation and packets processed per second.

Communications cost is the actual financial cost associated with a particular route. A network administrator may configure routers so that traffic uses a slower link if it is cheaper to do so.



Routers within the Internet are organised in a hierarchy. Some routers are used to move information through one particular group of networks under the same administrative authority and control, such an entity is called an Autonomous System. Routers used for information exchange within autonomous systems are called interior routers, and they use a variety of interior gateway protocols (IGP's) to accomplish this purpose. Examples of IGP are RIP, Interior Gateway Routing Protocol (IGRP), Enhanced - IGRP (EIGRP), Open Shortest Path First (OSPF), and Intermediate System to Intermediate System Intra-Domain Routing Exchange Protocol (IS-IS).

Exterior Gateway Protocols, such as Border Gateway Protocol (BGP), were developed because IGP's do not scale in networks that go beyond the enterprise level. IGPs were never designed for the purpose of global internetworking and they are unable to segregate enterprises into different administrations that are technically and politically independent from one another. Exterior routers move information between autonomous systems using exterior gateway protocols.

IGP's and EGPs allow a hierarchy of routers to be compiled in order that all routers have their route table size kept as small as possible.

ERICSSON

Distance Vector Algorithm (e.g. RIP)

- **Each Router sends all of its routing table**
  - **to its neighbours only**
  - **at pre-defined intervals.**
- **Advantages**
  - **Simple configuration**
- **Disadvantages**
  - **Slow convergence**
  - **Quantity of update traffic**
  - **Limits the size of internetwork**

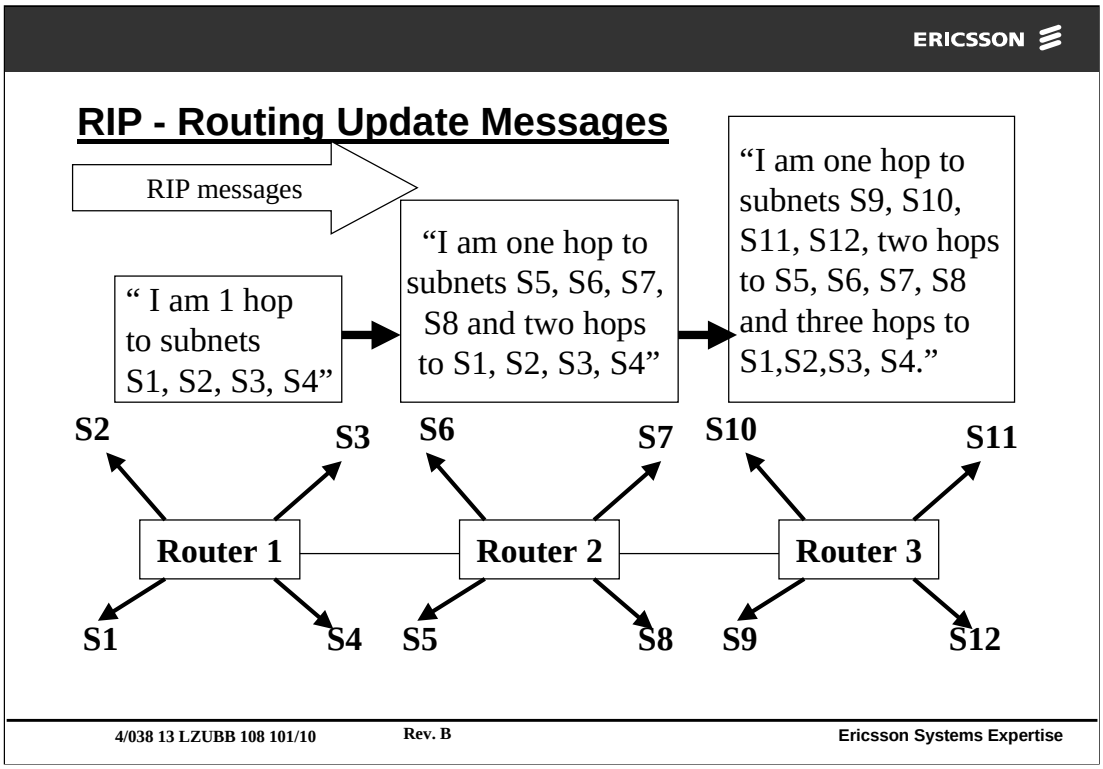
Routers using Distance Vector Algorithms (DVAs), for example, Routing Information Protocol (RIP), build their own “map” of the network and send this map to all neighbouring routers. As the maps are transferred through the network, each router progressively incorporates this second-hand information into its own map of the network.

In large networks, building an accurate map using DVAs can take a long time, and during this period some routers will have old and incorrect maps.

Each routing table entry learned through RIP is given a time-out value of 3 minutes (by default) past the last time it was received in a RIP advertisement. When a router goes down, it can take several minutes for the changes to be propagated through the network. This is known as the “slow convergence problem”.

DVA routers periodically broadcast their entire routing table, even if the routing tables have not changed. This typically occurs every thirty seconds. These broadcasts consume a large amount of bandwidth.

The simple metrics used in DVAs limit the size of an internetwork, for example, RIP permits a maximum hop count of 15.

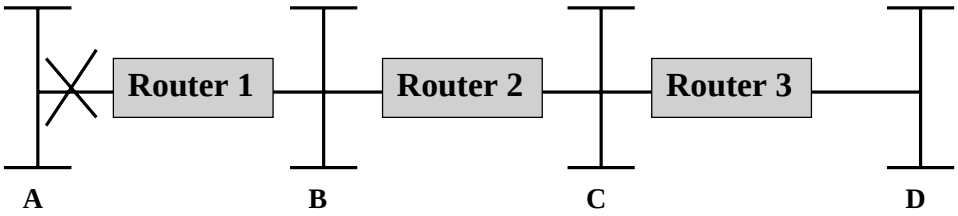


RIP calculates the best route to use by measuring the hop count only. Hop count is the quantity of routers that packet must pass through to reach a remote network.

Each RIP routing update message contains how far (in hops) each subnet is from the sending station. The messages are sent only between adjacent stations (and never backwards) so each station adds one hop to the values.

ERICSSON

Slow Convergence



Network	Hops
A	1
B	1
C	2
D	3

Network	Hops
A	2
B	1
C	1
D	2

Network	Hops
A	3
B	2
C	1
D	1

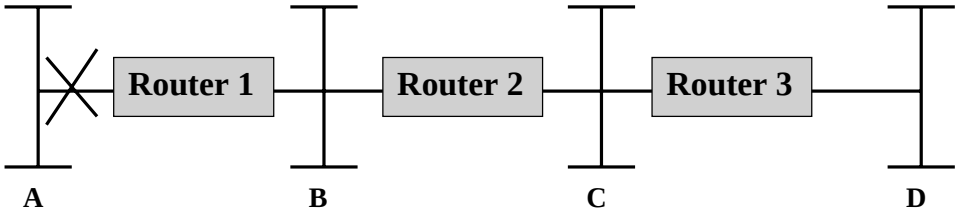
RIP is unable to detect loops and uses a hop count of 15 to denote infinity. This means that packets could circulate around a loop until the hop count reached 15. When the hop count exceeds 15, the route is marked unreachable.

The slow convergence problem can make routers wrongly believe they have a connection to a network, after the connection has failed.

In the example shown above, R1 is aware that its link to network A has failed. However, R2 continues to broadcast RIP messages stating that it can reach network A in 2 hops. Router 1 then assumes it can reach network A via R2 and changes its routing table to show the new route. Both R1 and R2 will continue to exchange RIP messages, increasing the hop count to network A each time, until the count reaches 15 and it is assumed to be unreachable.

ERICSSON

Split Horizons



Network	Hops
A	1
B	1
C	2
D	3

Network	Hops
A	2
B	1
C	1
D	2

Network	Hops
A	3
B	2
C	1
D	1

- **Never send routing information backwards**

RIP specifies a feature called “Split Horizons” to make its operation more stable. Split horizons take advantage of the fact that it is never useful to send information about a route back in the direction from which it came.

In the example shown above, R1 initially advertises that it has a route to network A. there is no reason for R2 to include this route in its update back to R1, as R1 is closer to Network A. The split horizon rule says that R2 should strike this route from any updates it sends to R1.

This rule helps prevent two-node routing loops, as R2 will not inform R1 that it can get to network A through R1, after R1’s link to network A has failed.

ERICSSON

Open Shortest Path First (OSPF)

- **Each router sends only the portion of its routing table that describes its own links**
  - **to all routers on the internetwork**
  - **when network changes occur**
- **Advantages**
  - **Fast Convergence**
  - **Conserve Network Bandwidth**
  - **Route Selection based on a cost metric**
  - **Heavy Memory use**
  - **CPU Utilisation**
  - **Expensive on bandwidth if frequent network changes**

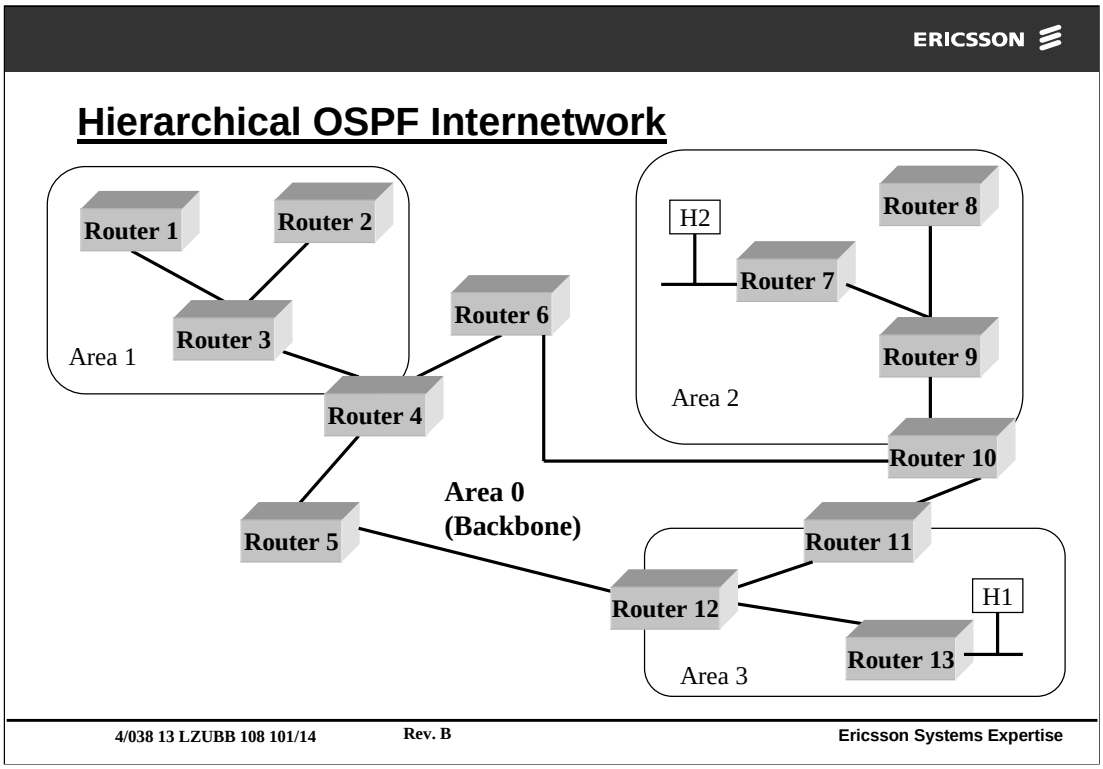
Open Shortest Path First (OSPF) routing protocol is an example of a link state algorithm, used by routers, that adjusts to network changes quicker than RIP and is more robust. Each router updates the rest of the network with information on the direct connections it has to its neighbours. OSPF is reliable because information is transferred unaltered between routers.

Link-state protocols, of which OSPF is one, broadcast **link state advertisements (LSA)** to their neighbours only when a network change has occurred. They also send an update following a large interval, typically every hour.

Unlike RIP, which is based entirely on “hop count”, OSPF uses a cost metric to select the best route. Network administrators can define a preferred path by assigning interface costs when configuring OSPF.

Link-state protocols use large amounts of router memory to store topological databases, as each router keeps a map of the entire network.

When a network experiences frequent changes, link-state routers use a large portion of network bandwidth by sending out LSAs at each network change. After receiving a new LSA, the router must run the Shortest Path First (SPF) algorithm and generate a new routing table. This process places heavy demands on the router’s CPU.



Routing Hierarchy

Unlike RIP, OSPF can operate within a hierarchy. The largest entity within the hierarchy is an A*utonomous System* (AS). An AS is a collection of networks under a common administration, sharing a common routing strategy. OSPF is an intra-AS (interior gateway) routing protocol, although it is capable of receiving routes from and sending routes to other AS.

An AS can be divided into a number of *areas.* An area is a group of contiguous networks and attached hosts. Routers with multiple interfaces can participate in multiple areas. These routers, which are called *area border routers,* maintain separate topological databases for each area.

A *topological database* is essentially an overall picture of networks in relation to routers. The topological database contains the collection of LSAs received from all routers in the same area. Because routers within the same area share the same information, they have identical topological databases.

The term *domain* is sometimes used to describe a portion of the network in which all routers have identical topological databases.

An area's topology is invisible to entities outside the area. By keeping area topologies separate, OSPF passes less routing traffic than it would if the AS were not partitioned.

Area partitioning creates two different types of OSPF routing, depending on whether the source and destination are in the same or different areas. Intra-area routing occurs when the source and destination are in the same area; interarea routing occurs when they are in different areas.

In the diagram shown above, Router 4,5,6,10,11 and 12 make up the backbone. If host H1 in area 3 wishes to send a packet to host H2 in area 2, the packet is sent to Router 13, which forwards the packet to Router 12, which sends the packet to Router 11. Router 11 forwards the packet along the backbone to area border router Router 10, which sends the packet through two intra-area routers (Router 9 and Router7) to be forwarded to host H2.

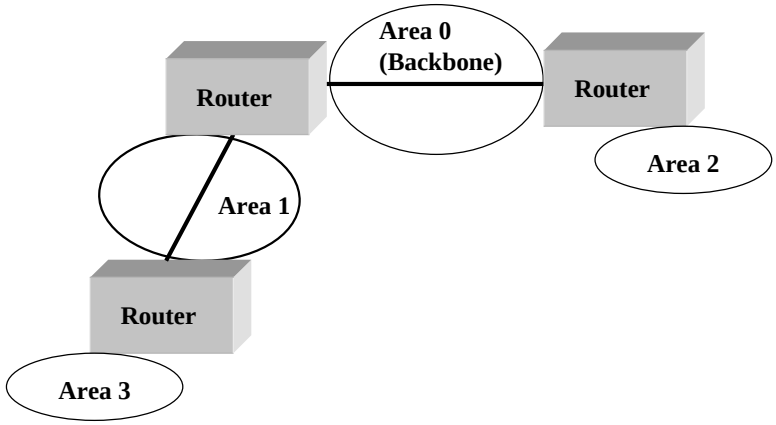
The backbone itself is an OSPF area, so all backbone routers use the same procedures and algorithms to maintain routing information within the backbone that any area router would. The backbone topology is invisible to all intra-area routers, as are individual area topologies to the backbone.

Areas can be defined in such a way that the backbone is not contiguous. In this case, backbone connectivity must be restored through *virtual links*. Virtual Links are configured between any backbone routers that share a link to a nonbackbone area, and function as if they were direct links.

Autonomous System (AS) border routers running OSPF learn about exterior routes through *exterior gateway protocols* (EGPs) such as Exterior Gateway Protocol (EGP) or Border Gateway Protocol (BGP), or through configuration information.

ERICSSON

The Backbone and Virtual Links



- **Backbone centre of communication**
- **Virtual links provide path of backbone**
- **Avoid configuring virtual links if possible**

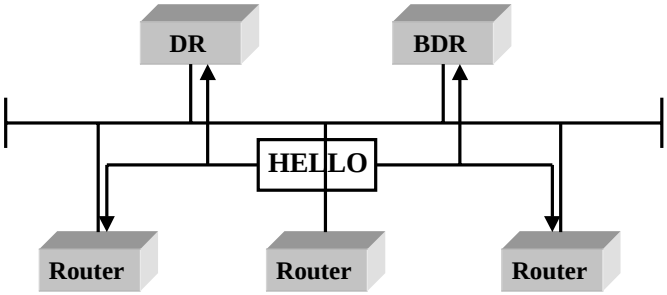
OSPF has certain restrictions when multiple areas are configured. One area must be defined as area 0. Area 0 is also called the backbone because all communication must go through it. Ideally, all areas should be physically connected to area 0. If they are not physically connected to area 0 then they should be logically connected to it. This is because all other areas inject routing information into area 0, which in turn disseminates that information to other areas.

In special cases where a new area is added after the OSPF network has been designed and configured, it is not always possible to provide that new area with direct access to the backbone. In these cases, a virtual link will have to be defined to provide that new area with direct access to the backbone. In these cases, a virtual link will have to be defined to provide the needed connectivity to the backbone. The virtual link provides the disconnected area a logical path to the backbone. The virtual link must be established between two routers that share a common area, and one of these routers must be connected to the backbone.

ERICSSON

Hello Protocol

- Acquire neighbours
- Elect designated router (DR) and back up designated router (BDR)
- Establish adjacencies



4/038 13 LZUBB 108 101/16 Rev. B Ericsson Systems Expertise

Hello Protocol

The SPF algorithm is the basis for OSPF operations. When an SPF router is powered up, it initialises its routing protocol data structures and then waits for indications from lower-layer protocols that its interfaces are functional.

Once a router is assured that its interfaces are functioning, it uses the OSPF *Hello protocol* to acquire neighbours. Neighbours are routers with interfaces to a common network. The router sends multicast hello packets to its neighbours and receives their hello packets. In addition to helping acquire neighbours, hello packets also act as keepalives to let routers know that other routers are still functional. On *multi-access networks* (networks supporting more than two routers), the hello protocol elects a designated router and a backup designated router. The designated router is responsible, among other things, for generating LSAs for the entire multi-access network. Designated routers allow a reduction in network traffic and in the size of the topological database.

When the link state databases of two neighbouring routers are synchronised, the routers are said to be *adjacent.* Adjacencies are established with some subset of router's neighbours. Routers connected by point-to-point networks, point-to-multipoint networks and virtual links always become adjacent. On broadcast and Non-Broadcast Multi Access (NBMA), all routers become adjacent to both the designated router and the back up designated router. Topological databases are synchronised between pairs of adjacent routers. Adjacencies control the distribution of routing protocol packets. These packets are sent and received only on adjacencies.

Each router periodically sends an LSA. LSAs are also sent when a router's state changes. LSAs include information on a router's adjacencies. By comparing established adjacencies to link states, failed routers can be quickly detected and the network's topology altered appropriately. From the topological database generated from LSAs, each router calculates a shortest-path tree, with itself as root. The shortest-path tree, in turn, yields a routing table.

ERICSSON

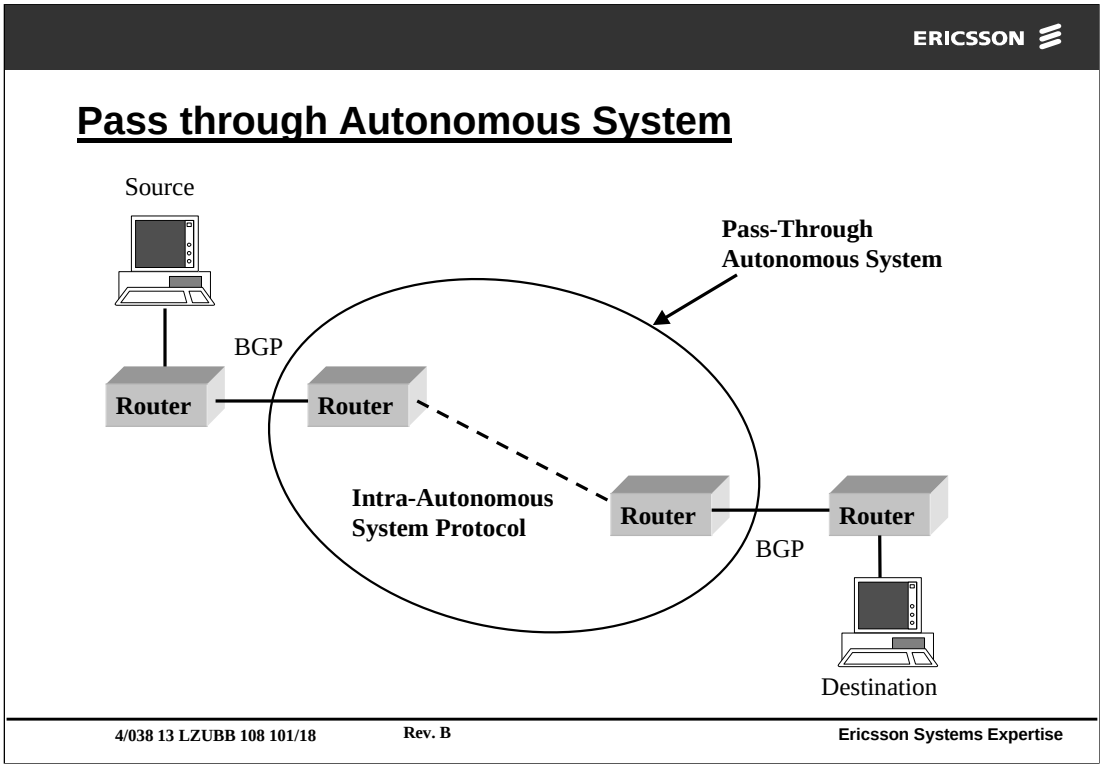
Border Gateway Protocol (BGP)

- **The BGP was designed to route between routing domains.**
- **BGP is an Exterior Gateway Protocol (EGP)**
- **BGP performs three types of routing:**
  - **interautonomous system routing**
  - **intra-autonomous system routing**
  - **pass-through autonomous system routing**

BGP performs interdomain routing in TCP/IP networks. BGP is an Exterior Gateway Protocol (EGP), which means that it performs routing between multiple autonomous systems or domains and exchanges routing and reachability information with other BGP systems.

BGP was developed to replace its predecessor, the now-obsolete Exterior Gateway Protocol (EGP), as the standard exterior gateway-routing protocol used in the global Internet. BGP solves serious problems with EGP and scales to Internet growth more efficiently.

The BGP was designed to route between routing domains. In the terminology of the Internet, a routing domain is called an autonomous system (AS).



Although BGP was designed an inter-AS protocol, it can be used both within and between Autonomous Systems. In total BGP performs three types of routing: *interautonomous system routing*, *intra-autonomous system routing*, and *pass-through autonomous system routing*.

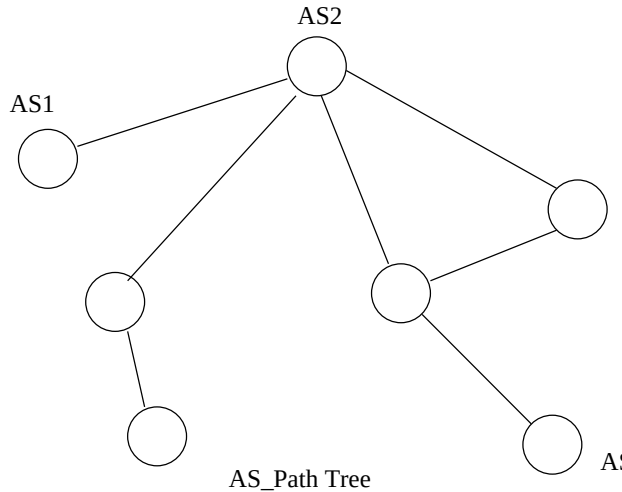
Interautonomous system routing occurs between two or more BGP routers in different autonomous systems. Peer routers in these systems use BGP to maintain a consistent view of the internetwork topology. BGP neighbours communicating between autonomous systems must reside on the same physical network. BGP is frequently used to provide path determination to provide optimal routing within the Internet.

Intra-autonomous system routing occurs between two or more BGP routers located within the same autonomous system. Peer routers within the same autonomous system use BGP to maintain a consistent view of the system topology. BGP is also used to determine which router will serve as the connection point for specific external autonomous systems. Once again, the Internet provides an example of interautonomous system routing. The BGP protocol can provide both inter- and intra-autonomous system routing services.

Pass-through autonomous system routing occurs between two or more BGP peer routers that exchange traffic across an autonomous system that does not run BGP. In a pass-through autonomous system environment, the BGP traffic did not originate within the autonomous system in question and is not destined for a node in the autonomous system. BGP must interact with whatever intra-autonomous system routing protocol is being used to successfully transport BGP traffic through that autonomous system.

ERICSSON

AS_Path Tree



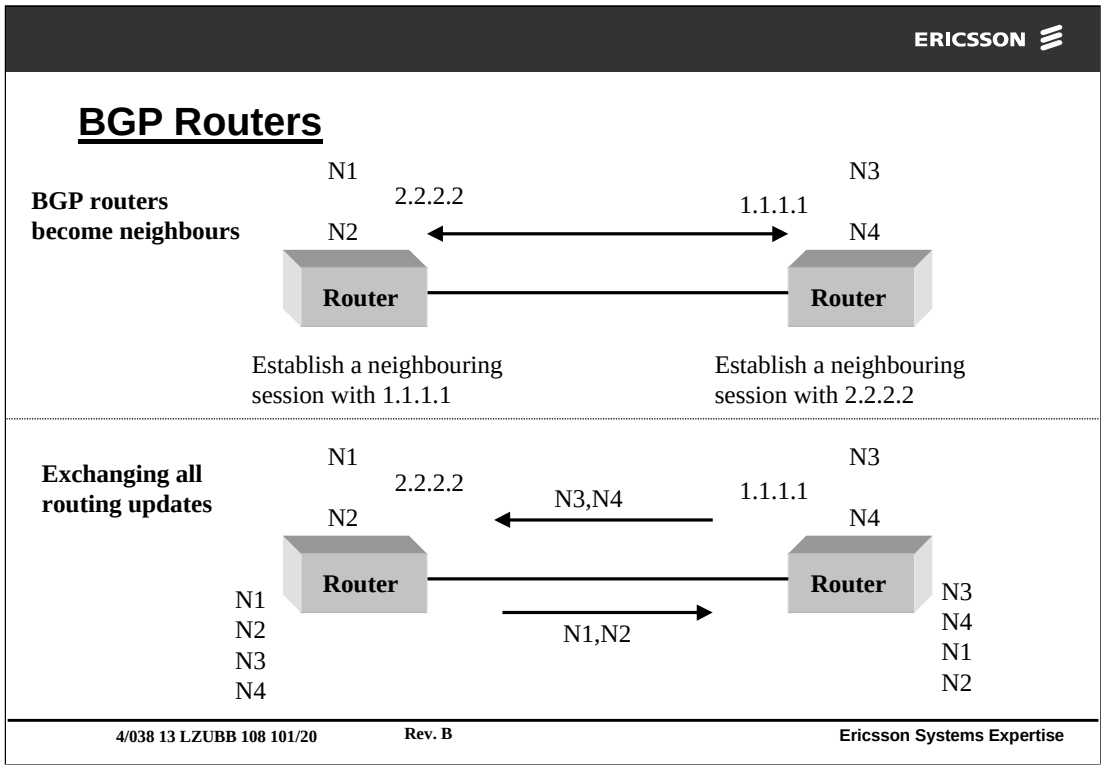
4/038 13 LZUBB 108 101/19 Rev. B Ericsson Systems Expertise

BGP Operation

As with any routing protocol, BGP maintains routing tables, transmits routing updates, and bases routing decisions on routing metrics. The primary function of a BGP system is to exchange network-reachability information, including information about the list of autonomous system paths, with other BGP systems. This information can be used to construct a graph of autonomous system connectivity from which routing loops can be pruned and with which autonomous system-level policy decisions can be enforced.

BGP constructs a graph of ASs based on the information exchanged between BGP neighbours. This graph is also referred to as a tree. BGP views the Internet as a collection of ASs, with each AS identified by an AS number. Connections between two ASs together form a path, and the collection of path information forms a route to reach a specific destination.

BGP is a path vector protocol used to carry routing information between autonomous systems. The term *path vector* comes from the fact that BGP routing information carries a sequence of AS numbers, which indicates the path a route has traversed. BGP uses TCP as its transport protocol. This ensures that all the transport reliability such as retransmission is taken care of by TCP and does not need to be implemented in BGP itself.

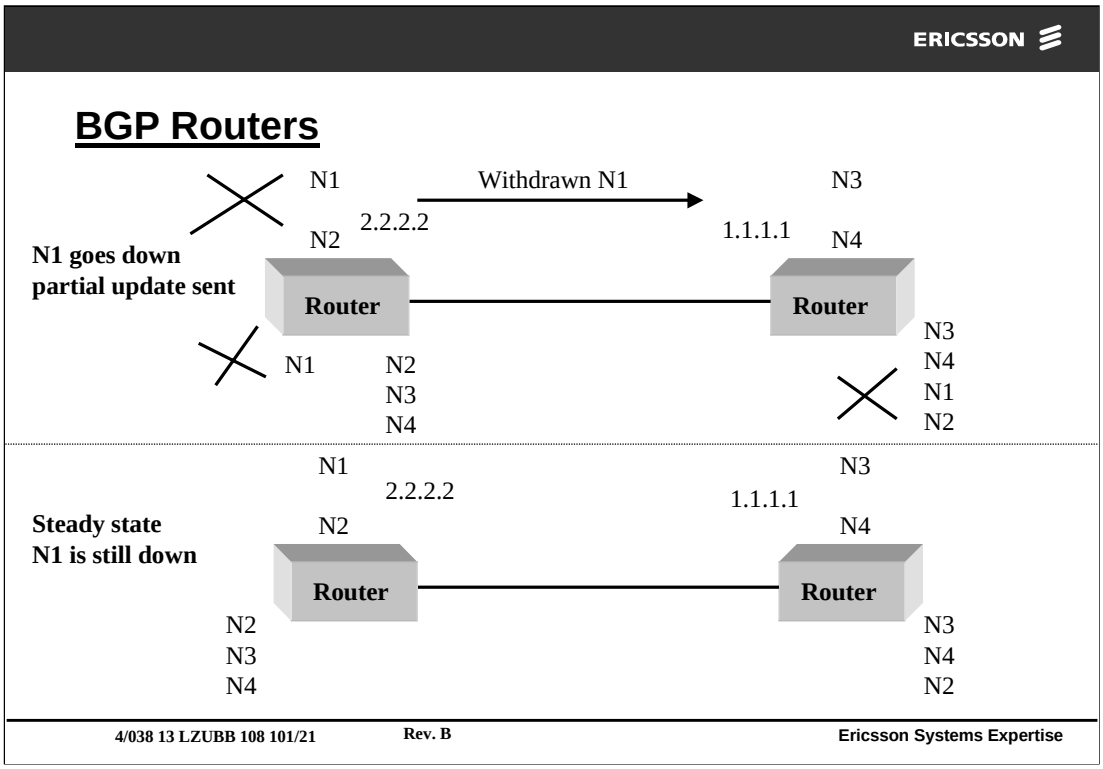


The following is an example on how BGP operates.

Two BGP routers form a transport protocol connection between each other. These routers are called *neighbours or peers.* The diagram above illustrates this relationship. Peer routers exchange multiple messages to open and confirm the connection parameters, such as the BGP version running between the two peers. In case of any disagreement between the peers, notification errors are sent, and the peer connection does not get established.

The initial data exchange between two routers is the entire BGP routing table as shown in the diagram above. Incremental updates are sent as network information changes. The incremental update approach has shown an enormous improvement as far as CPU overhead and bandwidth allocation compared with complete periodic updates used by the previous protocols such as EGP.

Routes are advertised between a pair of BGP routers in UPDATE messages. The update messages consist of network number and AS path pairs. The AS path contains the string of ASs through which the specified network can be reached. The UPDATE message also contains the path attributes, which include such information as the degree of preference or metric for a particular route. These metrics are typically assigned by a network administrator.



In case of information changes, such as route being unreachable or having a better path, BGP informs its neighbours by withdrawing the invalid routes and injecting new routing information. As shown in the diagram above, withdrawn routes are part of the UPDATE message. These are the routes that are no longer available for use. The diagram above also shows a steady state situation, if no routing changes occur, the routes exchange only KEEPALIVE packets.

KEEPALIVE messages are sent periodically between BGP neighbours to ensure that the connection is kept alive. KEEPALIVE packets should not cause any strain on the router CPU or link bandwidth as they consume a minimal bandwidth.

BGP keeps a table version number to keep track of the instance of the BGP routing table. If the table changes, BGP will increment the table version. A table version that is incrementing rapidly is usually an indication of instabilities in the network.

Summary

•In chapter 4 we looked at how routing works. We discussed the difference between default gateway and proxy ARP.

•We discussed the two types of routing, static and dynamic routing.
•We examined three routing protocols, namely RIP, OSPF and BGP.
•We explained the operation of RIP, OSPF and BGP.

Chapter 5

Application Protocols

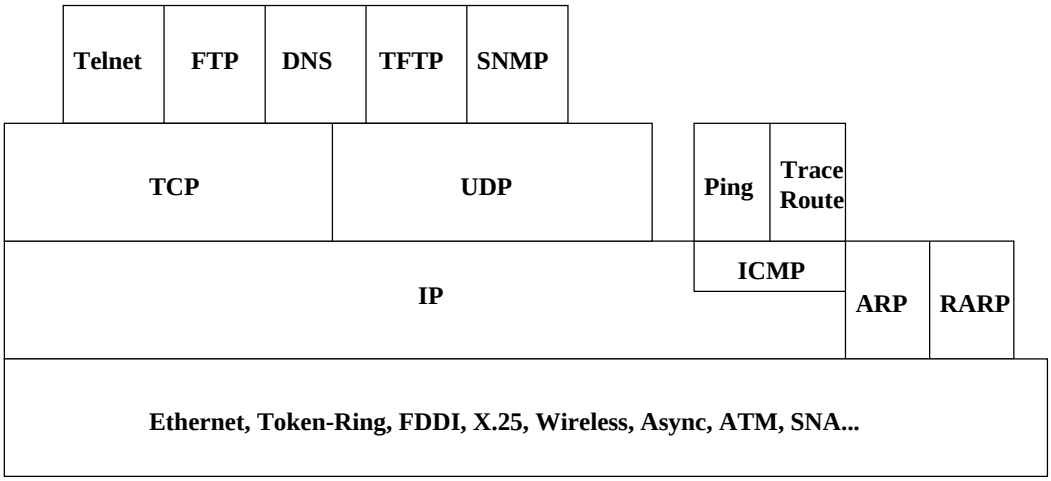
ERICSSON

Chapter 5. Application Protocols

- **After completing this chapter the student will be able to identify and describe in detail the following application protocols:**
  - **Telnet**
  - **FTP - File Transfer Protocol**
  - **TFTP - Trivial File Transfer Protocol**
  - **DNS - Domain Name System**
  - **SMTP - Simple Mail Transfer Protocol**
  - **SNMP - Simple Network Management Protocol**
  - **WWW - World Wide Web**

ERICSSON

Application Protocols



5/038 13 LZUBB 108 101/2 Rev. B Ericsson Systems Expertise

The TCP/IP protocol suite includes application protocols such as:

- Telnet for interactive terminal access to remote internet hosts
- FTP (File Transfer Protocol) for high-speed disk-to-disk file transfer
- SMTP (Simple Mail Transfer Protocol) as an internet mailing system.

These application protocols use either UDP or TCP as a transport mechanism. Remember that UDP is unreliable and offers no flow control, so in this case the application has to provide its own recovery and flow control routines. However, most application protocols use TCP, but there are applications built on UDP to provide better performance through reduced protocol overhead.

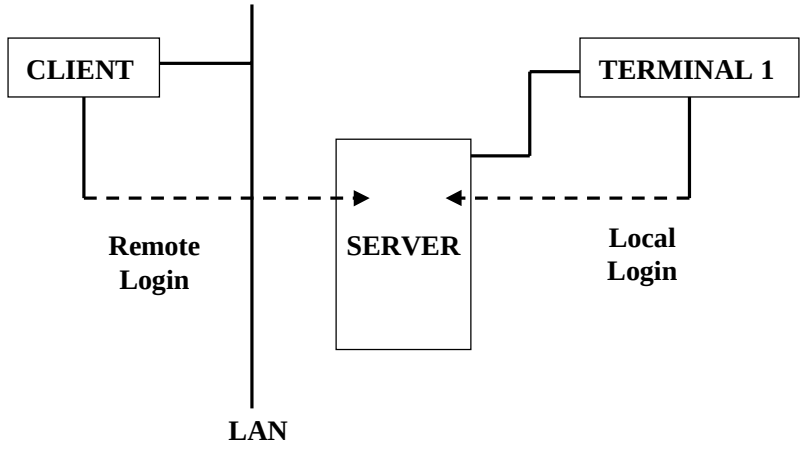
Most of the application protocols use the client/server model of interaction.

A server is an application that offers a service to Internet users. A client is a requester of a service. An application consists of both a server and a client part, which can run on the same or on separate systems.

Users usually invoke the client part of the application, which builds a request for a particular service and sends it to the server part of the application using TCP/IP as a transport vehicle.

ERICSSON

TELNET



5/038 13 LZUBB 108 101/3 Rev. B Ericsson Systems Expertise

The Telnet protocol provides a standardised interface, through which a program on one host (the Telnet client) may access the resources of another host (the Telnet server) as though the client were a local terminal connected to the server. Telnet is also used for logging into bridges, routers and other network devices for management purposes such as configuration.

For example, a user on a workstation on a LAN may connect to a host attached to the LAN as though the workstation were a terminal attached directly to the host. Of course, Telnet may be used across WANs as well as LANs.

Telnet works as follows:

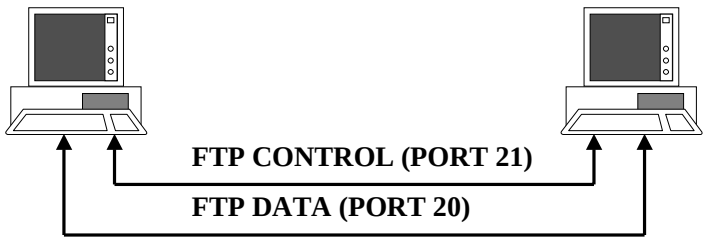
• A user accesses the Telnet application and selects the destination host (IP address or computer name).

• During the login session, the user interacts with the local Telnet service. The local Telnet sets up a TCP connection to the Telnet server (on port 23) at the remote host.

• The local Telnet exchanges data with the remote Telnet service. The remote Telnet service interacts with applications at the destination host. It is up to the local (client) and remote (server) Telnets to create the appropriate terminal emulation that enables the local terminal to talk to the remote application.

ERICSSON

FTP- File Transfer protocol



FTP is used to transfer/copy files from one machine to another. FTP allows anyone on the Internet to log in and download whatever files have been placed on an FTP server.

FTP uses TCP as a transport protocol to provide reliable end-to-end connections. Two connections are used: one for FTP control information, port 21 and one purely for the data being transferred, port 20.

As soon as a session is requested, TCP attempts to open a connection to a remote port (21). A user-name, password, all of the commands issued by the user and responses to them are passed over this connection.

If authentication is successful, an end-to-end connection is opened between port 20 on the client and server side for the data transfer.

ERICSSON

TFTP- Trivial File Transfer Protocol

- **TFTP is an extremely simple protocol to transfer files**
- **Communication between a TFTP client and server uses UDP (port 69) not TCP**
- **TFTP does not have authorisation**
- **TFTP always sends 512 byte blocks of data**

There exists a second file transfer service known as Trivial File Transfer Protocol (TFTP). TFTP is an extremely simple protocol to transfer files. TFTP differs from FTP in several ways.

•First communication between a TFTP client and server uses UDP (port 69) and not TCP. Therefore it lacks the FTP/TCP features such as sequencing, windowing and full duplex operation.

•TFTP only supports file transfer. That is, TFTP does not support interaction and does not have a large set of commands.

•TFTP does not have authorisation. A client does not send a login name or password; a file can be transferred only if the file permissions allow global access.

Although TFTP is not as powerful as FTP, TFTP has two advantages:

• TFTP can be used in environments where UDP only is available.

• The code for TFTP requires less memory than the code for FTP.

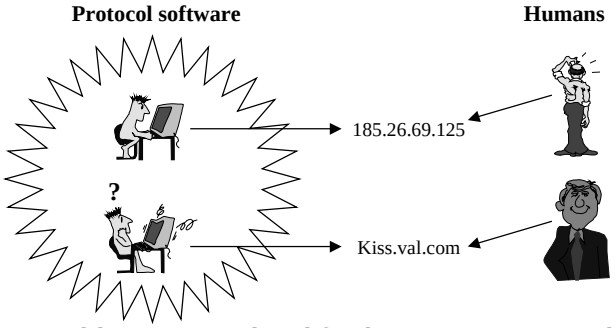
Protocol Description

Any data transfer begins with a request to read or write a file. If the server grants the request, the connection is opened and the file is sent in blocks of 512 bytes. Blocks of the file are numbered consecutively, starting at 1. Each data packet must be acknowledged by an acknowledgement packet before the next one can be sent. Note that it is the application (TFTP) that handles the acknowledgement, not the transport layer. Termination of the transfer is assumed to be on a data packet of less than 512 bytes.

Almost all errors will cause termination of the connection (lack of reliability). If a packet gets lost in the network, a timeout occurs, after which a retransmission of the last packet (data or acknowledgement) takes place.

ERICSSON

DNS- Domain Name System



- **Internet addresses are hard for humans to remember**
  - **Easy for protocol software to work with.**
- **Symbolic names are more natural for humans**
  - **Hard for protocol software to work with.**

Each computer is assigned an Internet Protocol address that appears in each IP datagram sent to the computer. Anyone who has used the Internet knows that users do not need to remember or enter the IP addresses. Computers are also assigned symbolic names; application software allows a user to enter one of the symbolic names when identifying a specific computer.

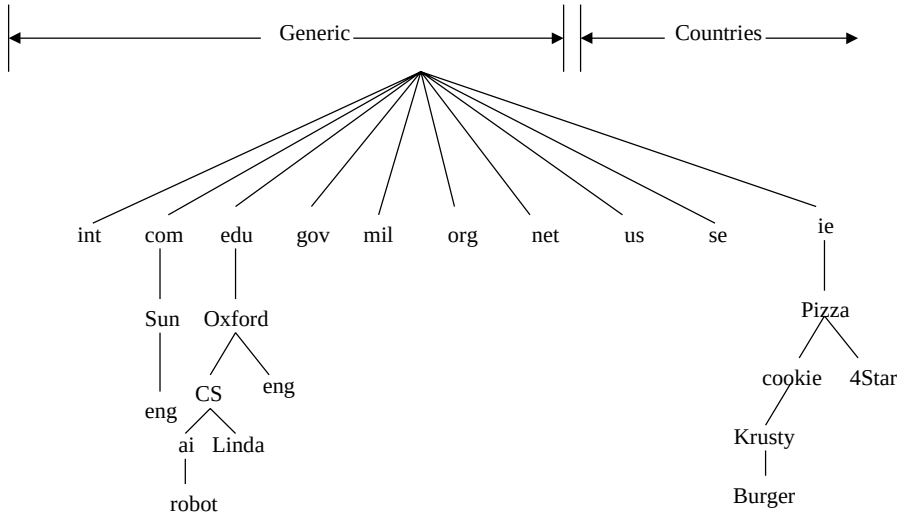
Although symbolic names are convenient for humans, they are inconvenient for computers. The underlying network protocols only understand addresses, so some mechanism to map symbolic names to IP addresses is required.

The application software translates computer names into equivalent Internet addresses. A database for implementing the naming scheme is distributed across the internet. This method of mapping the symbolic names to IP addresses through a distributed database is known as the Domain Name System (DNS).

Whenever an application program needs to translate a name, the application becomes a client of the naming system. The client sends a request message to a name server, which finds the corresponding address and sends a reply message. If a name server cannot answer a request, it temporarily becomes the client of another name server, until a server is found that can answer the request.

ERICSSON

Internet Domain Name Space



5/038 13 LZUBB 108 101/8 Rev. B Ericsson Systems Expertise

As previously mentioned, the naming scheme used in the Internet is called the Domain Name System (DNS). The DNS is based on a hierarchical scheme, with the most significant part of the name on the right. The leftmost segment is the name of the individual computer. Other segments in a domain name identify the group that owns the name. For example, the Burger Department at Pizza has the domain name:

Burger.Krusty.cookie.Pizza.ie

Basically the Internet is divided into hundreds of top-level domains, where each domain covers many hosts. Each domain is partitioned into subdomains, and these are further partitioned, and so on.

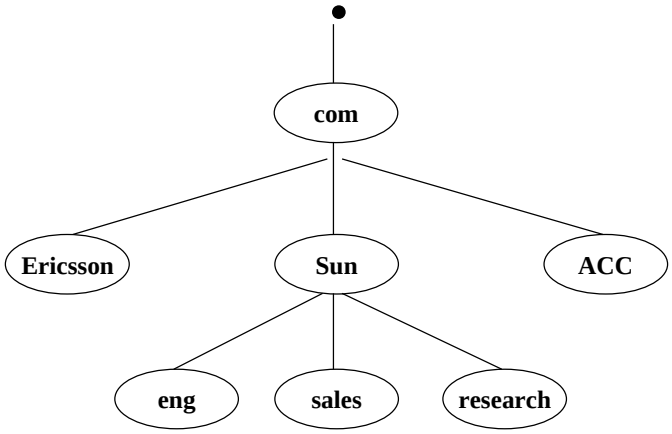
There are two types of top-level domains - generic and countries. The generic domains are:

com	**Commercial organisation**
edu	**Educational institution**
gov	**Government organisation**
mil	**Military group**
net	**Major network support center**
org	**Organisation other than those above**
int	**International organisation**

The country domains include a two letter entry for every country, as defined in ISO 3166. For example, .ie = Ireland**.**

ERICSSON

Domain Name Resolution



5/038 13 LZUBB 108 101/9 Rev. B Ericsson Systems Expertise

Each domain is named by the path upward to the root. The segments are separated by periods. For example, Sun Microsystems engineering department may have the domain name:

eng.sun.com.

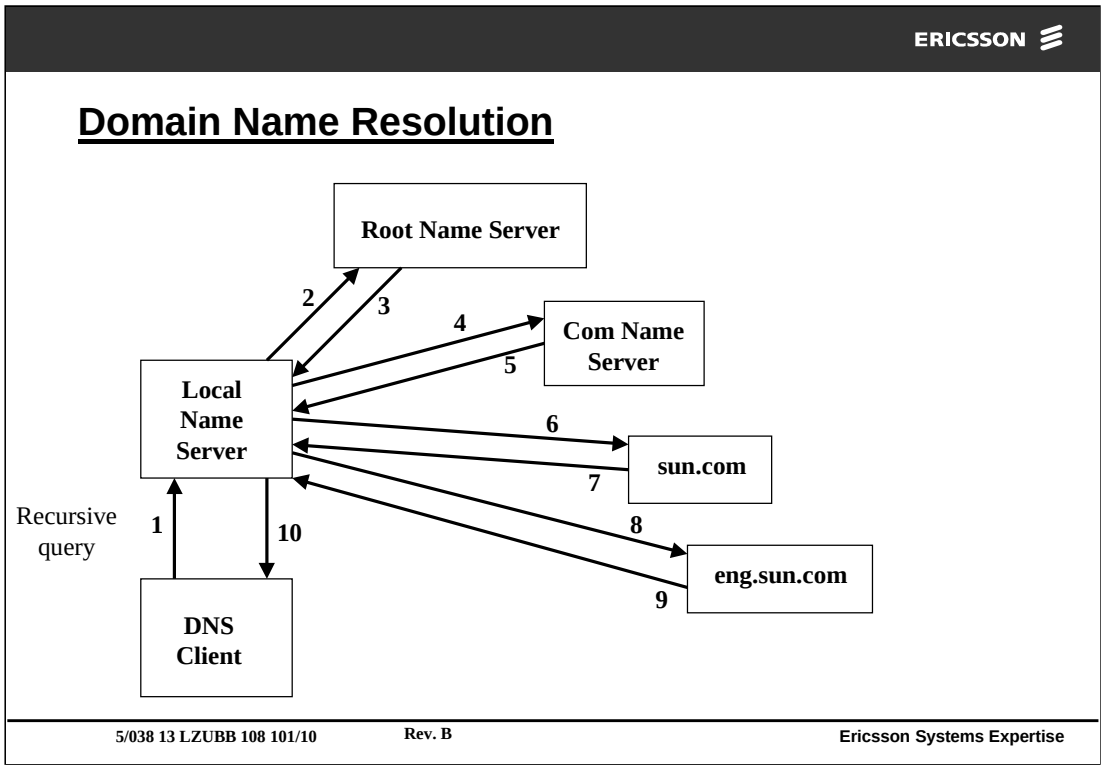
Resolving A Name

The translation of a domain name into an equivalent IP address is called name resolution, and the name is said to be resolved to an address. A host asking for DNS name resolution is called a resolver.

Each resolver is configured with the address of a local domain name server. If a resolver wishes to become a client of the DNS server, the resolver places the specified name in a DNS request message and sends the message to the local server.

The resolver then waits for the server to send a DNS reply message that contains the answer. When communicating with a DNS server, most resolvers are configured to use UDP because it requires less overhead for a single request.

When an incoming request specifies a name for which a server is an authority, the server answers the request directly. That is, the server looks up the name in its local database, and sends a reply to the resolver. However when a request arrives for a name outside the set for which the server is an authority, further client-server interaction results. The server temporarily becomes a client of another name server. When the second server returns an answer, the original server sends a copy of the answer back to the resolver from which the request arrived.



How does a DNS server know which other DNS server is the authority for a given name?

It does not know. However, each server knows the address of a root server. Knowing the location of a root server is sufficient because the name can be resolved from there.

1. The resolver sends a recursive DNS query to its local DNS server asking for the IP address of “server1.eng.sun.com”. The local name server is responsible for resolving the name and cannot refer the resolver to another name server

2. The local name server is not an authority for the name so it sends an iterative query for server1.eng.sun.com to a root name server.

3. The root name server has authority for the root domain and replies with the IP address of a name server for the com top-level domain.

4. The local name server sends an iterative query for “server1.eng.sun.com” to the com name server.

5. The com name server replies with an IP address for the name server servicing the sun.com domain.

6. The local name server sends an iterative query for “server1.eng.sun.com” to the sun.com name server.

7. The sun.com name server replies with an IP address for the name server servicing the eng.sun.com domain.

8. The local name server sends an iterative query for “server1.eng.sun.com” to the eng.sun.com name server.

9. The eng.sun.com name server replies with the IP address corresponding to server1.eng.sun.com (or an indication that no such name exists)

10. The local name server sends the IP address of “server1.eng.sun.com” to the original resolver.

ERICSSON

SMTP- Simple Mail Transfer Protocol

- **SMTP is a standard for exchange of mail between two computers**
- **SMTP is based on end-to-end delivery**
  - **The sending computer establishes a TCP connection to port 25 of the receiving computer**
  - **The receiver identifies itself**
  - **The sender announces whom the email is coming from and going to**
  - **if ok, sender sends and receiver acknowledges**

Electronic mail (E-mail) is probably the most widely used TCP/IP application. Simple Mail Transfer Protocol (SMTP) is a standard for exchange of mail between two computers. It specifies the protocol used to send mail between TCP/IP hosts.

SMTP is based on end-to-end delivery; an SMTP client contacts the destination host's SMTP server directly to deliver the mail.

Here is a typical SMTP mail transaction flow:

• The sender SMTP establishes a TCP connection with the destination SMTP (on port 25) and then waits for the server to send a *Service ready* message or a *Service not available* message when the destination is temporarily unable to proceed.

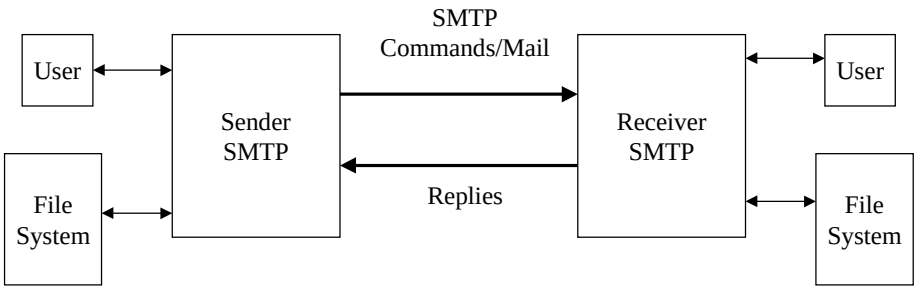
•Hello is sent, to which the receiver identifies itself by sending back its domain name. The sender SMTP can use this to verify if it contacted the correct destination SMTP.

•The sender now initiates the start of a mail transaction by sending a MAIL command to the receiver.

•The second step of the actual mail exchange involves providing the server SMTP with the destination for the message. This is done by sending one or more RCPT TO commands. Each of them receives a reply OK if the destination is known to the server, or a *No such user here* if it isn't.

ERICSSON

SMTP Mail Transaction Flow



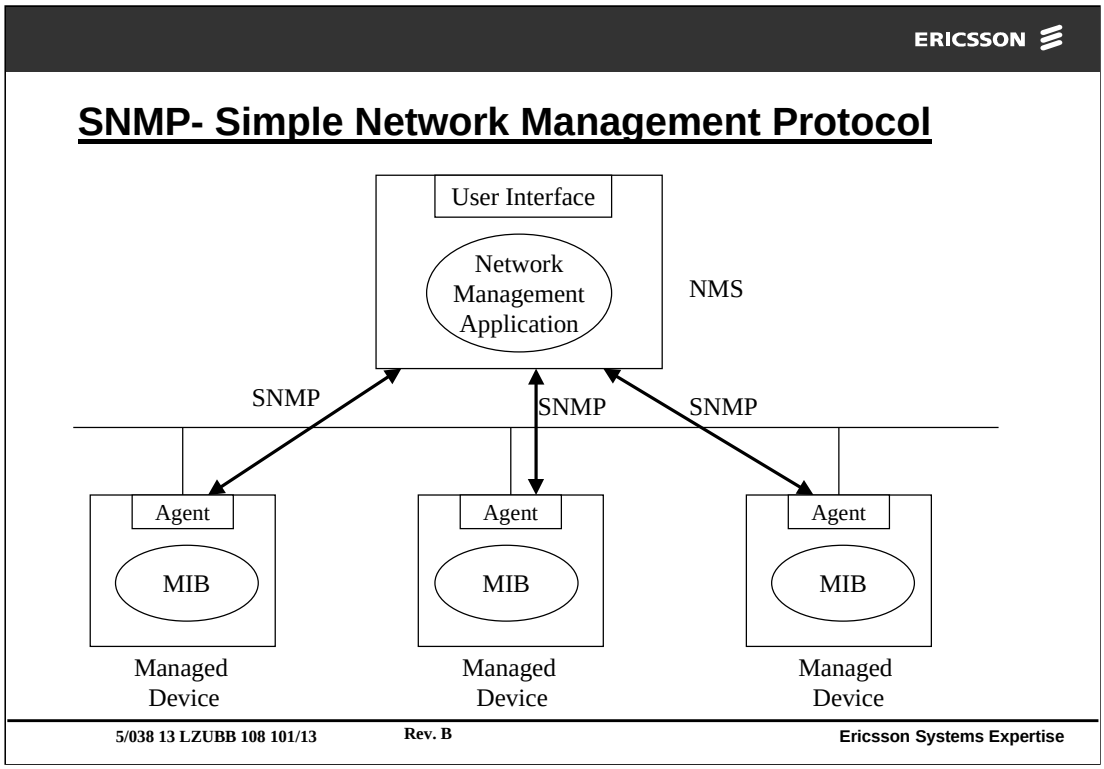
5/038 13 LZUBB 108 101/12 Rev. B Ericsson Systems Expertise

•When all RCPT commands are sent, the sender issues a DATA command to notify the receiver that the message contents are following. The server replies with *Start mail input.*

•The client now sends the data line by line, ending with a 5 character sequence <CRLF>.<CRLF> line upon which the receiver acknowledges with a OK or an appropriate error message if anything went wrong.

•We now have several possible actions:

- •The sender has no more messages to send; he ends the connection with a QUIT command, which is answered with a *Service closing transmission channel* reply.
- •The sender has no more messages to send, but is ready to receive messages from the other side. He issues the TURN command. The two SMTPs now switch their role of sender/receiver and the sender can now send messages.



The standard protocol used to manage an Internet is known as the Simple Network Management Protocol (SNMP). The SNMP protocol defines exactly how a manager communicates with an agent.

For example, SNMP defines the format of requests that a manager sends to an agent and the format of replies that an agent returns. There are two versions of SNMP. Version 1.0, which was the initial version of SNMP, and version 2.0, which incorporates security features as well as improvements in protocol operations and management architecture.

SNMP Version 1.0

In SNMP v.1, agents are software modules that run in managed devices. Agents have access to information about the managed devices in which they run and make this information available to Network Management Systems (NMSs) via SNMP v.1 as shown in the diagram above.

A managed device can be any type of node residing on a network, including computer hosts, communication servers, printers, routers, bridges and hubs. Because some of these devices may have limited ability to run management software (slow CPUs or limited memory), management software must assume the lowest common denominator. In other words, management software must be built in such a way as to minimise its own performance impact on the managed device.

Because managed devices contain a lowest common denominator of management software, the management burden falls on the NMS. Therefore, NMSs are typically engineering workstation-calibre computers that have fast CPUs, substantial memory and lots of disk space. NMSs run the network management applications that present management information to users. The user interface is typically based on a standard graphical user interface (GUI).

Communication between managed devices and NMSs is governed by SNMP. The Internet standard Network Management Framework assumes a remote-debugging paradigm, where managed devices maintain values for a number of variables and report those, on demand, to NMSs. For example, a managed device might keep track of the following:

•Number and state of its virtual circuits

•Number of certain kinds of error messages received

•Number of bytes and packets in and out of the device

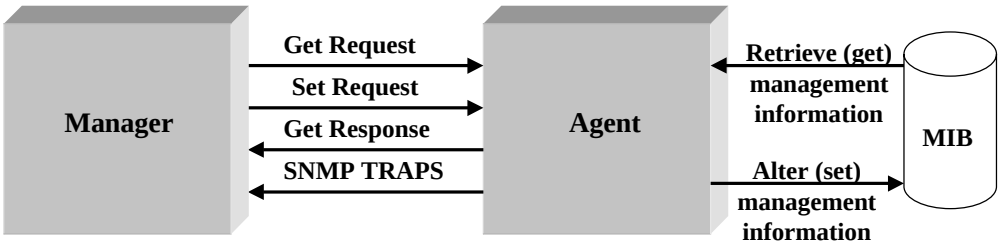
•Maximum output queue length

•Broadcast messages sent and received

•Network interfaces going down and coming up.

ERICSSON

SNMP Operations



5/038 13 LZUBB 108 101/14 Rev. B Ericsson Systems Expertise

All agents support a standard set of managed information called the Management Information Base (MIB). The MIB is a basic flat file database. Messages are sent as UDP datagrams between an SNMP manager and agent in a managed system.

Five SNMP operations are defined:

- *Get Request*: returns current MIB value for a specified item.
- *Get-next-request*: transversely returns current MIB values for a specified item.
- *Set-Request*: sets the specified MIB item to the enclosed value.
- *TRAP*: reports events such as link failures.
- *Get Response*: replies to a get request.

ERICSSON

WWW- World Wide Web

- The WWW is a large-scale, on-line repository of information that users can search using an interactive application called a browser
- The WWW was developed in 1989
- The Internet has grown immensely

5/038 13 LZUBB 108 101/15 Rev. B Ericsson Systems Expertise

The World Wide Web (WWW) is a large-scale, on-line repository of information that users can search using an interactive application program called a browser. The WWW was initially developed in 1989 by Tim Berners Lee at the European Laboratory for Particle Physics, CERN in Switzerland. In 1993 the Web started to grow rapidly due mainly to the NCSA (National Center for Supercomputing Applications) developing a Web browser program called Mosaic, an X Windows-based application. This application provided the first graphical user interface to the Web and made browsing more convenient. Today there are Web browsers and servers available for nearly all platforms.

The number of Web servers is growing very rapidly and the traffic over port 80, which is the well known Web port, on the NSF backbone has a phenomenal rate of growth too.

There are thousands of companies doing business on the Web. Most companies have a Web server in place to distribute product specific information, their portfolio or simply to get in contact with customers.

A page is just a Web term for a document and the home page is a starting point or a table of contents for a collection of documents.

ERICSSON

WWW- World Wide Web

- **The standard communication protocol between Web servers and clients is the Hypertext Transfer Protocol (HTTP)**
- **The standard language for writing Web documents is Hypertext Markup Language (HTML)**
- **Every Web page is assigned a unique URL (Uniform Resource Locator), for example:**
  - **http://www.ericsson.com/datacom/solutions**

The standard communication protocol between Web servers and clients is the Hypertext Transfer Protocol (HTTP) which is an Internet standard.

The standard language for writing Web documents is HTML (Hypertext Markup Language) which is an Internet standard and is presently under construction by several IETF working groups.

To create a Web document you must use the HTML tags to build the logical structure of the document, for example headers, lists and paragraphs.

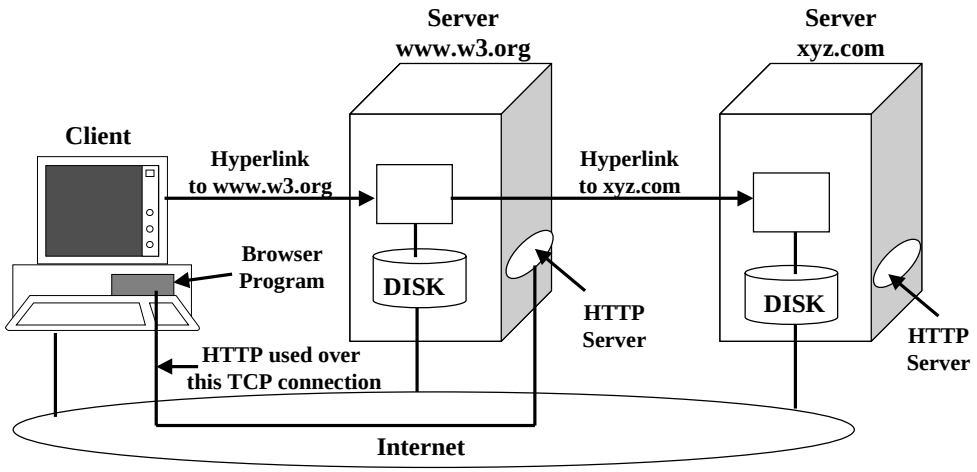
All documents, images, audio or video clips on the Web are called resources. To address and identify the access method for these resources the Web uses URLs (Uniform Resource Locators). URL is an Internet standards tracking protocol and can be found under RFC 1738.

Every Web page is assigned a unique URL (Uniform Resource Locator) that effectively serves as the page's world-wide name. URLs have three parts: the protocol, the DNS name of the host on which the page is located, and a local name uniquely indicating the specific page (usually a file name and path). For example; http://www.ericsson.com/datacom/solutions

Access Protocol **Fully qualified domain name** **File name and path**

ERICSSON

Web Operation



5/038 13 LZUBB 108 101/17 Rev. B Ericsson Systems Expertise

The steps that occur between the user's click and the page being displayed are as follows:

1. The browser determines the URL (by seeing what was selected).
2. The browser asks DNS for the IP address of www.w3.org.
3. DNS replies with 18.23.0.23.
4. The browser makes a TCP connection to port 80 on 18.23.0.23.
5. It then sends a GET /hypertext/WWW/TheProject.html command.
6. The www.w3.org server sends the file TheProject.html.
7. The TCP connection is released.
8. The browser displays all the text in TheProject.html.
9. The browser fetches and displays all images in TheProject.html.

Summary

- In chapter 5 we discussed the TCP/IP protocol suite which includes application protocols such as: Telnet, FTP and SMTP.
- We examined the Telnet protocol and defined its method of operation.
- We discussed how the FTP protocol is used to transfer files between two machines. We compared FTP with TFTP.
- We examined how the DNS system is used to map symbolic names to IP addresses.
- We discussed the basic operation of the SMTP protocol.
- We examined how large networks are managed by SNMP.
- We looked at how the World Wide Web was introduced and some of its terminology.

Chapter 6

Voice over IP

Chapter 6. Voice over IP (VoIP)

- **After completing this chapter the student will understand the basics of transporting voice traffic over an IP network. The chapter deals with:**
  - **VoIP gateways**
  - **VoIP benefits & issues**
  - **Real-time protocol (RTP)**
  - **Reservation resource protocol (RSVP)**

ERICSSON

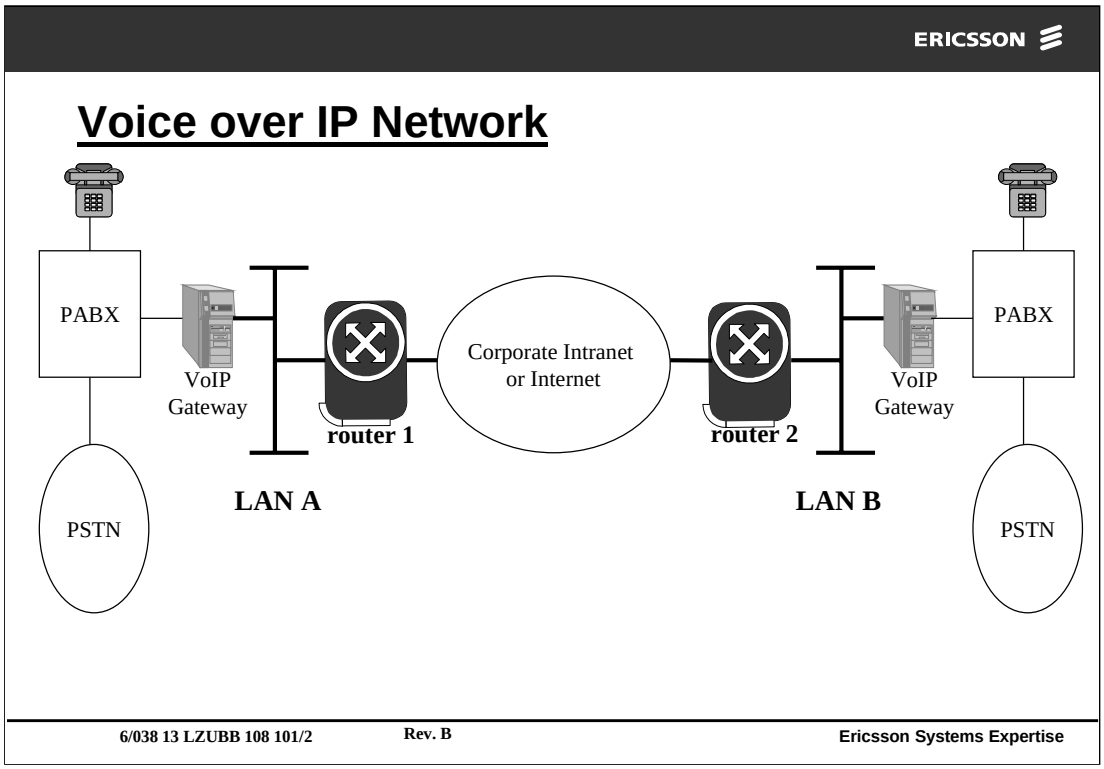
Traditional telephony vs IP telephony

- **Traditional telephony**
  - **Circuit switched**
  - **End-to-end dedicated circuit with fixed bandwidth**
- **IP telephony**
  - **Packet switched**
  - **Shared network with other IP traffic**

Traditional telephony uses circuit switched technology, where an end-to-end circuit is set up between two telephones. A circuit switched connection is established for the duration of every telephone call, with a fixed bandwidth (64 kbit/s) reserved even during silent periods.

IP telephony uses the Internet Protocol to transmit voice as packets over an IP network. In an IP telephony connection, the voice signal is digitised, compressed and converted into IP packets, which are transmitted over the IP network and shared with other IP traffic. An IP packet-based network moves information at a much lower cost by making better use of the network capacity. Not only is a packet-based shared network more efficient than a fixed 64 kbit/s circuit switched connection, but it also compresses the voice signal.

IP telephony can be implemented, at least in principle, on any data network that uses IP, such as the Internet, intranets or LANs. To accomplish this, a device called a “voice over IP gateway” provides the connection between the telephone network and the IP network.



Typically IP telephony is handled by a voice over IP gateway that is placed between a PABX and the IP network.

The gateway provides the physical interface between the telephone network and the IP network. The gateway handles the signalling to and from the telephone network, reception of telephone numbers, conversion between telephone numbers and IP addressing in the IP network, as well as voice processing.

The voice processing includes reception of the voice signal, compression and packetisation, echo cancellation, and silence suppression. The gateway compresses the voice signal for two reasons: to reduce the amount of bandwidth required in order to reduce cost and to reduce the delay impact from the network.

Generally users dial the telephone number of the gateway. The gateway responds with an audio request for the destination telephone number, and a routing table identifies which gateway is located closest to the destination telephone network. The IP address of that gateway is then used to route the telephone call as packets through the IP network.

The gateway also reverses the operation for packets coming in from the IP network and going out to the phone network. Both operations (coming from and going to the phone network) can take place at the same time, allowing a full duplex conversation.

The following describes the typical steps in placing a IP telephony call in a corporate network:

1. The caller unhooks a standard desk telephone, which is supported by a PABX. The PABX is physically connected to the gateway via an access card, for example an E1 card.

2. The caller then dials an access code (for example, five) that tells the PABX to route this call over the PABX trunk connected to the gateway. Next the caller types the branch or extension number (for example 830-1234).

3. The gateway routes call set-up messages over the corporate network to the remote gateway. The gateway sets up the call via the PABX and, if the called party is available, voice bits are encapsulated within the IP payload.

More precisely:

• The access number to the gateway, the destination office number, and the remote extension number trigger a calling-out signal that travels from the telephone through the PBX.

• The calling-out signal goes into an origination gateway interface card.

•The origination gateway undertakes call set-up based on the digits entered. The gateway's telephone database maps the destination office number to the remote gateway's IP address.

• The gateway establishes the availability of an open channel to the remote gateway. If a priority queuing protocol such as resource reservation protocol (RSVP) is available, the gateway can use it to request allocation of bandwidth on the network. Otherwise, a standard best-effort IP service is utilised.

• During the course of a conversation, the voice signal is digitised and compressed into datagrams. The datagrams are encapsulated into IP packets. The packets are transmitted from the gateway's voice interface card through the PC's (Ethernet) network interface card over the LAN medium to the router. The router forwards these packets across the network on a priority/RSVP basis or on a best-effort basis.

• The destination gateway reassembles the original voice packets and delivers them to the destination PABX.

ERICSSON

Types of Gateway

- PC-based gateway
- Chassis-based gateway
- Standalone PC
- Standalone IP telephone

Rev. B

Ericsson Systems Expertise

The voice gateway is a bridge between the PSTN network and the IP network. It takes phone or fax calls from the PSTN and puts them on the IP network in a suitable packet format. It also takes packets corresponding to phone or fax calls from the IP network and reconstructs the original voice stream and sends it to the PSTN network.

A number of different gateway types exist as described in the following list:

• PC-based gateways that use the CPU for voice processing

• PC server-based gateways equipped with specialised Digital Signal Processing (DSP) cards designed for IP telephony

• Chassis-based gateways that use DSP technology

• Standalone PCs with sound cards and IP telephony software

• Standalone IP telephones

These gateways have very different capabilities, both with regard to processing power and scalability. A PC-based gateway using the CPU for all voice processing would not be very scalable compared to chassis-based gateways with DSP boards. An IP PC-based gateway also might not be able to provide the same voice quality as a DSP-based gateway.

ERICSSON

IP telephony scenarios

- Phone-to-phone
- PC-to-phone
- Phone-to-PC
- Fax-to-fax

Rev. B

Ericsson Systems Expertise

IP telephony encompasses a number of different services including phone-to-phone, PC-to-phone, phone-to-PC, PC-to-PC and fax-to-fax, as well as video conferencing and collaboration from the desktop.

In an IP telephony solution, it is possible to have a combination of PC-based telephony applications and phones connected to the PSTN .

In a phone-to-phone scenario the gateway has the functionality needed in order to send and receive voice over an IP network in real time.

In a PC scenario an IP telephony client is needed. The client digitises, compresses and packetises the voice signal and transmits it over the IP network. Standard telephone calls are connected to a voice gateway and IP telephony calls are connected to a telephone or a PC.

IP telephony client software may also allow users with multimedia PCs to have video and audio conferences, share documents and use a whiteboard, enabling a more efficient work environment.

Fax-to-fax IP telephony includes both real-time fax and store-and-forward fax.

In real time the fax is sent directly from the sending machine to the receiving machine. Store-and-forward faxing connects a server to a gateway. This server acts as a destination fax, storing the fax until it is retransmitted to the real destination. Using this method, faxes can be held back when there are high network loads.

ERICSSON

Benefits of IP telephony

- Cost reduction
- Better use of network capacity
- Bandwidth utilisation
- Reduced management and operational costs
- Service integration

Rev. B

Ericsson Systems Expertise

Cost Reduction

The possibility of making cheap calls over the Internet/intranet generates most of the interest in IP telephony today. Internet home users can call abroad for the price of a local call, totally bypassing the long distance telephone network. Companies can do likewise. IP telephony provides company headquarters with an alternative way to contact branch offices and can result in significant cost savings, especially if the company has a corporate intranet that it can reuse for voice traffic.

Better use of Network Capacity

IP telephony makes more efficient use of the existing communications infrastructure. Since IP telephony uses a packet switched network, a number of calls share the same network link, and there is better use of that link and lower transmission costs.

Bandwidth Utilisation

With voice/data integration, voice and fax are converted into data and placed on the IP network for transport to a remote location. With the compression techniques now available it is possible to use high-capacity IP networks for real-time traffic as well as for the traditional, less demanding traffic such as e-mail and file transfers.

Reduced Management and Operational Costs

IP telephony makes it easier to integrate voice communications with

different applications and services on the IP network thereby providing a unified service. Integrating the data and voice networks into one network, reduces management and operational costs.

Service Integration

Service integration makes it possible for one network to support a large number of services. As a result, operational costs can be reduced and new advanced services created.

ERICSSON

Issues with IP Telephony

- Voice quality
- Interoperability
- Security
- Integration with PSTN
- Scalability

Rev. B

Ericsson Systems Expertise

Voice quality

When data is being transferred through an IP network, a slight delay with packets is usually not noticeable. Moreover, retransmission of discarded packets usually compensates for the loss of packets. However when IP packets transport digitised voice the loss or delay of packets results in the disruption of speech intelligibility.

The voice gateway introduces delay because of compression and decompression, and packetisation and depacketisation. The router-induced delay depends on the capacity of the router and the number of router hops from originating voice gateway to terminating voice gateway.

From an end-user point of view, the delay encountered in the communication has to be below a certain threshold (approximately 200ms); otherwise the communication will be considerably less useful.

The Internet Engineering Task Force (IETF) is developing several techniques for reducing delay and jitter (variable delays) in real-time traffic over a large, routed network.

In smaller networks, it is possible to use techniques such as resource reservation protocol (RSVP), or the IP precedence indicator (TOS) in the IP header.

Delays in the voice gateway itself vary greatly between different vendors' products, but, in general, these gateways are constantly improving.

Interoperability

Interoperability relates to products from different vendors as well as different carrier networks. Interoperability issues exist mainly because standardisation is not yet mature in IP telephony. There are no universally implemented or agreed standards for signalling, calling, accounting or billing.

The Voice over IP forum and the ITU are developing standards, such as H.232 and Tiphon (Telecommunications and Internet Protocol Harmonisation Over Networks), to improve interoperability between different vendors' products.

Security

The basic security issues are:

• User and data authentication

• Data privacy (integrity and confidentiality)

• Access control

• Policy management

Network security is related to the routing side of IP. In a public Internet, the packets can traverse through any router and can be intercepted by anyone. Acceptable security can be obtained by encryption (secure sockets layer - SSL) and tunnelling (layer 2 tunelling protocol - L2TP).

Integration with PSTN

The major issue for integration of IP telephony and PSTN is making the PSTN and the IP telephony network appear to be the one network to the end-user and easy to manage by the operator.

Currently the IP telephony network is a separate network that the end-user often must access via a two-stage dial-up, that is, dial the gateway and receive a voice prompt for the destination E.164 number.

The IP telephony network is not yet ubiquitous, that is, it is not possible to call anywhere via IP telephony, and the user will often need to know to which destinations the operator offers IP telephony.

It is not yet possible to route between the public telephone network and the Internet (or intranet) based on class of service, cost of service, and quality of service.

ITU is addressing this issue in the Tiphon standard. Tiphon deals with issues such as network architecture, numbering and supplementary service integration are being specified.

Scalability

Most gateways currently on the market have PC-based hardware with a limited number of ports.

ERICSSON

Voice over IP Forum & Standards

- The Voice over IP (VoIP) Forum operates as a working group in the International Multimedia Teleconferencing Consortium (IMTC)
- The Voice over IP Forum is responsible for:
  - Defining and establishing a set of open, consistent guidelines for the implementation of devices that perform telephony communications over IP data networks
  - Ensuring seamless product interoperability and high quality of service
  - defining a scheme for gateways to know about one another's existence and location, and to know how to address users
- H.323 is a framework of standards that defines how voice, data, and video will be transported over IP networks

The Voice over IP (VoIP) Forum operates as a working group in the International Multimedia Teleconferencing Consortium (IMTC). It hopes to ensure a smooth interoperability of PCs and phones on the Internet.

The responsibilities of the VoIP Forum are divided into two main areas. The first aims to define and establish a set of open, consistent guidelines for the implementation of devices that perform telephony communications over IP protocol data networks. The second aims to ensure seamless product interoperability and high quality of service. The working group also needs to develop standards for addressing. This group has to define a scheme for gateways to know about one another's existence and location, and to know how to address users. Addressing a particular user is especially challenging in today's infrastructure, where IP addresses are constantly changing. To facilitate product interoperability, the VoIP Forum is creating a master guideline and reference model, called the Service Interoperability Implementation Agreement. This agreement encompasses both Internet telephony client software and gateways to the public telephone network.

H.323 is a framework of standards that defines how voice, data, and video are transported over IP networks. The framework includes international standards for audio and video compression and decompression (CODECs). H.323 utilises the Real-Time Transport Protocol (RTP/RTCP) from the IETF.

ERICSSON

Real-time Transport Protocol (RTP)

- Designed to provide end-to-end delivery services for data with real-time characteristics
- Can be carried inside a UDP payload
- Provides data source and payload type identification that is used to determine payload contents
- Provides packet sequencing that is used to determine correct ordering of packets at the receiver
- Provides timing and synchronisation that is used to set timing at the receiver during content playback
- Provides monitoring that is used to facilitate diagnosis or feedback to the sender on the quality of data transmission.

Rev. B

Ericsson Systems Expertise

Real-time Transport Protocol (RTP)

RTP provides end-to-end delivery services for data with real-time characteristics, such as voice over IP. Those services include timestamping, sequence numbering, delivery monitoring and identification of the type of data transported. Applications typically run RTP on top of UDP (User Datagram Protocol) to make use of UDP's multiplexing and checksum services; both protocols contribute parts of the transport protocol functionality.

RTP works as follows. When packets arrive at the destination the sequence number of each packet is examined to determine the correct sequencing of data and also to record the fraction of lost frames. The RTP packet's timestamp is used to determine the interpacket gap. The timestamp value is set by the source when it encodes the data and transmits packets into the network. As packets arrive at the destination, the change in interpacket gap can be examined, and during playback this information can be used to regenerate the contents at the same rate as they were encoded. By utilising the buffer at the receiver, the source can attempt to pace the outgoing traffic independently of the jitter introduced by the packet network.

The IETF has designed the RTP Control Protocol (RTCP), as a companion to RTP, to monitor the quality of service and provide feedback on the quality of the information transmitted so that modifications can be made if required.

ERICSSON

Resource Reservation Protocol (RSVP)

- RSVP is a signalling mechanism used to request a particular QoS
- RSVP uses underlying routing protocols
- Routers implement admission control and policy control
- The sending device uses a packet classifier and a packet scheduler
- Two main issues
  - all routers, between source and destination, must support RSVP
  - charging mechanism for reserved bandwidth

Rev. B

Ericsson Systems Expertise

Resource reservation Protocol (RSVP)

RSVP is a signalling mechanism that VoIP device uses to request a particular Quality of Service (QoS) across a network. RSVP carries the request through the network, visiting each router that the network uses to carry the data stream. At each router, RSVP attempts to make a resource reservation for the data stream.

RSVP is designed to use the robustness of current IP routing algorithms. This protocol does not perform its own routing; instead it uses underlying routing protocols to determine where it should carry reservation requests. As routing changes path to adopt to topology changes, RSVP adapts its reservations to the new paths wherever reservations are in place.

To make a resource reservation at a router, the RSVP daemon communicates with two local decision modules, admission control and policy control. Admission control determines whether the node has sufficient available resources to supply the requested QoS; policy control determines whether the user has administrative permission to make the reservation. If either check fails the RSVP program returns an error notification to the application process that originated the request. If both checks succeed, the RSVP daemon sets parameters in a packet classifier and packet scheduler to obtain the desired QoS. The packet classifier determines the QoS class for each packet, and the scheduler orders packet transmission to achieve the promised QoS for each stream.

Although RSVP provides a mechanism to reserve bandwidth from source to destination through an IP network, it is currently better suited for intranets rather than the Internet. There are two keys issues to be resolved.

First, the ability to establish an RSVP connection through the Internet depends on all routers between source and destination supporting RSVP. This means that anyone linked to an ISP (Internet Service Provider) or NSP (Network Service Provider) that does not support RSVP cannot reserve bandwidth. Whereas on a private intranet an organisation can ensure that all of its own routers are upgraded to support RSVP.

Second, bandwidth is not free. This means ISPs and NSPs can be expected to bill for the use of reserved bandwidth. How this billing will occur, how multiple providers along a path will co-ordinate billing, and the ultimate manner of billing ISP subscribers remain to be determined.

Summary

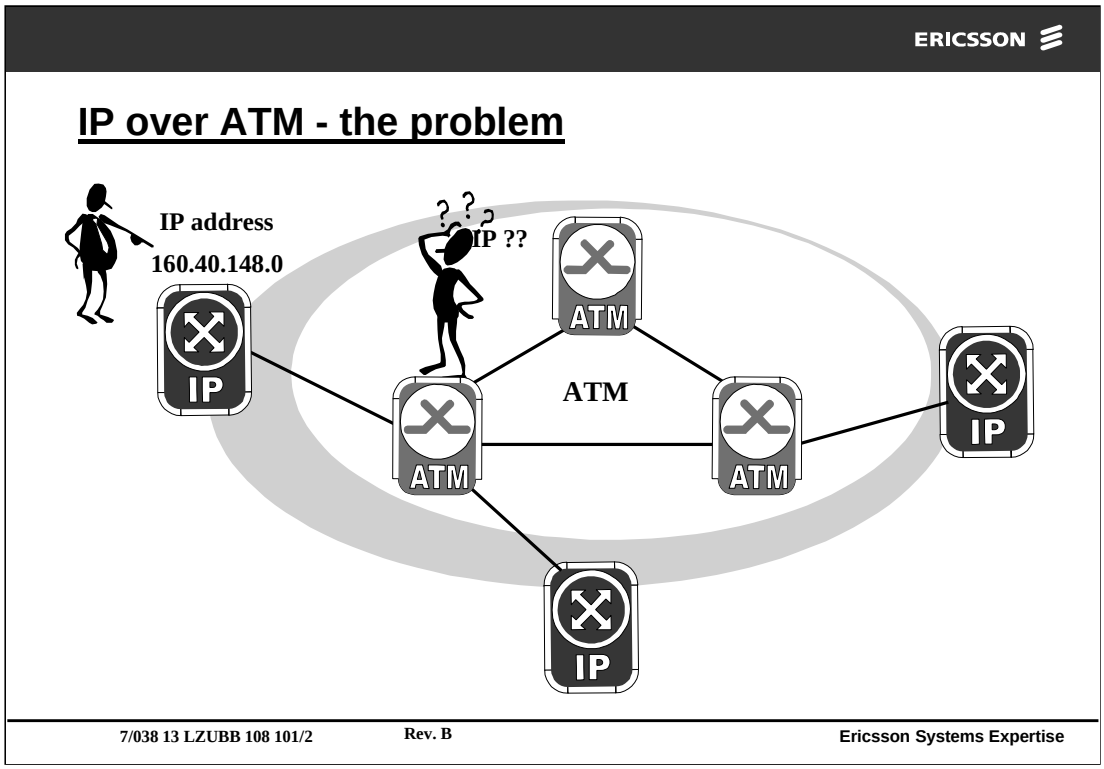
- In chapter 6 we examined the function and operation of VoIP gateways
- We discussed some of the benefits and issues of using VoIP
- We looked at two of the main protocols involved, RTP and RSVP

Chapter 7

IP over ATM

Chapter 7. IP over ATM

- After completing this chapter students will understand:
  - IETF's Classical IP over ATM
  - ATM Forum's LAN Emulation



Let us first consider the differences between ATM networks, and IP networks. ATM is connection-oriented and uses 20-byte ATM destination addresses to set up a connection through a network. IP is connectionless and use 4-byte IP addresses to determine the next hop on the way to the destination. In order to transfer IP data across an ATM network, some method of mapping IP addresses to ATM addresses is needed.

There are a number of fundamentally different ways of achieving this. In this chapter we will discuss two of these, the IETF's Classical IP over ATM (CIOA) and the ATM Forum's LAN Emulation (LANE). Alternative methods which will not be discussed include NHRP (Next Hop Resolution Protocol), MPOA (Multi-Protocol over ATM) and MPLS (Multi Protocol Label Switching).

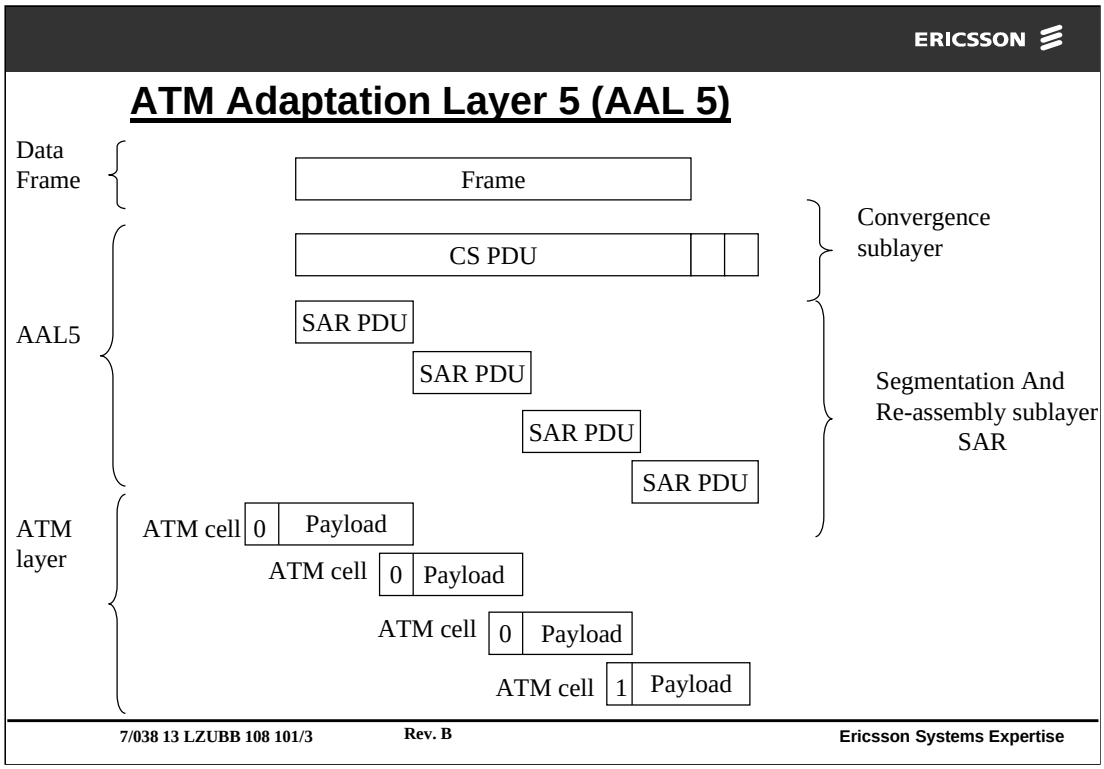
In classical IP, address resolution mechanisms are used to map IP addresses directly to ATM addresses, and the network layer packets are then carried across the ATM network.

As the name suggests, the function of the LANE is to emulate a local area network on top of an ATM network. In other words, the LANE protocol makes an ATM network look and behave like an Ethernet or Token Ring LAN, albeit much faster. With LANE, complete MAC frames, for example Ethernet frames, are carried across the ATM network.

With CIAO and LANE there are two main issues to resolve:

(1) How to encapsulate the large IP packets or Ethernet frames into small fixed size ATM cells.

(2) How to map IP addresses into ATM addresses so that ATM connections can be set up across the ATM network.



The ATM adaptation layer 5 (AAL5) receives packets from upper-level protocols and breaks them into 48-byte segments that form the payload field of an ATM cell. AAL5 is the adaptation layer used in both Classical IP over ATM and LAN emulation.

The diagram above shows how AAL5 prepares a cell for transmission. First, the convergence sublayer of AAL5 appends a variable-length pad and an 8-byte trailer to a frame. The pad is long enough to ensure that the resulting *protocol data unit* (PDU) falls on the 48-byte boundary of the ATM cell. The trailer includes the length of the frame and a 32-bit CRC, computed across the entire PDU, which allows AAL5 at the destination to detect bit errors, and lost cells or cells that are out of sequence.

Next, the segmentation and reassembly segments the CS PDU into 48-byte blocks. Then the ATM layer places each block into the payload field of an ATM cell. For all cells except the last cell, a bit in the payload type (PT) field is set to zero. For the last cell, the bit in the PT field is set to one to indicate that it is the last cell in a series that represent a single frame.

When the cell arrives at its destination, the ATM layer extracts the payload field from the cell; the SAR sublayer reassembles the CS PDU; and the CS uses the CRC and the length field to verify that the frame has been transmitted and reassembled correctly.

ERICSSON

Classical IP over ATM

- All clients register with pre-configured ATMARP server
- Sending client sends IP ARP request directly to ATMARP server
- ATMARP server returns ATM address of destination device
- Sending device uses this ATM address to set-up a direct connection to the destination device

Classical IP over ATM (CIOA) only operates within the confines of an IP subnet; in the terminology of CIOA, an IP subnet is referred to as a Logical IP Subnet (LIS). A LIS consists of a group of IP devices (such as hosts or routers) that connect to a single ATM network and belong to the same IP subnet.

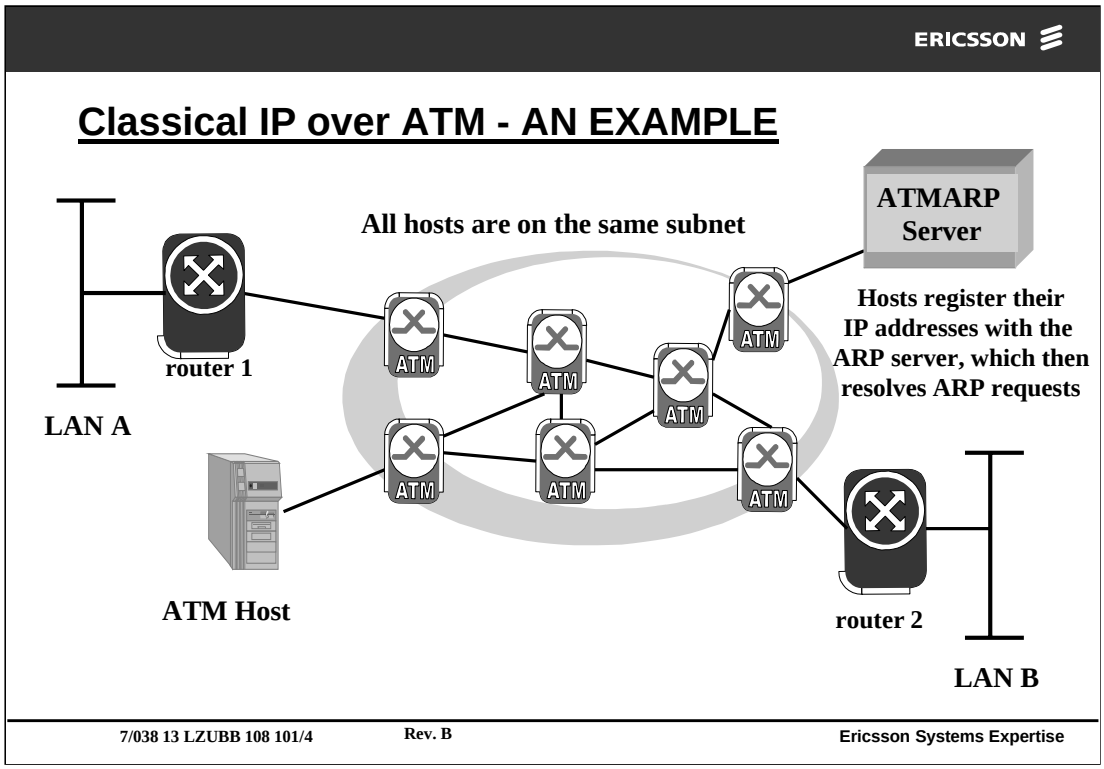
In order to operate IP over ATM, a mechanism must be used to resolve IP addresses to their corresponding ATM addresses. An address resolution table could be configured manually on all devices. Note, this manual method is used on public service provider's networks.

The IETF IP-Over-ATM working group defined the "Classical IP over ATM" protocol to support automatic resolution of IP addresses.

When a client is powered on within the LIS, it first establishes a connection to an ATMARP server, using a pre-configured ATM address. Once the ATMARP server detects a connection from a new LIS client, it transmits an Inverse ARP request to the attaching client and requests the client's IP and ATM addresses, which it stores in its ATMARP table.

Subsequently, any client within the LIS wishing to resolve a destination IP address would send an ATMARP request to the server, which would then respond with an ATMARP reply if an address mapping is found. If not, it returns an ATM_NAK response to indicate the lack of a registered address mapping. The ATMARP server times out its address table for robustness after 15 minutes, unless clients periodically refresh their entry with responses to the server's Inverse ARP queries.

Once an LIS client has obtained the ATM address that corresponds to a particular IP address, it can then use PNNI signalling to set up an ATM connection to the address.



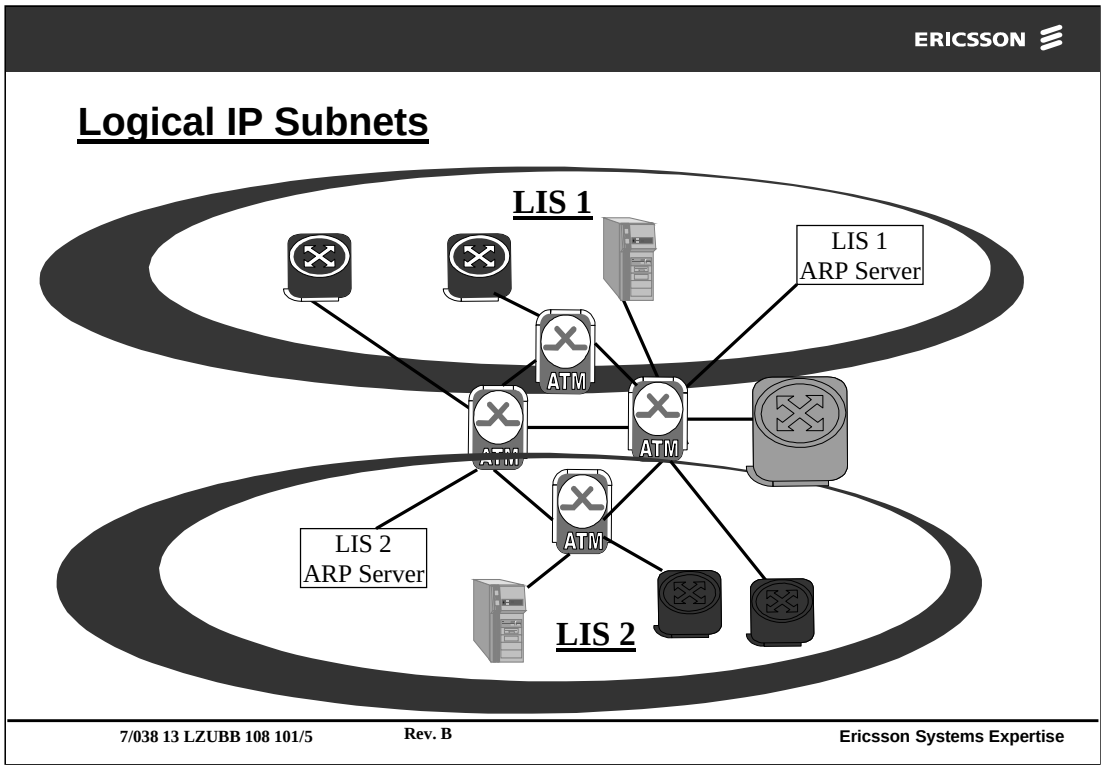
Classical IP over ATM (CIOA)

In the example above two routers are connected together across an ATM network. If router 1 receives a packet via its LAN interface, it first checks its routing table to determine through which port, and to what next-hop router, it should forward the packet. If this check indicates that the packet is to be sent across an ATM interface, the router then needs to consult an address resolution table to determine the ATM address of the destination next-hop router (router 2).

On an Ethernet network, a router, like all other IP devices, would broadcast an ARP request to find out the Ethernet address of the destination, and cache the results in its address resolution table. However an ATM network does not support broadcasts, because it is a connection-oriented protocol.

To resolve the addresses of clients within the subnet, each subnet supports a single ATMARP server, which is responsible for resolving all ARP requests. All clients within the subnet are configured with the unique ATM address of the ATMARP server.

Router 1 sends an ATMARP request to the server, which then supplies the ATM address of router 2. Router 1 can then set up an ATM connection directly to router 2’s ATM interface.



Logical IP Subnet (LIS)

Multiple subnets may be simulated within a single ATM network. Each LIS requires its own ATMARP server. Communication between LISs must be passed through a router which is a member of both (or more) LISs.

Traffic must go through this router even if there are more direct paths within the ATM network. This means that the “classical” model of routing is preserved, that is, traffic going from one subnet to another has to go through a router.

This requirement, however, is inefficient in "non-broadcast multi-access" (NBMA) networks, including ATM, since the traffic must be processed through routers even if a direct ATM connection could be set up between end devices.

When a client on one LIS tries to send an IP packet to a client on another LIS, the IP software on the sending client notes that the destination address is not on the same LIS and sends the packet to the default gateway (router) on the LIS. This triggers an ATMARP request to discover the router’s ATM address, followed by a call set up from the sending client to the router. Once the router receives IP packets from the sending client, the router issues an ATMARP request to find the ATM address of the receiving client. It then sets up the call to that client.

What this means is that when clients communicate through a router, there are two virtual circuits across the ATM network: one from the sending client to the ATM interface on the router, and another from the router to the receiving client. The sending client segments the IP packets into cells for transmission across the ATM network which are then reassembled in the router. The router then makes a forwarding decision based on the IP packet header information, and resegments the packet for sending over another virtual circuit to the receiving station.

This is inefficient and resulted in the development of the Next Hop Resolution Protocol (NHRP) which allows ATM connections to be set up directly between two ATM devices in different logical IP subnets. For more details on how this works consult RFC 2332 - “NBMA Next Hop Resolution Protocol (NHRP)”

ERICSSON

LAN Emulation (LANE)

- **LANs V ATM**
  - **Connectionless V connection-oriented**
  - **Broadcast media v point-to-point**
- **LANE simulates connectionless data transfer & broadcasting**
- **LANE maps MAC addresses to ATM addresses**
- **LANE allows existing LAN applications to run over an ATM network without any modification**

Let us first consider the differences between ATM networks, and LAN networks, for example, Ethernet. Note, approximately 95% of LANs world-wide use Ethernet technology.

•LANs offer connectionless service, so applications can communicate without any prior set-up. In contrast, ATM networks are connection-oriented that is, before two end-systems can communicate, they must establish a connection between them, either via signalling (SVC) or via provisioning (PVC).

•LANs are broadcast media. ATM networks are point-to-point, so broadcasts/multicasts have to be accomplished by making explicit copies in the network.

From the above differences, we conclude that LAN Emulation (LANE) must provide for connectionless data transfer, simple broadcast/multicast, address mapping from IEEE 802 (MAC address) addresses to ATM addresses, and the ability to interconnect with existing LANs.

LAN Emulation (LANE), as defined by the ATM Forum, is a service provided to allow existing LANs, such as Ethernet and Token Ring, to run over an ATM network without any modifications.

IP applications rely on Address Resolution Protocol (ARP) in order to determine the MAC address of the destination host. ATM does not support this ARP service. Therefore, LAN Emulation must emulate this service on a connection-oriented network.

ERICSSON

LAN Emulation Components

- LAN Emulation Client (LEC)
  - ATM device that translates between ATM and LAN interface
- LAN Emulation Configuration Server (LECS)
  - Assigns new clients to their designated Emulated LAN
- LAN Emulation Server (LES)
  - Allows clients to register and resolves MAC to ATM addresses
- Broadcast and Unknown Server (BUS)
  - Forwards broadcast traffic to all registered clients

LAN Emulation Client (LEC)

Any ATM device that wishes to use the services of Ethernet (IEEE 802.3) or Token Ring (IEEE 802.5) must have a LAN Emulation Client (LEC) for that type of LAN. The LEC communicates with other LECs to transfer data between communicating applications. To identify the destination, a LEC uses the LAN emulation service components, described below.

LAN Emulation Configuration Server (LECS)

The LECS's role is to assign newly initialised clients (LECs) to the correct Emulated LANs (ELANs), since there may be multiple ELANs on an ATM network. Typically there will be one LECS per administrative domain that will serve all the ELANs, whether Ethernet or Token Ring.

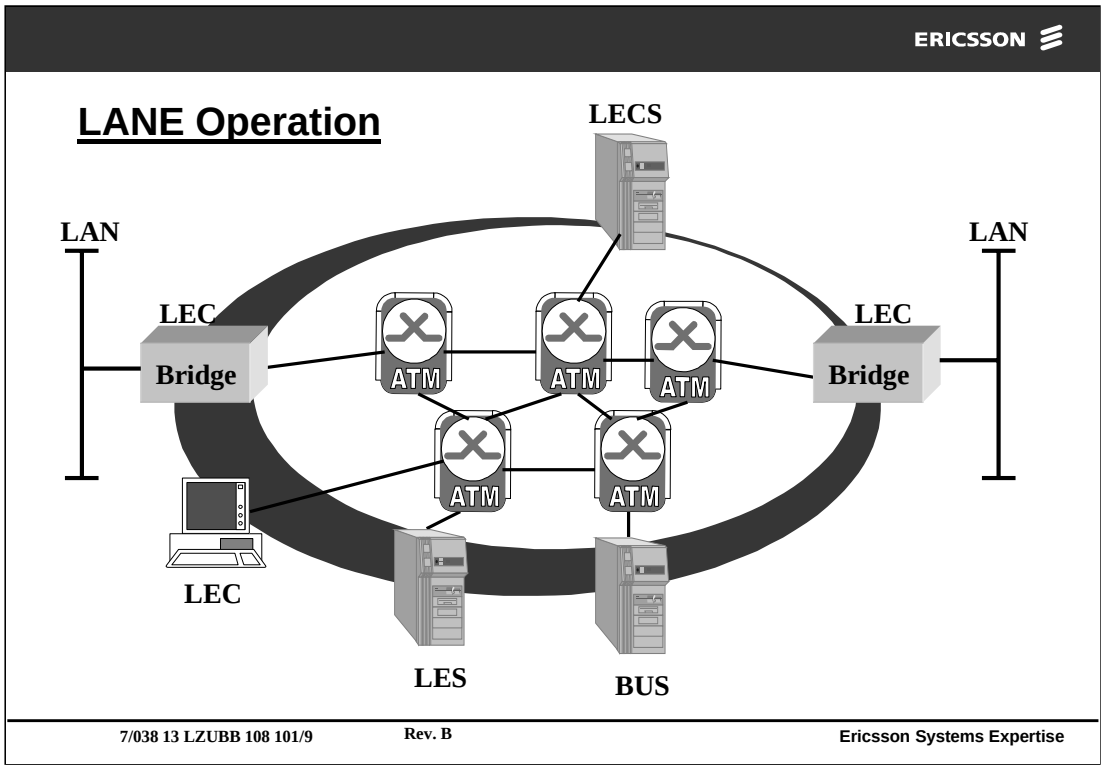
LAN Emulation Server (LES)

There is always one LES for each ELAN. The LES provides control and co-ordination functions within the ELAN. It controls the joining of LECs to its ELAN, registers client's MAC and ATM addresses, and provides address translation services.

Broadcast and Unknown Server (BUS)

The BUS forwards data from one LEC to all other LECs. It handles broadcast traffic for a given LAN.

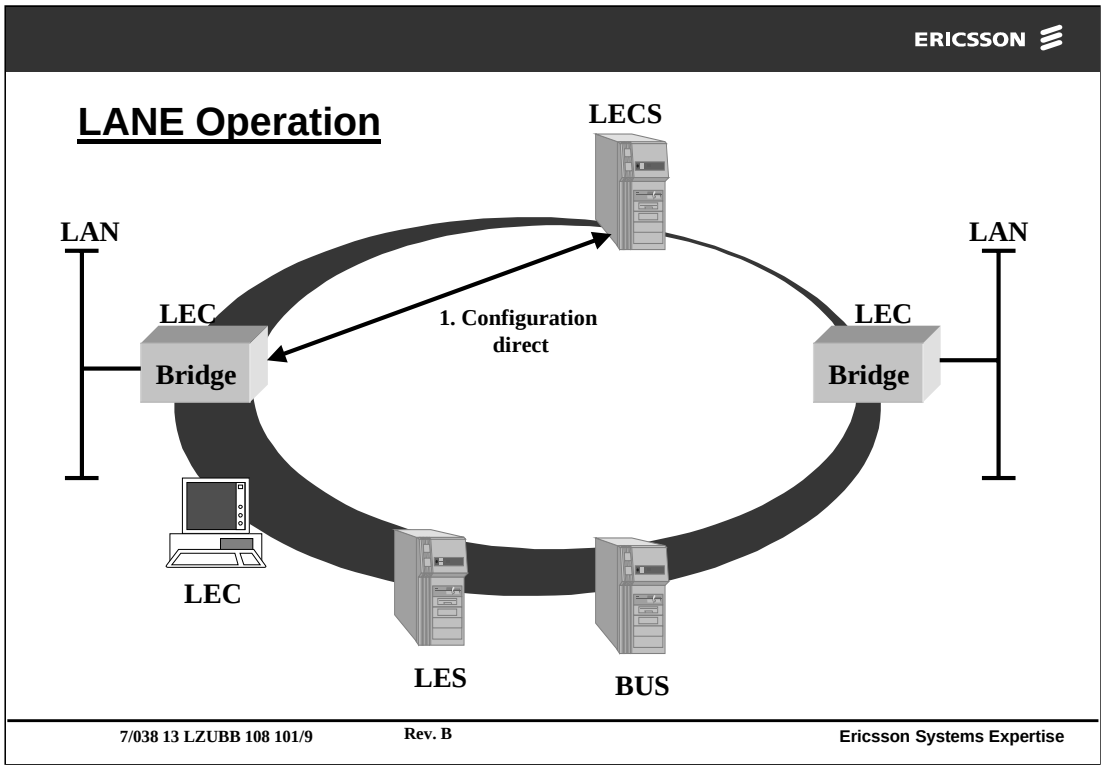
Each LEC is assigned to a specific BUS that handles its broadcast and multicast traffic. There is always one BUS for each ELAN.



LANE OPERATION

The following is a summary of LANE operation; the subsequent 5 pages describe this process in more detail:

The LANE protocol includes facilities for initialisation, registration, address resolution, and data transfer. The initialisation phase includes obtaining the ATM address of the LES and joining or leaving a given ELAN. Registration is the next step and consists of the LEC informing the LES of the MAC addresses of the non-ATM devices that the LEC represents. An alternative to sending an explicit list of addresses is for the LEC to indicate its willingness to act as a proxy, in which case it will participate in the LAN Emulation (LE)_ ARP protocol to resolve the addresses for non-LES-registered LAN-connected hosts. Next, a source LEC must obtain the ATM address of the distant client through which a given MAC address is reachable. Once this is accomplished, the LEC can set up an SVC (switched virtual circuit) to the destination using PNNI, encapsulate the data, and forward it on its way.



Initialisation and Configuration

The LEC must first find the location of the LECS by using one of the following:

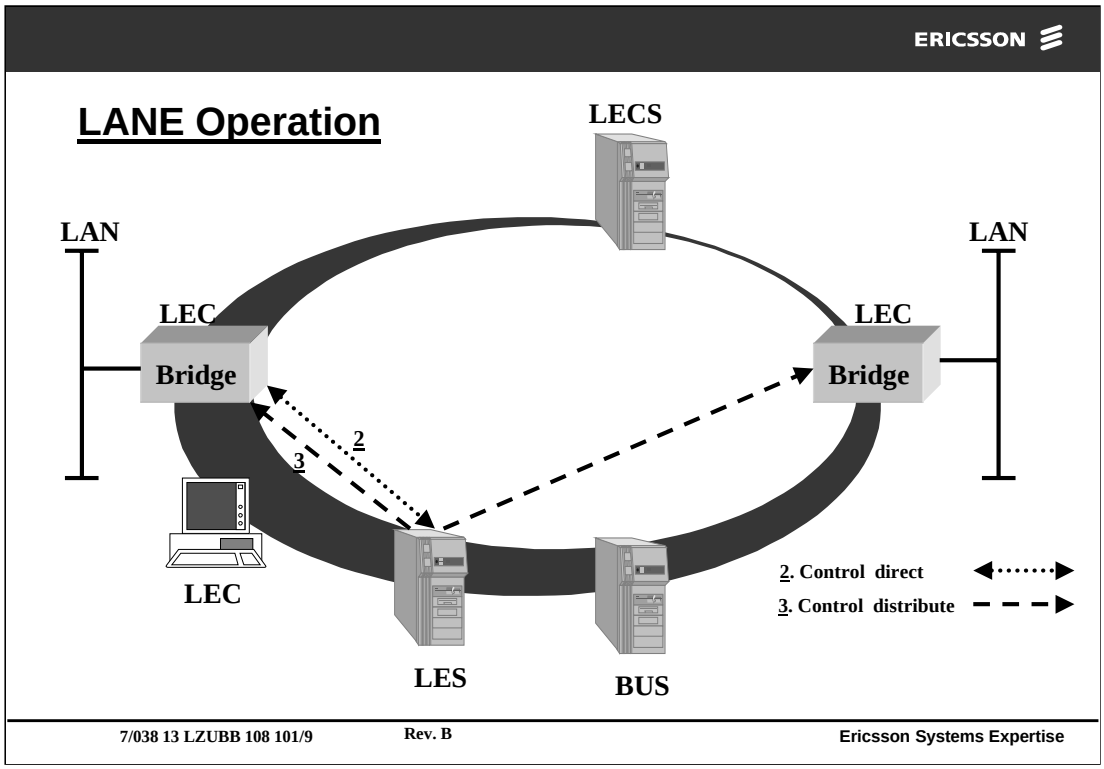
• a defined ILMI (Integrated Local Management Interface) procedure to determine the LECS's address;

• its well-known ATM address (47:00:79:00:00:00:00:00:00:00:00:00:00:A0:3E:00:00:01:00);

• a well-known permanent connection to the LECS (VPI=0, VCI=17).

After finding the location of the LECS, the LEC establishes a configuration-direct VCC (virtual channel connection) to the LECS. Once connected a configuration protocol is used by the LECS to inform the LEC of the information it requires to connect into its target ELAN. This includes the ATM address of the LES, the type of LAN being emulated, maximum packet size on the ELAN, and the ELAN name. The LECS is generally configured by network management with this information, which effectively indicates which emulated LAN the LEC belongs to.

Once the LEC obtains the LES address, it may optionally clear the configuration-direct VCC to the LECS.



Joining and Registration

The LEC sets up a control direct VCC to the LES. Once this is done, the LES assigns to the LEC a unique LEC identifier (LECID). The LEC then registers its own MAC and ATM addresses with the LES. It may optionally also register any other MAC addresses for which it is proxying, for example, learned addresses in the case of a bridge. Alternatively the LEC may register with the LES as a "proxy" node, indicating that it may proxy for other addresses and needs to obtain LE_ARPs.

The LES then sets up, back to the LEC, a control distribute VCC. The control direct and distribute VCCs can then be used by the LEC for the LAN Emulation ARP (LE_ARP) procedure for requesting the ATM address that corresponds to a particular MAC address.

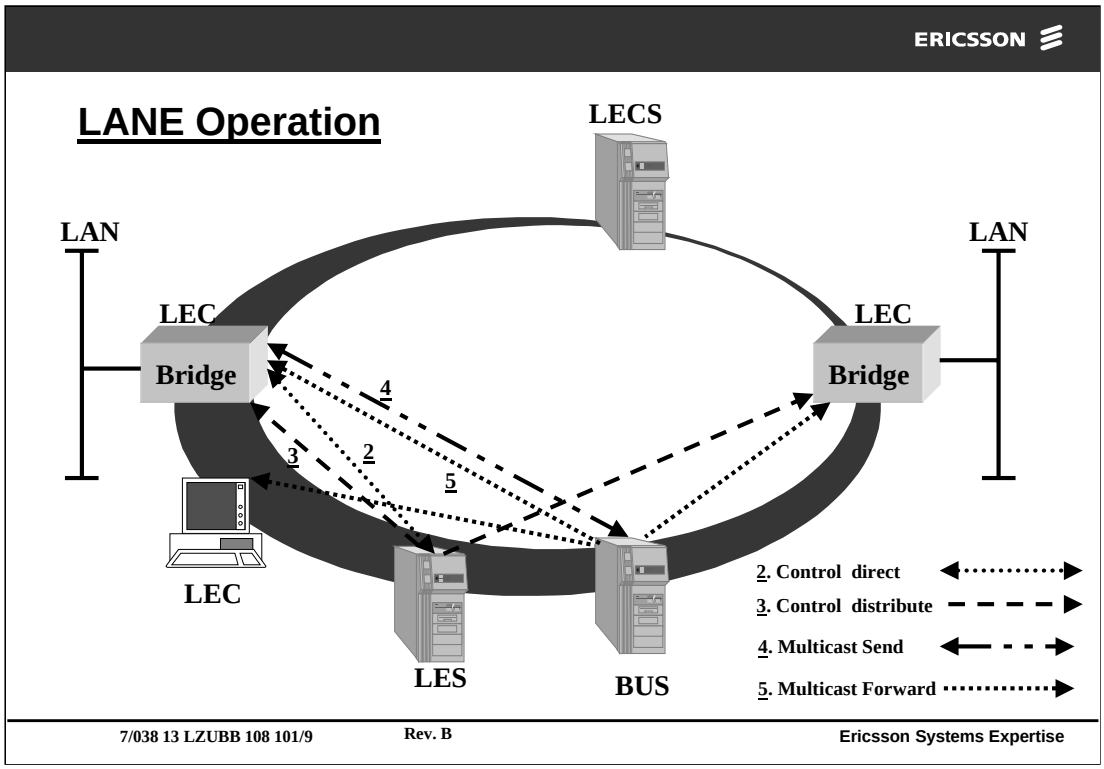
LAN Emulation ARP Procedure

To resolve MAC to ATM addresses, the LEC formulates an LE_ARP and sends it to the LES. If the LES recognises this mapping (because some LEC registered the relevant MAC address) it may choose to reply directly on the control direct VCC. If not, it forwards the request on the control distribute VCC to solicit a response from an LEC that knows the requested MAC address.

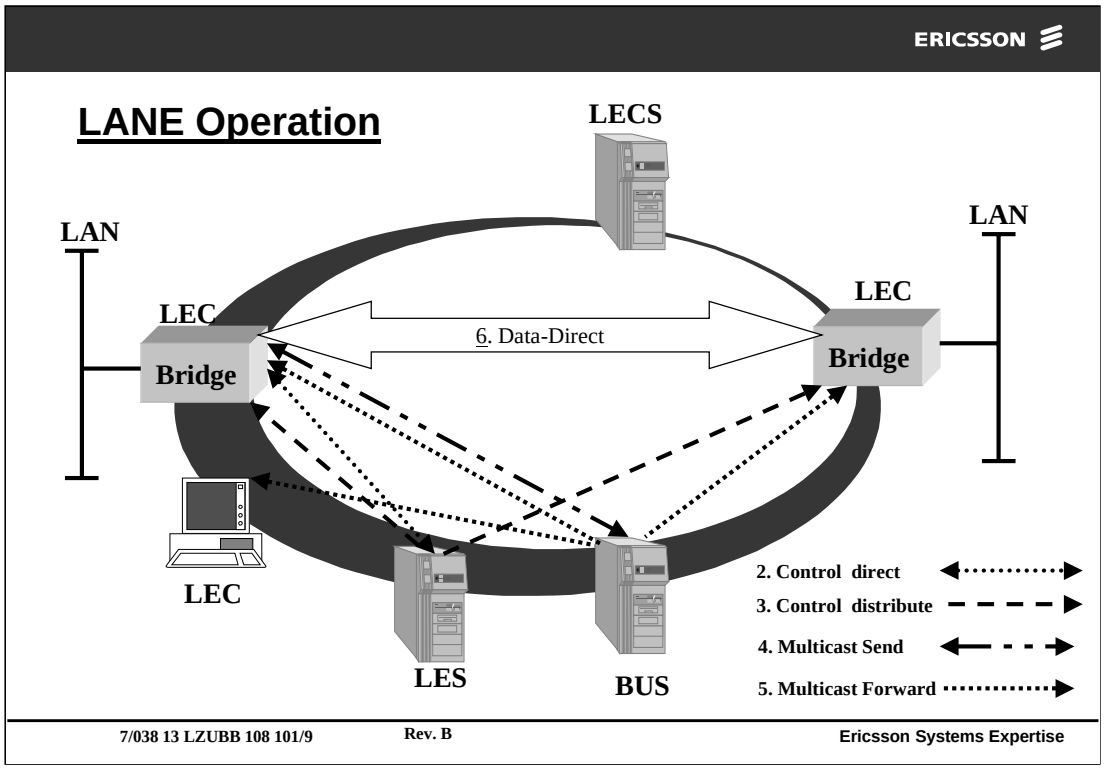
The typical reason why the LES would not know a mapping is that the address is "behind" a MAC bridge, and the bridge may not have registered the address. An ATM NIC, on the other hand, would presumably only support one or a small number of MAC addresses, all of which could easily be registered. Typically, any MAC address not known to the LES would be found only in an LEC within a bridge, and not within an NIC, and only the LECs within such “proxy”nodes need necessarily receive re-directed LE_ARPs.

The LES then has the option of setting up the control distribute VCCs so that LE_ARPs are only sent to such proxy LECs; for example, through two point-to-multipoint connections, the first connecting the LES to all of the proxy nodes, and the second connecting the LES to all of the non-proxy nodes. However setting up two control distribute VCCs is not a requirement, and the LES may choose to simply distribute the LE_ARP to all LECs.

In any case, if a LEC can respond to an LE_ARP, because it is proxying for that address, it responds to the LES on the control direct VCC. The LES then forwards this response back either to the requesting LEC only, or, optionally, on the control distribute VCC to all LECs, so that all LECs can learn and cache the particular address mapping (and hence perhaps save future LE_ARPs).



To complete initialisation, a LEC uses the LE_ARP mechanism to determine the ATM address of the BUS. It does this by sending an LE_ARP for the MAC broadcast address (FF:FF:FF:FF:FF:FF) to the LES, which responds with the BUS's ATM address. The LEC then sets up a multicast send VCC to the BUS. The BUS, in turn, sets up a multicast forward VCC back to the LEC, typically by adding the LEC as a leaf to a point-to- multipoint connection. The LEC is now ready for data transfer.



Data Transfer

During data transfer, a LEC receives either a network layer packet to transmit from a higher layer protocol (in the case of NIC) or a MAC packet to forward across a LAN port (in the case of a LAN switch or bridge). In the first instance, the source LEC will not have the ATM address of the destination LEC through which the particular destination MAC address can be reached. In this case, the LEC first formulates and sends to the LES an LE_ARP request.

While waiting for a response from this LE_ARP, the LEC also forwards the packet to the BUS, using a defined encapsulation. The BUS will, in turn, flood the packet to all LECs. This must be done because, in the case of a passive device behind a LAN switch, no LEC may know where the MAC address is located. Additionally, resolving an LE_ARP may take some time and many network protocols are intolerant of either loss (if the LEC chooses to discard the packet while awaiting the LE_ARP response) or latency (if the LEC chooses to buffer the packet). The flooding of packets by the BUS provides the analogue of the flooding procedure used by spanning tree bridges for unknown destination packets, hence its name.

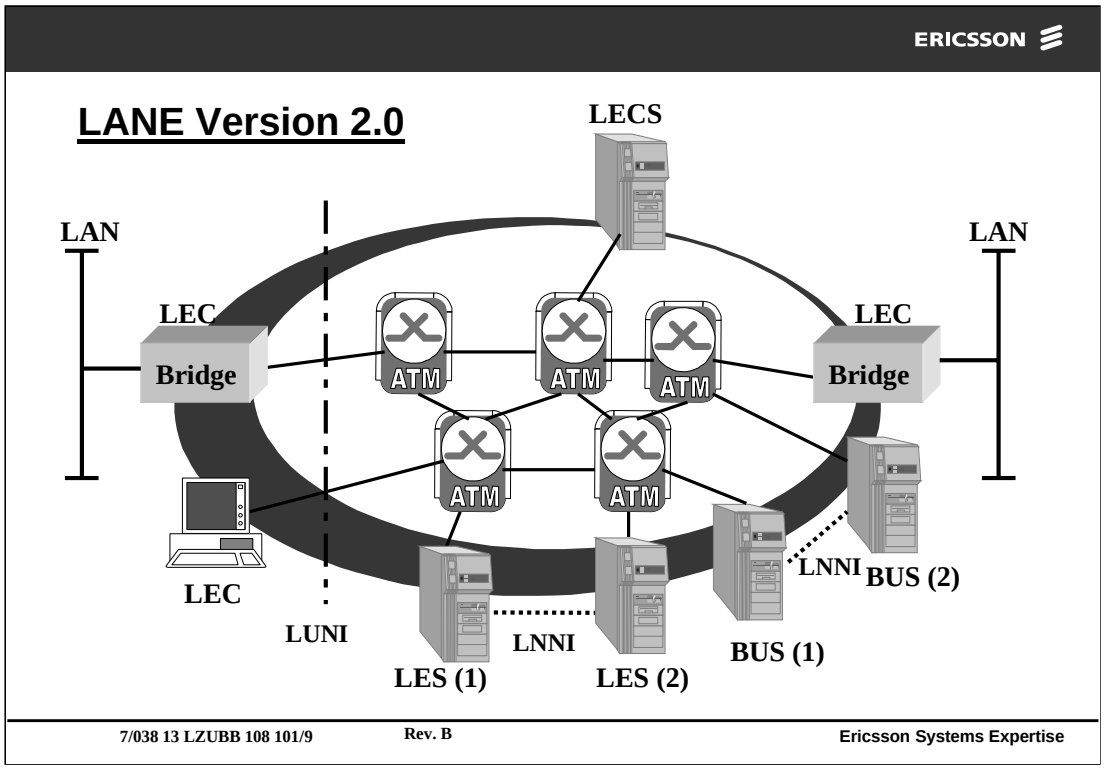
If an LE_ARP response is received, the LEC then sets up a data-direct VCC to the destination node, and uses this for data transfer rather than the BUS path. Before it can do this, however, the LEC may need to use the LANE "flush" procedure to ensure that all packets previously sent to the BUS were delivered to the destination prior to the use of the data direct VCC. In this procedure, a control cell is sent down the first transmission path, following the last packet; not until the receipt of this flush cell is acknowledged by the destination is the second path used to send packets. This procedure is the guaranteed way to meet current LAN standards that require LAN bridges to strictly preserve frame ordering.

If a data direct connection already exists to the LEC (in the same ELAN) through which a particular MAC address is reachable, the source LEC may optionally choose to re-use this same data direct connection, so as to conserve connection resources and save on the connection set-up latency.

If a response is not received to an LE_ARP, the LEC will continue to send packets to the BUS, but will regularly re-send LE_ARPs until a response is received. Typically once a packet is flooded through the BUS, and the destination responds to the source, some LEC will learn the location of the destination, and then respond to a subsequent LE_ARP.

A LEC will locally cache any mappings of MAC address to ATM addresses it learns through an LE_ARP. If and when the LEC receives for transmission another packet to that same MAC address, it will then consult that local cache table and use the cached mapping, rather than sending out another LE_ARP. Such cached entries are normally timed out over a configurable time period (typically 5 minutes). Similarly, data direct connections will be cleared if the connection remains inactive over a configurable period (typically 20 minutes).

The BUS is also used by LECs for broadcast and multicast packets. Such packets are forwarded to the BUS, which then redirects them to all LECs. This implies that the source LEC may receive a copy of its own broadcast or multicast packet. Since some LAN protocols cannot tolerate such a condition, the LANE packet encapsulation requires that all MAC packets be prefixed with the LECID. LECs can then use this field to filter on all frames that are received from the BUS to ensure that it never receives its own frames.



LANE Version 2

The interface between the LEC and the LE Service is known as the LAN Emulation User to Network Interface (LUNI). With LANE version 1, each LEC is in contact with one LES and one BUS, with a single LECS providing configuration services for the multi-ELAN domain.

In LANE version 2.0 a server-to-server interface known as the LAN Emulation Network-Network Interface (LNNI) allows inter-LES and inter-BUS communication. This allows an LEC to contact multiple LECs and BUSs for redundancy or load sharing.

LANE version 2.0 supports Multiprotocol Over ATM (MPOA). MPOA allows LANE devices to operate across subnets with ATM vitual connections. For more details on MPOA consult the ATM Forum.

Summary

- We examined the differences between IP and ATM
- We looked at encapsulation and address resolution
- We discussed in Classical IP over ATM and LAN Emulation

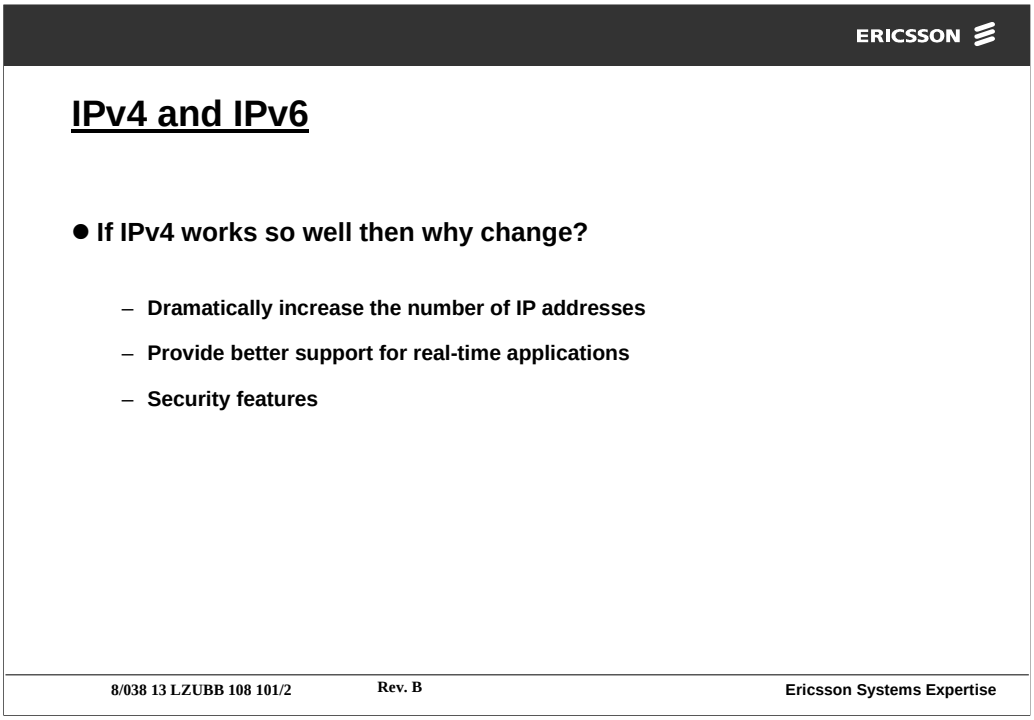
Chapter 8

IP Version 6

ERICSSON

Chapter 8. IP Version 6

- **After completing this chapter students will be able to describe:**
  - **The differences between IPv4 and IPv6**
  - **The packet format**
  - **The base header format and extension header format**
  - **IPv6 Addressing**
  - **IPv6 Colon Hexadecimal Notation**

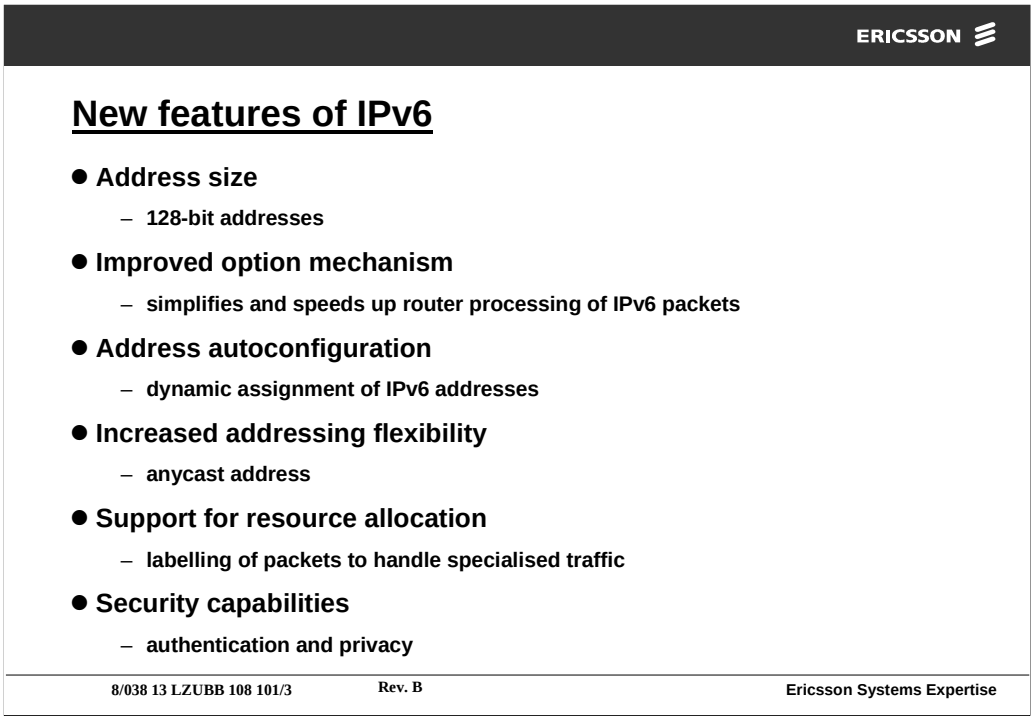


If IPv4 works so well then why change?

The main cause for change was due to the limited address space. When IP was defined only a few computer networks existed. The designers decided to use 32-bit addresses which would allow them to include over a million networks. However, the global internet is growing exponentially, with the size more than doubling annually. At this current rate, all prefixes will soon be assigned and no further growth will be possible.

Secondary reasons for change are new Internet applications. For example, applications that deliver audio and video need to deliver data at regular intervals. To keep such information flowing through the Internet without disruption, IP must avoid changing routes frequently. These new requirements led to the development of IPv6.

The security implemented in IPv6 guarantees that that a packet is actually coming from the host indicated in its source address, unlike in IPv4 where the packet could be coming from a host other than that indicated in the source -this is known as “spoofing”.



The new features in IPv6 can be grouped into the following categories:

•**Address size:** IPv6 uses 128-bit addresses instead of 32-bit addresses of IPv4. This is an increase of address space by a factor of 2^{96}. The address space provided by IPv6 is large enough to accommodate continued growth of the Internet for many decades. There are enough addresses supported by IPv6 to provide an order of 6 * 10^{23} unique addresses per square meter of the surface of the earth.

•**Improved options mechanism:** IPv6 options are placed in separate optional headers that are located between the IPv6 header and the transport layer header. Most of these optional headers are not examined or processed by any router on the packet's path; which simplifies and speeds up router processing of IPv6 packets compared to IPv4 packets.

•**Address autoconfiguration:** this capability provides for dynamic assignment of IPv6 addresses via stateful or stateless address autoconfiguration. DHCP is termed a stateful address configuration tool because it maintains static tables that determine which addresses are assigned to a new or moved stations. A version of DHCP has been developed for IPv6. IPv6 also supports a stateless address autoconfiguration service that does not require a manually configured server. Stateless autoconfiguration makes it possible for devices to configure their own addresses with the help of a local IPv6 router. Typically the device combines its 48-bit MAC address with a network prefix it learns from a neighbouring router.

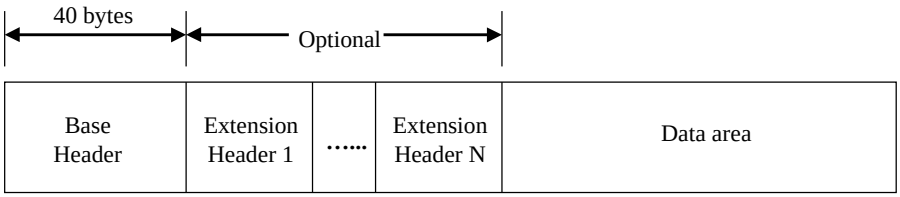
•**Increased addressing flexibility:** IPv6 includes the concept of an anycast address, for which a packet is delivered to just one of a set of nodes. The scalability of multicast routing is improved by adding a scope field to multicast addresses.

•**Support for resource allocation:** instead of the type of service field in IPv4, IPv6 enables the labelling of packets belonging to a particular traffic flow for which the sender requests special handling. This aids in the support of specialised traffic, such as real-time video.

•**Security capabilities:** IPv6 includes features that support authentication and privacy.

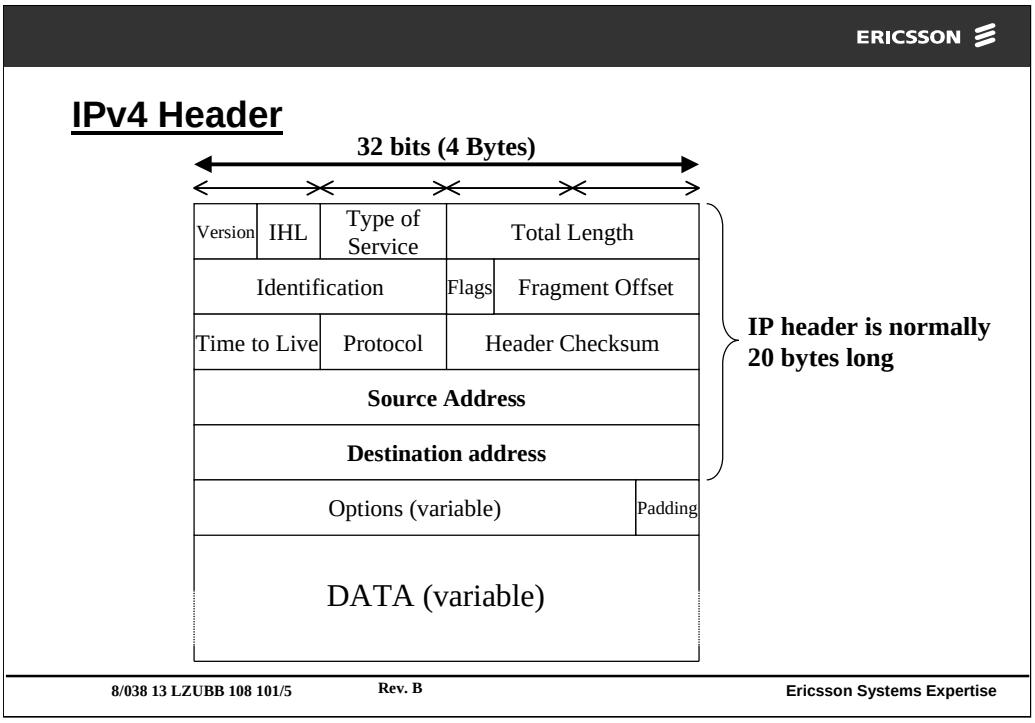
ERICSSON

The IPv6 Packet Format



8/038 13 LZUBB 108 101/4 Rev. B Ericsson Systems Expertise

The IPv6 datagram begins with a base header, which is followed by zero or more extension headers, followed by data. The only header required is that of the IPv6 header. This is of fixed size with a length of 40 octets compared to 20 octets for the mandatory portion of the IPv4 header.



•The IPv4 header contains 14 fields (including options and padding) whereas IPv6 only requires 8 fields.

•Both headers carry version numbers and source/destination addresses.

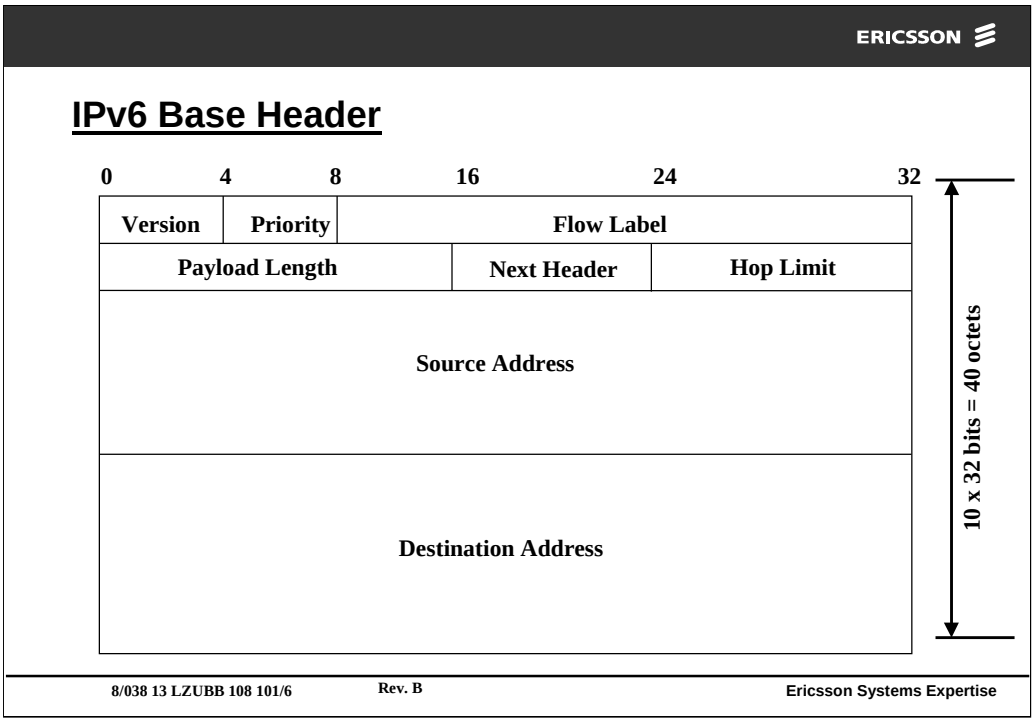
•In IPv6 the following IPv4 fields are not included in the base header: Internet header length, type-of-service, identification, flags, fragment offset and header checksum.

•The Internet header length is no longer required due to the fixed header length of all IPv6 packets.

•The functionality of the IPv4 type-of-service field has been transferred to the two new IPv6 fields: flow label and priority. The IPv4 fragmentation fields have been made optional headers in IPv6. The IPv4 checksum fields have been abandoned in IPv6, due to the prevalence of error checking at other levels of the protocol stack.

•The total length of the IPv4 packet has been retained in the guise of the IPv6 payload length field. But this field does not include the length of the IPv6 header, which is always assumed to be 40 bytes.

•The Time-To-Live(TTL) field of IPv4 has been changes to the IPv6 hop limit field. Although the names are different, both fields are used by routers to decrement a maximum hop value by 1 for each hop of the end-to-end route.



The IPv6 header has a fixed length of 40 octets, consisting of eight fields:

•**Version (4 bits):** IP version number; the value is 6.

•**Priority (4 bits):** Priority value of each packet specifies the traffic class. Values between 0 and 7 are defined for congestion controlled traffic (data) and between 8 and 15 for non-congestion controlled traffic (video and audio).

•**Flow Label (24 bits):** used by applications that require a performance guarantee to specify the path. The IPv6 standard defines a flow as a sequence of packets sent from a particular source to a particular destination. A flow is uniquely identified by the combination of source address and a 24-bit flow label. Thus all packets that are to be part of the same flow are assigned the same flow label by the source.

•**Payload Length (16 bits):** specifies the size of the data being carried.

•**Next Header (8 bits):** identifies the type of header immediately following the IPv6 header, for example, a TCP/UDP header or a IPv6 optional header.

•**Hop Limit (8 bits):** the remaining number of hops for this packet. The hop limit is set to a desired maximum value by the source and decremented by 1 by each node that forwards this packet. The packet is discarded if the hop limit is decremented to zero.

Although the IPv6 Header is longer than that of the IPv4 header, it contains fewer fields. Thus routers have less processing to do per header, which should speed up routing.

•**Source Address (128 bits):** the address of the sender of the packet.

•**Destination Address (128 bits):** address of the intended recipient of packet.

ERICSSON

IPv6 Extension Header

Extension header	Description
Hop-by-hop options	Miscellaneous information for routers
Destination options -1	Information for 1st destination
Routing	Full or partial route to follow
Fragmentation	Management of datagram fragments
Authentication	Verification of the sender's identity
Encrypted security payload	Information about the encrypted contents
Destination options -2	Additional information for the final destination only

There are seven kinds of extension header:

•**Hop-by-hop Options Header**: defines special options that require hop-by-hop processing

•**Destination Option Header -1:** contains optional information to be examined by the first destination listed in the IPv6 address field. This header can also be read by a subsequent destination listed in the source routing header address fields.

•**Routing Header**: allows a source node to specify a list of IP addresses that dictate what path a packet will traverse.

•**Fragment Header**: contains fragmentation and reassemble information.

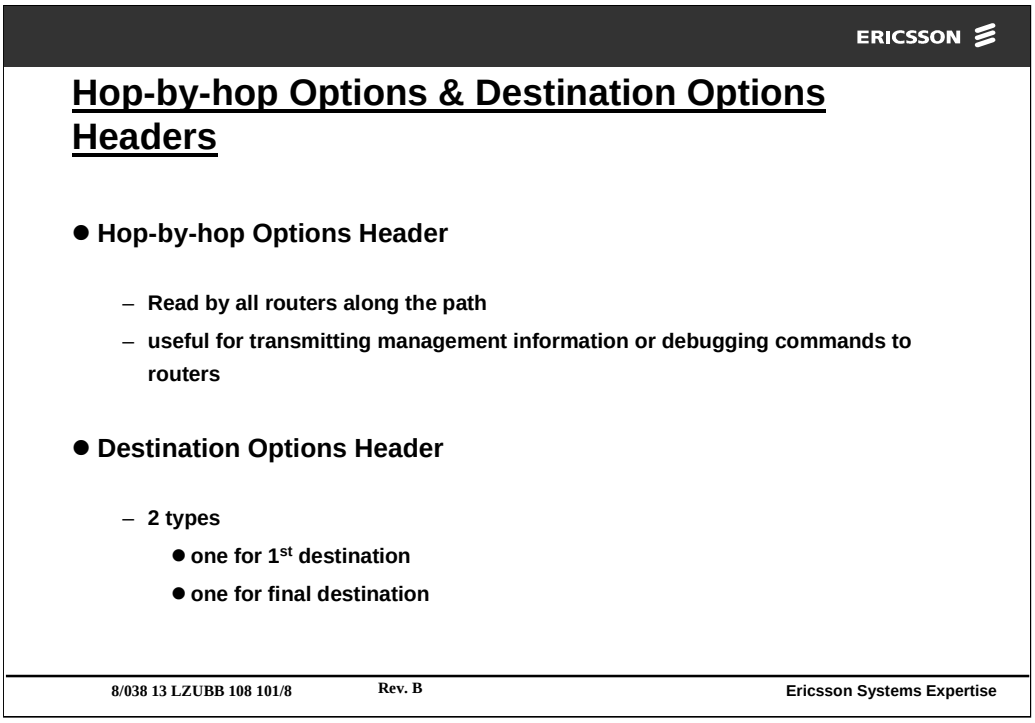
•**Authentication Header**: provides packet integrity and authentication.

•**Encapsulated Security Payload Header**: Provides privacy.

•**Destination Options Header -2**: contains optional information to be examined only by the final destination node.

These headers are supplied to provide extra information, but are encoded in an efficient way. Processing speed is increased because each header is a fixed length. Each one of the seven extension headers is optional and if more than one is present, they must appear directly after the fixed header and in the preferred order.

The IPv6 header and extension header contain a **Next Header Field.** This field identifies the type of the header immediately following. If the next header is an extension, then this field contains the type identifier of that header; otherwise, this field contains the protocol identifier of the upper layer protocol using IPv6.



Hop-by-hop Options Header

The hop-by-hop option header carries optional information that, if present, must be examined by every router along the path. This header consists of the following:

•**Next Header (8 bits):** identifies the type of header immediately following this header.

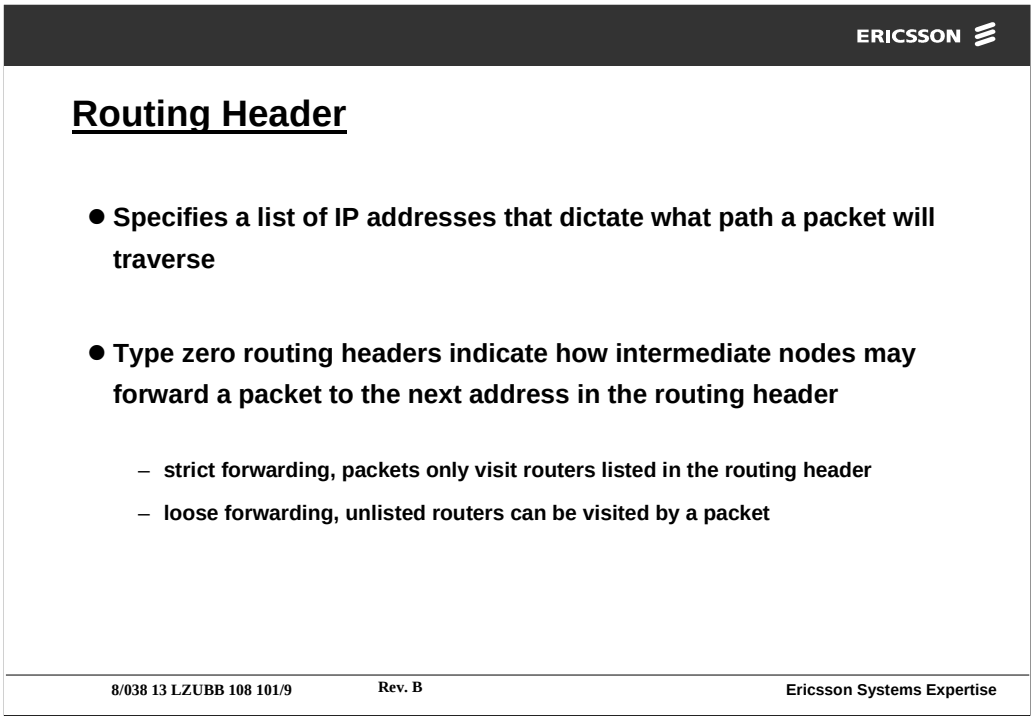
•**Header Extension Length (8 bits):** Length of this header in 64-bit units, not including the first 64 bits.

•**Options:** a variable length field consisting of one or more option definitions. Each definition is in the form of three subfields: option type (8bits), which identifies the option; length (8 bits), which specifies the length of the option data field in octets; and option data, which is a variable-length specification of the option.

Destination Options Headers

There are two variations of this header, each with a different position in the packet. The first variation is for carrying information to the first destination listed in the IPv6 address field. This header can also be read by a subsequent destination listed in the source routing header address fields. The second variation is used for optional information that is only to be read by the final destination.

For efficiency, the first variation is typically located towards the front of the header chain, directly after the hop-by-hop header (if any). The second variation is relegated to a position at the end of the extension header chain, which is typically the last IPv6 optional header before transport and payload.

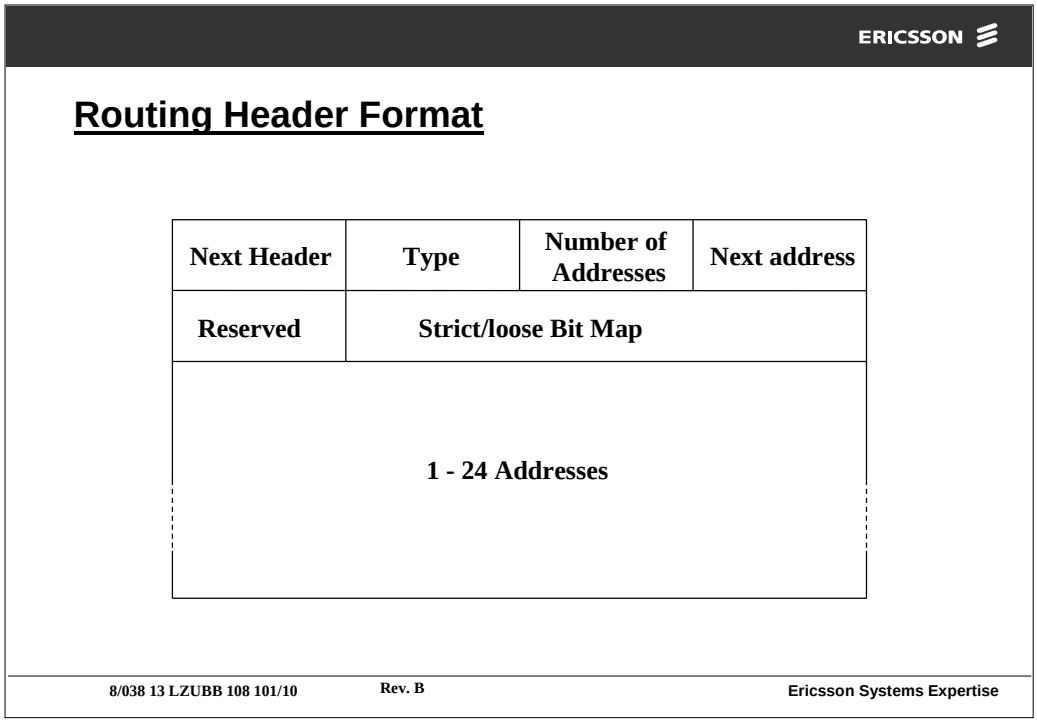


Routing Header

This routing header allows a source node to specify a list of IP addresses that dictate what path a packet will traverse. RFC 1883 defines a version of this routing header called “Type 0”, which gives a sending node a great deal of control over each packet’s route. Type zero routing headers contain a 24-bit field that indicates how intermediate nodes may forward a packet to the next address in the routing header. Each bit in the 24-bit field indicates whether the next corresponding destination address must be a neighbour of the preceding address (1=strict, must be a neighbour; 0=loose, need not be a neighbour).

When routing headers are used for “strict” forwarding, a packet visits only routers listed in the routing header. For example, if routers 2 and 3 are listed as strict but are not adjacent to each other (that is, in order to get from 2 to 3, a packet must pass through some other router), packets will be dropped at 2. This is a valuable feature when security and traffic control require that packets take a rigidly defined path. In “loose” forwarding, unlisted routers can be visited by a packet.

When Type 0 routing headers are used, the initial IPv6 header contains the destination address of the first router in the source route, not the final destination address.



If a router does not recognise the routing type value, it must discard the packet. Type 0 routing has been defined and has the following fields:

•**Next Header (8 bits):** identifies the type of header immediately following this header.

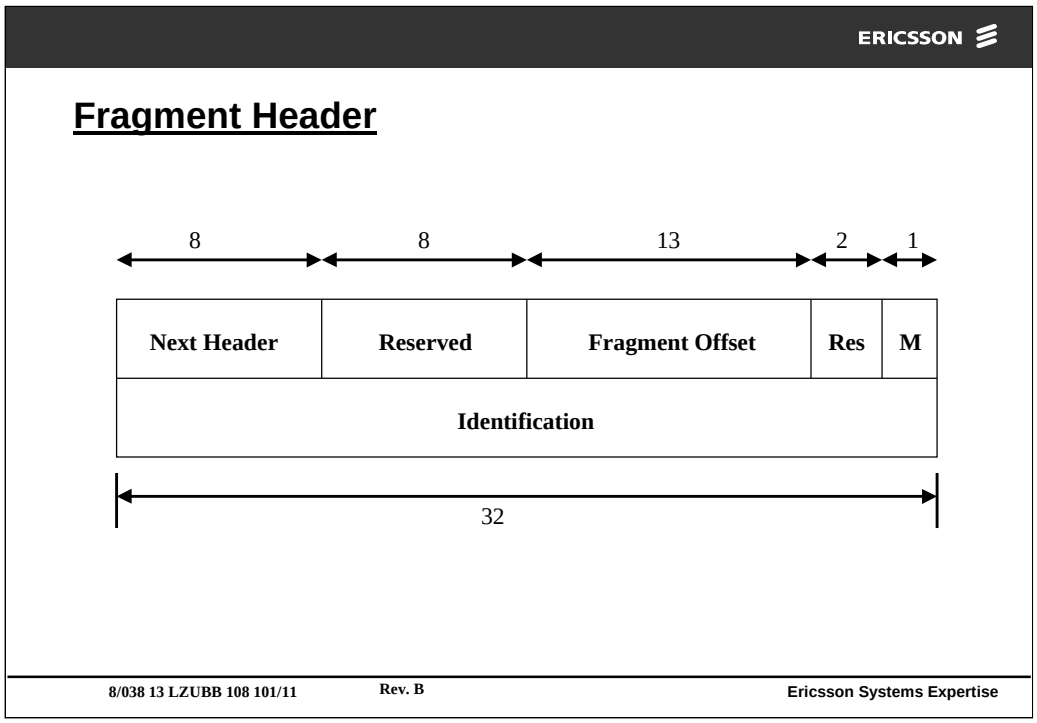
•**Routing Type (8 bits):** currently set to zero **.**

•**Num Addrs (8 bits):** number of addresses in the routing header; the maximum value is 24.

•**Next Addr (8 bits):** index of the next address to be processed; initialised to zero by the originating node, and is incremented as each address is visited.

•**Reserved (8 bits):** for future use.

•**Strict/Loose Bit Mask (24 bits):** numbered from left to right, with each bit corresponding to one of the addresses. Each bit indicates whether the corresponding next destination address must be a neighbour of the preceding address (1 = strict, must be a neighbour; 0 = loose, need not be a neighbour).



Fragment Header

In IPv6, fragmentation may only be performed by source nodes, not by routers along a packet's delivery path. End nodes performing fragmentation can determine the smallest MTU of a path using the MTU path discovery process. The source node sends out a packet with an MTU as large as the local interface can handle. If this MTU is too large for some link along the path, an ICMP "Datagram too big" message will be sent back to the source. This message will contain a "Datagram too big" indicator and the MTU of the affected link. The source can then adjust the packet size downward and retransmit another packet. This process is repeated until a packet gets all the way to the destination node. The discovered MTU is then used for fragmentation purposes.

The fragment header consists of the following:

•**Next Header (8 bits):** identifies the type of header immediately following this header.

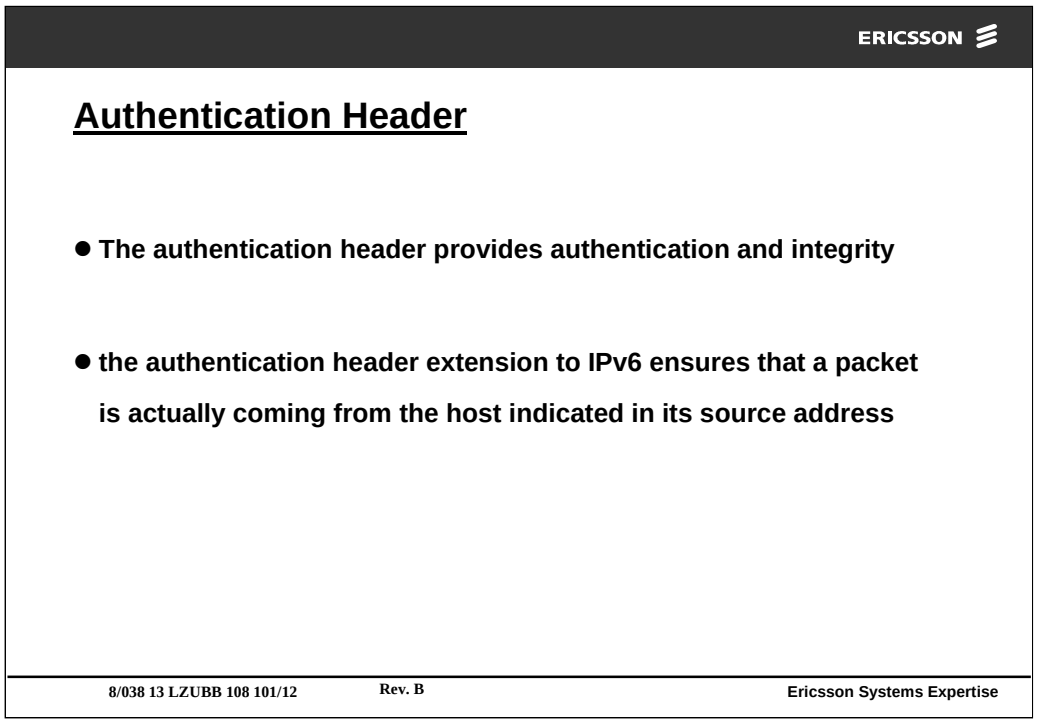
•**Reserved (8 bits):** for future use.

•**Fragment Offset (13 bits):** indicates where in the original packet the payload of this fragment belongs. It is measured in 64-bit units; which implies that fragments must contain a data field that is a multiple of 64 bits.

•**Res (2 bits):** reserved for future use.

•**M Flag (1 bit):** 1 = more fragments; 0 = last fragment.

•**Identification (32 bits):** intended to uniquely identify the original packet.

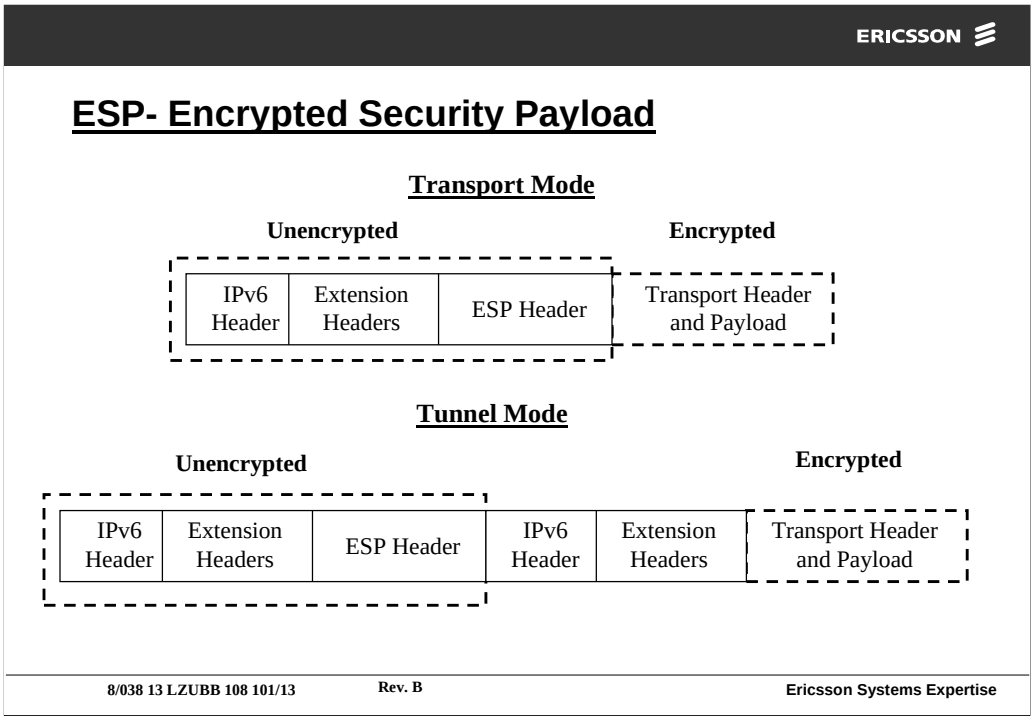


Authentication Header

The IPv6 authentication extension header gives network applications a guarantee that the packet did in fact come from an authentic source as indicated in its source address. This authentication is particularly important to safeguard against intruders who configure a host to generate packets with forged source addresses. This type of source address masquerading can spoof a server so that access may be gained to valuable data, passwords or network control utilities.

With IPv6 authentication headers, hosts establish a standards-based security association that is based on the exchange of algorithm-independent security keys (for example MD5).

In a client/server session, for instance, both the client and the server need to know the key. Before each packet is sent, IPv6 authentication creates a checksum based on the key combined with the entire contents of the packet. This checksum is then re-run on the receiving side and compared. This approach provides authentication of the sender and guarantees that data within the packet has not been modified by an intervening party. Authentication can take place between clients and servers or clients and clients on the corporate backbone. It can also be deployed between remote stations and corporate dial-in servers to ensure that the perimeter of the corporate security is not breached.



Encryption Security Payload Header

Authentication headers eliminate a number of host spoofing and packet modification hacks, but they do not prevent nondisruptive reading, such as sniffing (reading network traffic), of data as it traverses the Internet and corporate backbone networks. This area is dealt with by the Encapsulating Security Payload (ESP) service of IPv6 which provides integrity and confidentiality in IPv6 datagrams. ESP provides encryption at the network layer, making it available to all applications in a highly standardised fashion.

IPv6 ESP is used to encrypt the transport-layer header and payload (for example, TCP or UDP) or the entire IP datagram. Both these methods are accomplished with an ESP extension header that carries encryption parameters and keys end-to-end.

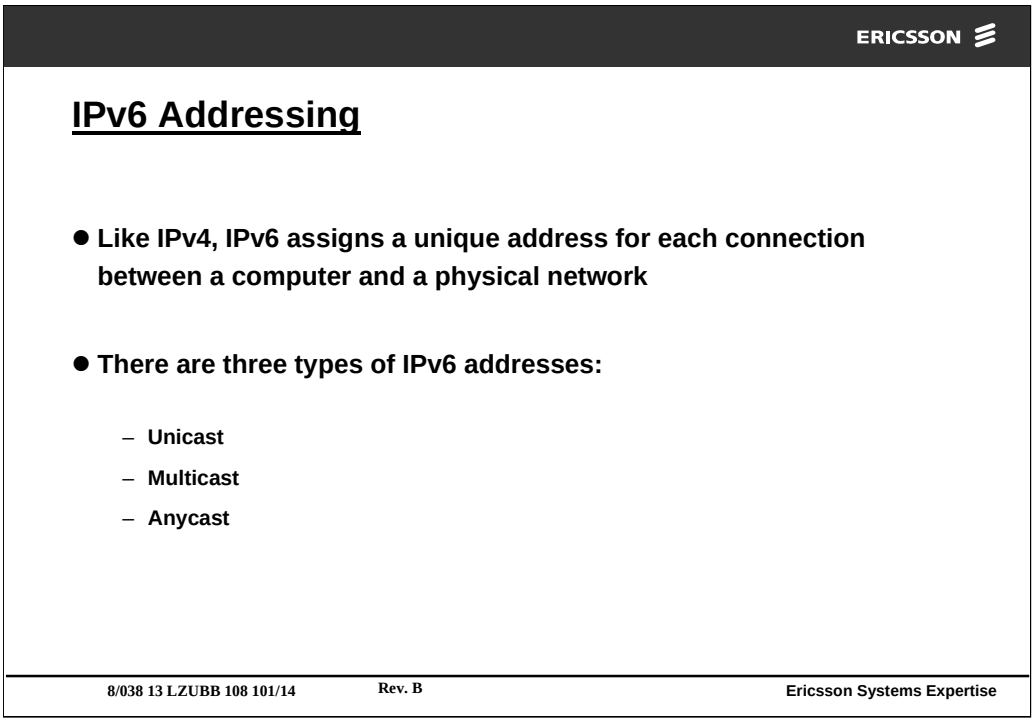
IPv6 has two modes to provide confidentiality:

•Transport mode ESP

In this mode only the payload is encrypted. The IP header and IP options are unencrypted and are used for routing the packet. The receiver decrypts the ESP and continues to use the unencrypted header as an IP header if necessary.

•Tunnel mode ESP

In this mode, the original IP datagram and header are encrypted. This entire ESP frame is placed within a new datagram with an unencrypted IP header. All additional unencrypted information such as routing header are placed between the IP header and the encapsulated security payload. The receiver strips off the cleartext IP header and decrypts the ESP.



Like IPv4, IPv6 assigns a unique address for each connection between a computer and a physical network. Thus, if a computer connects to three physical networks, the computer is assigned three addresses. IPv6 also separates each such address into a prefix that identifies the network and a suffix that identifies a particular computer on the network.

Although IPv6 adopts the same approach for assigning computer addresses, IPv6 addressing differs in several ways.

First, all address details are completely different.

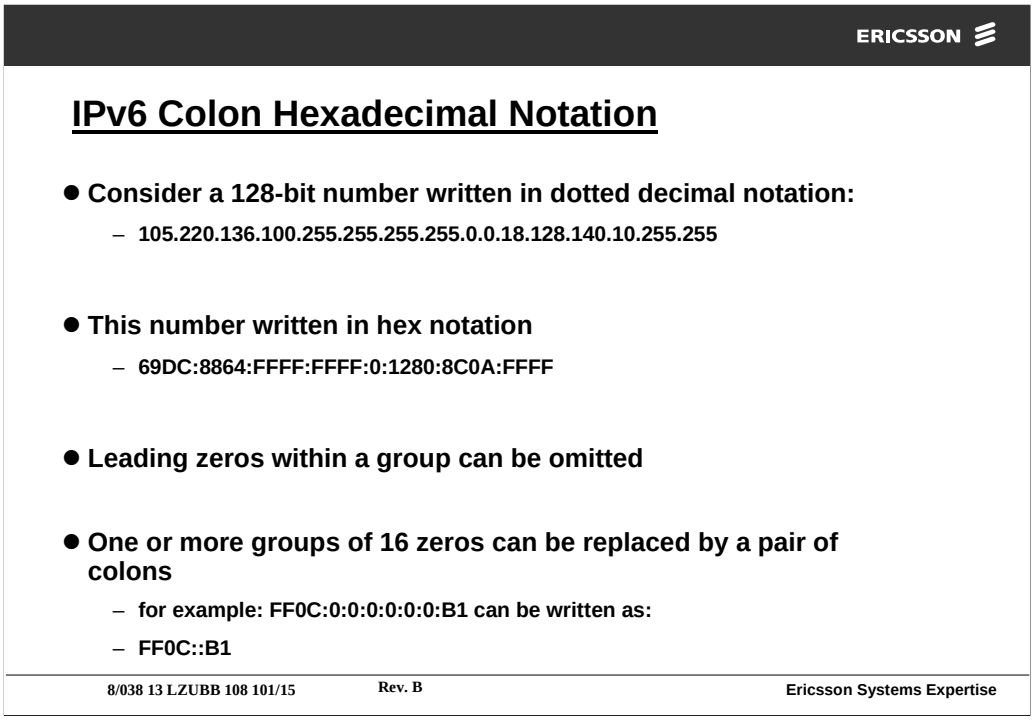
Second, IPv6 defines a set of special addresses that differ dramatically from IPv4 special addresses. In particular, IPv6 does not include a special address for broadcasting on a given network. Instead it uses a form of multicasting. There are three types of IPv6 addresses:

•**Unicast:** the address corresponds to a single computer. A datagram sent to the address is routed along the shortest path to the computer.

•**Multicast:** the address corresponds to a set of computers, possibly at many locations. Membership in the set can change at any time. When a datagram is sent to the address, IPv6 delivers one copy of the datagram to each member of the set.

•**Anycast:** the address corresponds to a set of computers that share a common address prefix. A datagram sent to the address is routed along the shortest path and delivered to one of the computers, typically the “nearest” one according to current routing protocol metrics.

For example, this would allow an enterprise to use a single anycast address to forward packets to a number of different routers on its ISPs backbone. If all of the providers routers have the same anycast address, traffic from the enterprise will have several redundant access points to the Internet. If one of the backbone routers goes down, the next nearest device automatically will receive the traffic.



Writing 128-bit numbers can be confusing. For example, consider a 128-bit number written in dotted decimal notation:

105.220.136.100.255.255.255.255.0.0.18.128.140.10.255.255

To help reduce the number of characters in an address, the designers of IPv6 propose using a more compact syntactic form known as hexadecimal notation in which each group of 16 bits is written in hexadecimal with a colon separating groups. For example, when the above number is written in colon hex, it becomes:

69DC:8864:FFFF:FFFF:0:1280:8C0A:FFFF

As illustrated in the example shown, colon hex notation requires fewer characters to express an address. An additional optimisation known as zero compression further reduces the size. Zero compression replaces sequences of zeroes with two colons. For example, the address:

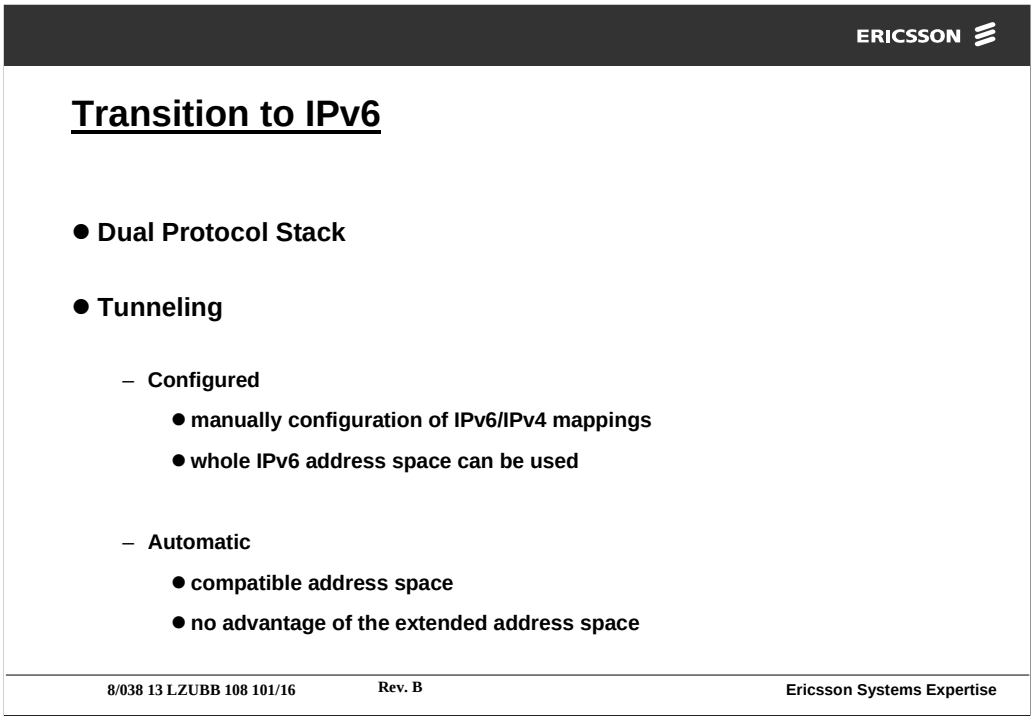
FF0C:0:0:0:0:0:0:B1

can be written as:

FF0C::B1

Leading zeros within a group can be omitted, so 0123 can be written as 123.

To help ease the transition to the new protocol, the designers mapped the existing IPv4 addresses into the IPv6 address space. Any IPv6 address that begins with 98 zero bits contains an IPv4 address in the low-order 32 bits.



IPv6 hosts and routers will need to retain backward compatibility with IPv4 devices for an extended time period (possibly years or even indefinitely) and will probably have the option of retaining their IPv4 addressing. To accomplish these goals, IPv6 transition relies on several special functions that have been built into the IPv6 standards work, including dual-stack hosts and routers and tunnelling IPv6 via IPv4.

Dual Stack

Once a few nodes have been converted to IPv6, there is the strong possibility that these nodes will require continued interaction with existing IPv4 nodes. This is accomplished with the dual-stack IPv4/IPv6 approach. When running a dual IPv4/IPv6 stack, a host can access both IPv4 and IPv6 resources. Routers running both protocols can forward traffic for both IPv4 and IPv6 end nodes. Dual Stack machines can use totally independent IPv4 and IPv6 addresses, or they can be configured with an IPv6 address that is IPv4 compatible.

Tunnelling

In most organisations where IPv6 is deployed incrementally, there is a strong possibility that all IPv6 hosts will not have direct connectivity to each other via IPv6 routers. In many cases there will be islands of IPv6 topology surrounded by an ocean of IPv4. Fortunately, IPv6 designers have fashioned transition mechanisms that allow IPv6 hosts to communicate over intervening IPv4 networks. The essential technique of these mechanisms is IPv6 over IPv4 tunnelling, which encapsulates IPv6 packets in IPv4 packets.

Tunnelling allows IPv6 implementations to co-exist with IPv4 without any change to IPv4 components. A dual-stack router or host on the “edge” of the IPv6 topology simply appends an IPv4 header to each IPv6 packet and sends it as native IPv4 traffic through existing links. IP v4 routers forward this traffic without knowledge that IPv6 is involved. On the other side of the tunnel, another dual-stack router or host de-encapsulates the IPv6 packet and routes it to the ultimate destination using standard IPv6 protocols.

To facilitate different administrative needs, IPv6 transition mechanisms include two types of tunnelling: automatic and configured.

To build configured tunnels, administrators manually define IPv6-to-IPv4 address mappings at the tunnel endpoints.

Automatic tunnels use “IPv4-compatible” addresses, which are hybrid IPv4/IPv6 addresses. Compatible addresses are created by adding leading zeros to the 32-bit IPv4 address to pad them out to 128 bits. When traffic is forwarded with compatible addresses, the device at the tunnel entry point can automatically address encapsulated traffic by simply converting the IPv4-compatible 128-bit address to a 32-bit IPv4 address. On the other side of the tunnel, the IPv4 header is removed to reveal the original IPv6 address. Automatic tunnelling allows IPv6 hosts to dynamically exploit IPv4 networks, but it does require the use of IPv4-compatible addresses, which do not bring the benefits of the 128-bit address space.

IPv6 nodes using IPv4-compatible addresses cannot take advantage of the extended address space, but they can exploit the other IPv6 enhancements, including flow labels, authentication, encryption, multicast and anycast. Once a node is migrated to IPv6 with IPv4-compatible addressing, the door is open for a fairly painless move to the full IPv6 address space.

Summary

•In chapter 8 we looked at IPv6 which is the new IP addressing protocol. We discussed the differences between IPv4 and IPv6.

•We examined the IPv6 packet format and how it differs from IPv4.

•We examined the base header format and extension header format and discussed them in detail.

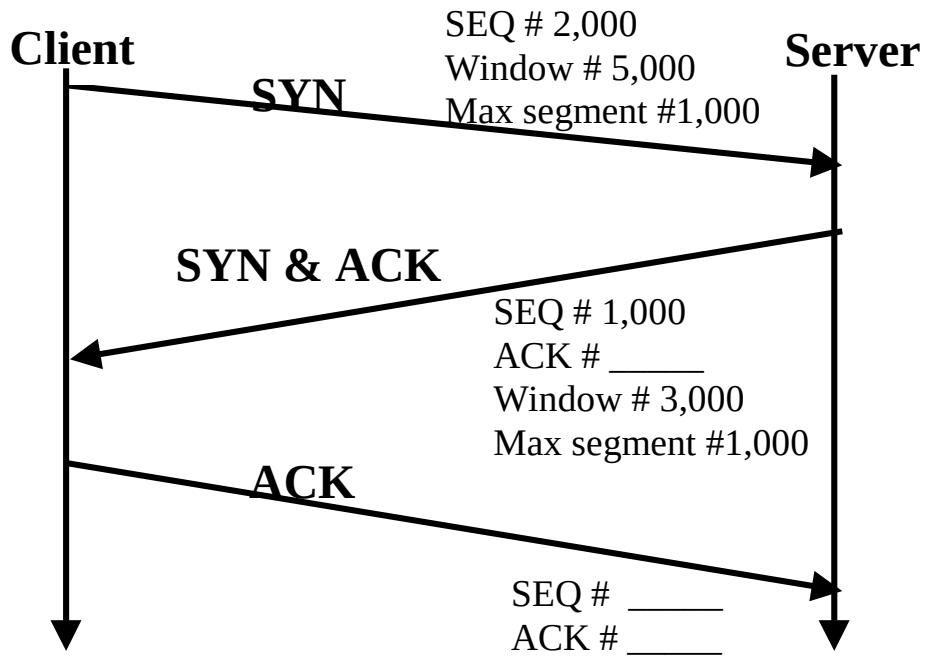
•We discussed IPv6 addressing and explained how IPv6 addresses are organised in hexadecimal notation.

•We discussed IPv4/IPv6 transition issues.

Exercises & Solutions

Exercise 1

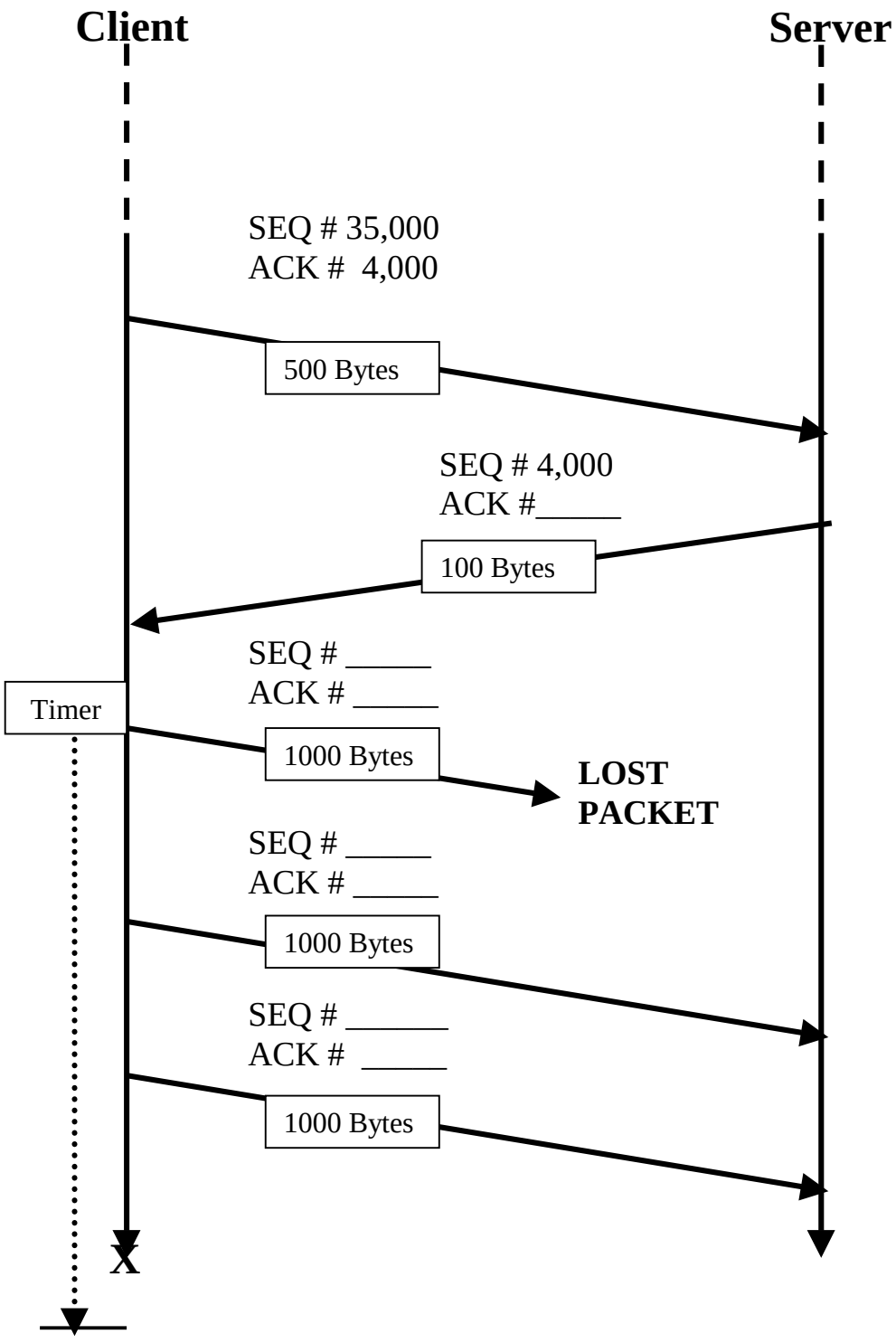
Opening a connection – TCP three-way handshake



1. Fill in the gaps above (sequence numbers and acknowledgement numbers).
2. The server's window size is _____ bytes.

3. Assume the diagram below represents the same TCP connection between the client and server later on. Fill in the gaps below (sequence numbers and acknowledgement numbers).

4. At the time marked X below, why can the client not send the next 1,000 byte segment? _____

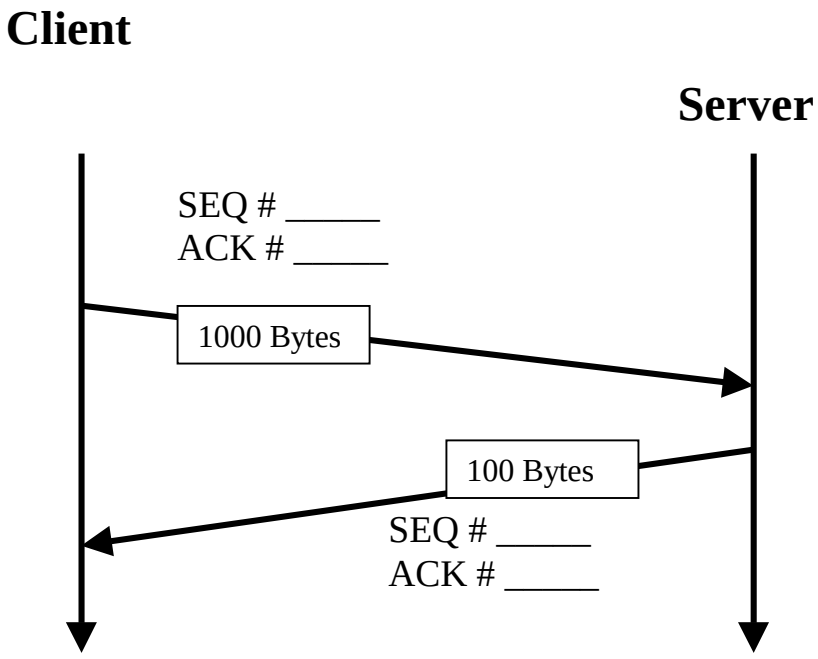


5. What will happen when the timeout period expires on the client?

_____

_____

6. Assume the diagram below represents the same TCP connection between the client and server later on and that the timeout period has just expired. Fill in the gaps below (sequence numbers and acknowledgement numbers).



TCP/IP Exercise

The following pages contain a trace of Etherenet frames between a PC and a router.

---------- Frame 1 ----------

Source: PC
Destination: Router

```
TCP:  ----- TCP header -----
TCP:
TCP:  Source port             = 1417
TCP:  Destination port        = 23 (Telnet)
TCP:  Initial sequence number = 91338341
TCP:  Data offset             = 24 bytes
TCP:  Flags                   = 02
TCP:                ..0. .... = (No urgent pointer)
TCP:                ...0 .... = (No acknowledgment)
TCP:                .... 0... = (No push)
TCP:                .... .0.. = (No reset)
TCP:                .... ..1. = SYN
TCP:                .... ...0 = (No FIN)
TCP:  Window                  = 8192
TCP:  Checksum                = 9EC5 (correct)
TCP:
TCP:  Options follow
TCP:  Maximum segment size    = 1460
TCP:
```

```
ADDR  HEX                                                ASCII
0000  AA 00 04 00 F4 43 00 A0  C9 61 84 02 08 00 45 00  .....C...a....E.
0010  00 2C DB 9A 40 00 80 06  07 47 A3 21 E9 A5 A3 21  .,..@....G.!...!
0020  E8 01 05 89 00 17 05 71  B6 65 00 00 00 00 60 02  .......q.e....`.
0030  20 00 9E C5 00 00 02 04  05 B4 7E 7E              .........~~
```

Frame 1 is the first frame from the PC to the router.

(1) Why is the data offset 24 bytes?

_____

(2) What is a more common size for the data offset field? _____
(3) What is the purpose of this TCP segment? _____
(4) What is the window size in bytes? _____
(5) **What is the maximum segement size?**_____

Take a note of the sequence number

{ Note: the HEX output gives all the information about the frame
The first 14 bytes is the Ethernet header AA 00 04 00 F4 43 00 A0 C9 61 84 02 08 00
The next 20 bytes is the IP header 45 00 00 2C DB 9A 40 00 80 06 07 47 A3 21 E9 A5 A3 21 E8 01
The next 24 bytes is the TCP header 05 89 00 17 05 71 B6 65 00 00 00 00 60 02 20 00 9E C5 00 00 02 04 05 B4 }

------------------ Frame 2 ----------------------------

Source: Router
Destination: PC

```
TCP:  ----- TCP header -----
TCP:
TCP:  Source port             = 23 (Telnet)
TCP:  Destination port        = 1417
TCP:  Initial sequence number = 1162329827
TCP:  Acknowledgment number   = 91338342
TCP:  Data offset             = 24 bytes
TCP:  Flags                   = 12
TCP:                ..0. .... = (No urgent pointer)
TCP:                ...1 .... = Acknowledgment
TCP:                .... 0... = (No push)
TCP:                .... .0.. = (No reset)
TCP:                .... ..1. = SYN
TCP:                .... ...0 = (No FIN)
TCP:  Window                  = 4288
TCP:  Checksum                = AD65 (correct)
TCP:
TCP:  Options follow
TCP:  Maximum segment size    = 536
TCP:
```

```
ADDR  HEX                                                ASCII
0000  00 A0 C9 61 84 02 AA 00  04 00 F4 43 08 00 45 00  ...a.......C..E.
0010  00 2C 00 00 00 00 FF 06  A3 E1 A3 21 E8 01 A3 21  .,.........!...!
0020  E9 A5 00 17 05 89 45 47  BE E3 05 71 B6 66 60 12  ......EG...q.f`.
0030  10 C0 AD 65 00 00 02 04  02 18 00 00              ...e........
```

Frame 2 is from the router to the PC.
(6) What is the purpose of this TCP segment?
(7) What is the maximum segement size?
(8) What is the window size?
(9) How many maximum size segments can the PC send without receiving an acknowledgement?

Take a note of the sequence number and the acknowledgement number.

---------- **Frame 3** ----------

Source: PC
Destination: Router

```
TCP:  ----- TCP header -----
TCP:
TCP:  Source port             = 1417
TCP:  Destination port        = 23 (Telnet)
TCP:  Sequence number         = 91338342
TCP:  Acknowledgment number   = 1162329828
TCP:  Data offset             = 20 bytes
TCP:  Flags                   = 10
TCP:                ..0. .... = (No urgent pointer)
TCP:                ...1 .... = Acknowledgment
TCP:                .... 0... = (No push)
TCP:                .... .0.. = (No reset)
TCP:                .... ..0. = (No SYN)
TCP:                .... ...0 = (No FIN)
TCP:  Window                  = 8576
TCP:  Checksum                = B0C6 (correct)
TCP:  No TCP options
TCP:
```

```
ADDR  HEX                                                ASCII
0000  AA 00 04 00 F4 43 00 A0  C9 61 84 02 08 00 45 00  .....C...a....E.
0010  00 28 DC 9A 40 00 80 06  06 4B A3 21 E9 A5 A3 21  .(..@....K.!...!
0020  E8 01 05 89 00 17 05 71  B6 66 45 47 BE E4 50 10  .......q.fEG..P.
0030  21 80 B0 C6 00 00 7E 7E  7E 7E 7E 7E              !.....~~~~~~
```

Frame 3 is from the router to the PC.
(10)Why is the data offset 20 bytes?
(11)What is the purpose of this TCP segment?

Take a note of the sequence number and the acknowledgement number

---------- Frame 4 ----------
Source: Router
Destination: PC

```
Telnet:----- Telnet -----
Telnet:
Telnet:IAC Will Echo
Telnet:IAC Will Suppress go-ahead
Telnet:IAC Do Terminal-type
Telnet:IAC Do Negotiate about window size
Telnet:
Telnet:
```

```
ADDR  HEX                                                ASCII
0000  00 A0 C9 61 84 02 AA 00  04 00 F4 43 08 00 45 C0  ...a.......C..E.
0010  00 34 00 01 00 00 FF 06  A3 18 A3 21 E8 01 A3 21  .4.........!...!
0020  E9 A5 00 17 05 89 45 47  BE E4 05 71 B6 66 50 18  ......EG...q.fP.
0030  10 C0 AE 56 00 00 FF FB  01 FF FB 03 FF FD 18 FF  ...V............
0040  FD 1F                                            ..
```

Frame 4 is from the router to the PC. Note that the analyser has decoded the Telnet info but not the TCP info. Therefore, you require the additional following information:
Sequence number is 1162329828 (Hex 45 47 BE E4).
Acknowledgement number is 91338342 (Hex 05 71 B6 66).
Length of data is 12 bytes (the last 24 hex numbers in the sequence above).

Take a note of the sequence number, the acknowledgement number and the length of data.

---------- Frame 5 ----------

Source: Router
Destination: PC

```
Telnet:----- Telnet -----
Telnet:
Telnet:<0D0A0D0A>User Access Verification<0D0A0D0A>Password:
Telnet:
```

```
ADDR  HEX                                                ASCII
0000  00 A0 C9 61 84 02 AA 00  04 00 F4 43 08 00 45 C0
...a.......C..E.
0010  00 52 00 02 00 00 FF 06  A2 F9 A3 21 E8 01 A3 21
.R.........!...!
0020  E9 A5 00 17 05 89 45 47  BE F0 05 71 B6 66 50 18
......EG...q.fP.
0030  10 C0 17 C8 00 00 0D 0A  0D 0A 55 73 65 72 20 41
..........User A
0040  63 63 65 73 73 20 56 65  72 69 66 69 63 61 74 69  ccess
Verificati
0050  6F 6E 0D 0A 0D 0A 50 61  73 73 77 6F 72 64 3A 20
on....Password:
```

Frame 5 is from the router to the PC. Note that the analyser has decoded the Telnet info but not the TCP info. Therefore, you require the additional following information:
Sequence number is 1162329840 (Hex 45 47 BE F0)
Acknowledgement number is 91338342 (Hex 05 71 B6 66)
Length of data is 42 bytes (the last 84 hex numbers in the sequence above)

Take a note of the sequence number, the acknowledgement number and the length of data.

---------- Frame 6 ----------

Source: PC
Destination: Router

```
TCP:  ----- TCP header -----
TCP:
TCP:  Source port             = 1417
TCP:  Destination port        = 23 (Telnet)
TCP:  Sequence number         = 91338342
TCP:  Acknowledgment number   = 1162329882
TCP:  Data offset             = 20 bytes
TCP:  Flags                   = 10
TCP:                ..0. .... = (No urgent pointer)
TCP:                ...1 .... = Acknowledgment
TCP:                .... 0... = (No push)
TCP:                .... .0.. = (No reset)
TCP:                .... ..0. = (No SYN)
TCP:                .... ...0 = (No FIN)
TCP:  Window                  = 8522
TCP:  Checksum                = B0C6 (correct)
TCP:  No TCP options
TCP:
```

```
ADDR  HEX                                                ASCII

0000  AA 00 04 00 F4 43 00 A0  C9 61 84 02 08 00 45 00
.....C...a....E.
0010  00 28 DD 9A 40 00 80 06  05 4B A3 21 E9 A5 A3 21
.(..@....K.!...!
0020  E8 01 05 89 00 17 05 71  B6 66 45 47 BF 1A 50 10
.......q.fEG..P.
0030  21 4A B0 C6 00 00 7E 7E  7E 7E 7E 7E             !J....~~~~~~
```

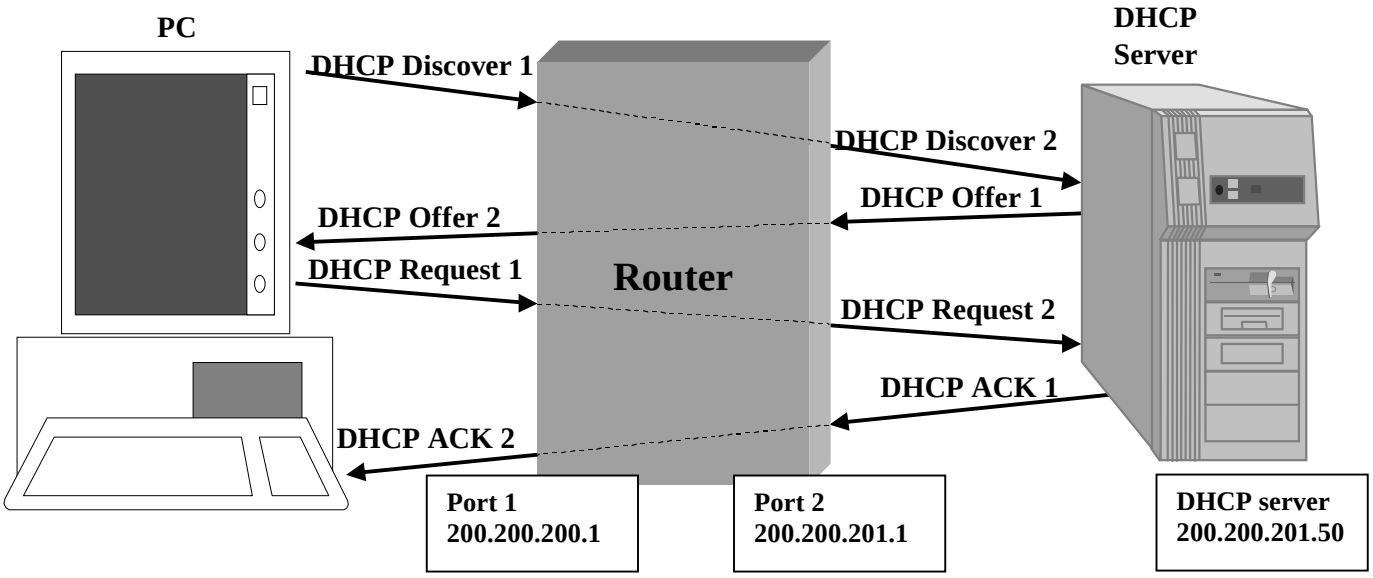
Take a note of the sequence number and the acknowledgement number.

	PC to Router		Router to PC	
Frame No.	Sequence number	Acknowledgement Number	Sequence number	Acknowledgement Number
1	#91338341	N/A		
2			#1162329827	#91338342
3	#	#		
4			#	#
5			#	#
6	#	#		

1. **Fill in the acknowledgement and sequence numbers in the table above.**
2. **Why is the sequence number of frame 5 twelve greater than the sequence number of frame 4?** _____
3. **Frame 5 was sent before an acknowledgement was received for frame four. Under what circumstances would this be impossible?**_____
   _____
   __
4. **Why is the sequence number of frame 6 the same as the sequence number of frame 3?**_____
5. **What does the acknowledgement number of frame 6 tell us?**
   _____
   _____

EXERCISE 3 - DHCP

Fill in the gaps in the sequence below.



		IP address
DHCP discover 1	Source Destination	0.0.0.0 255.255.255.255
DHCP discover 2	Source Destinatio	200.200.201.1 200.200.201.50
DHCP offer 1	Source Destination	? ?
DHCP offer 2	Source Destination	? ?
DHCP Request 1	Source Destination	? ?
DHCP Request 2	Source Destination	? ?
DHCP ACK 1	Source Destination	? ?
DHCP ACK 2	Source Destination	? ?

EXERCISE 4 - VARIABLE LENGTH SUBNET MASKING EXERCISE

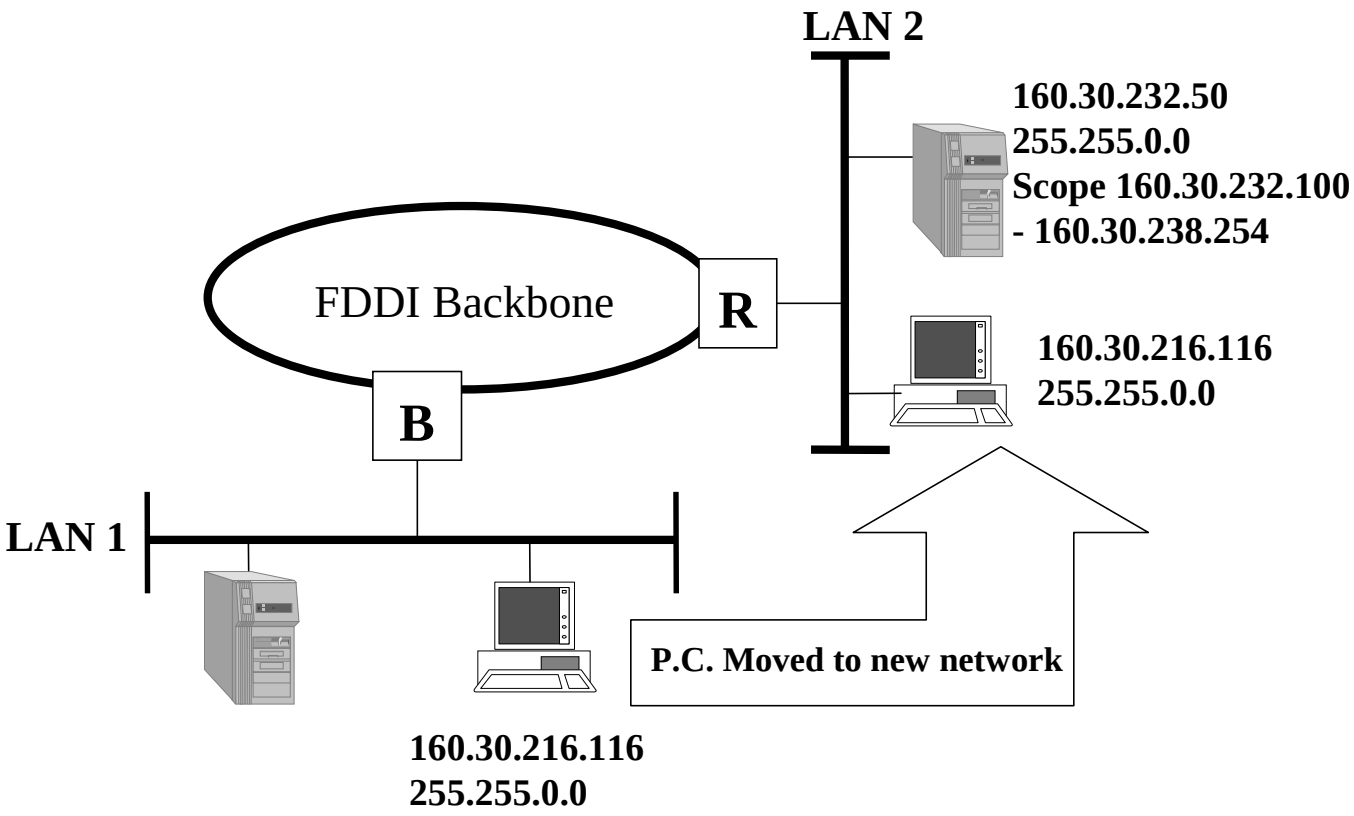
A company is allocated 4 class C addresses:
200.100.150.0
200.100.151.0
200.100.152.0
200.100.153.0

With no subnetting, they would have 4 separate networks, each with a maximum of 254 devices.

This is unsuitable for the company's requirements, which are as follows:
One "central" site with 200 hosts.
Two "large" remote sites with 50 and 40 hosts respectively.
Four "small" remote sites, two with 25 hosts and two with 20 hosts each.
All six remote sites are linked to the central site across a WAN link, with separate routers at both ends.

(1) Sketch a network diagram, showing the twelve routers and seven LANs.
(2) Calculate a network ID and subnet mask for every subnet, including the WAN links.
(3) Assign a valid IP address to every router interface (LAN and WAN) .

Exercise 5 - DHCP and proxy ARP



B = Bridge
R = Router
Note: The router on LAN 2 was configured such that valid IP addresses for that LAN were 160.30.232.1 to 160.30.239.254.

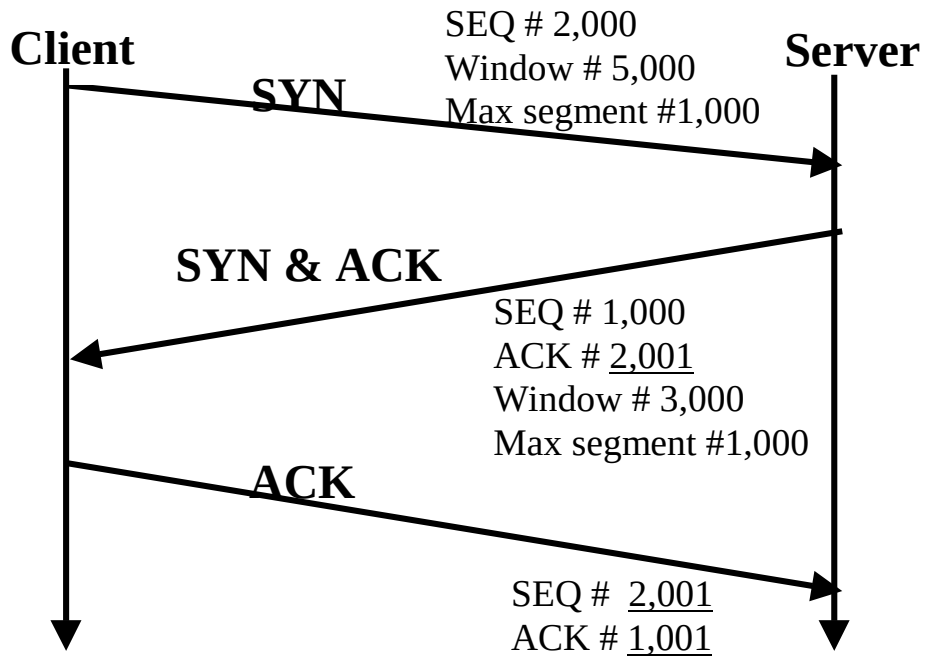
The PC was moved from LAN 1 to LAN 2. The expectation was that the PC would lose its IP address and get a new address which was valid for LAN2. This did not happen and the PC retained the address it had prior to moving.

Questions
1. Why did the PC not change IP address?
2. What must be done to ensure that for future moves, PCs will change address automatically?

Note: If you require additional information or clarification, please consult the instructor.

Solution 1

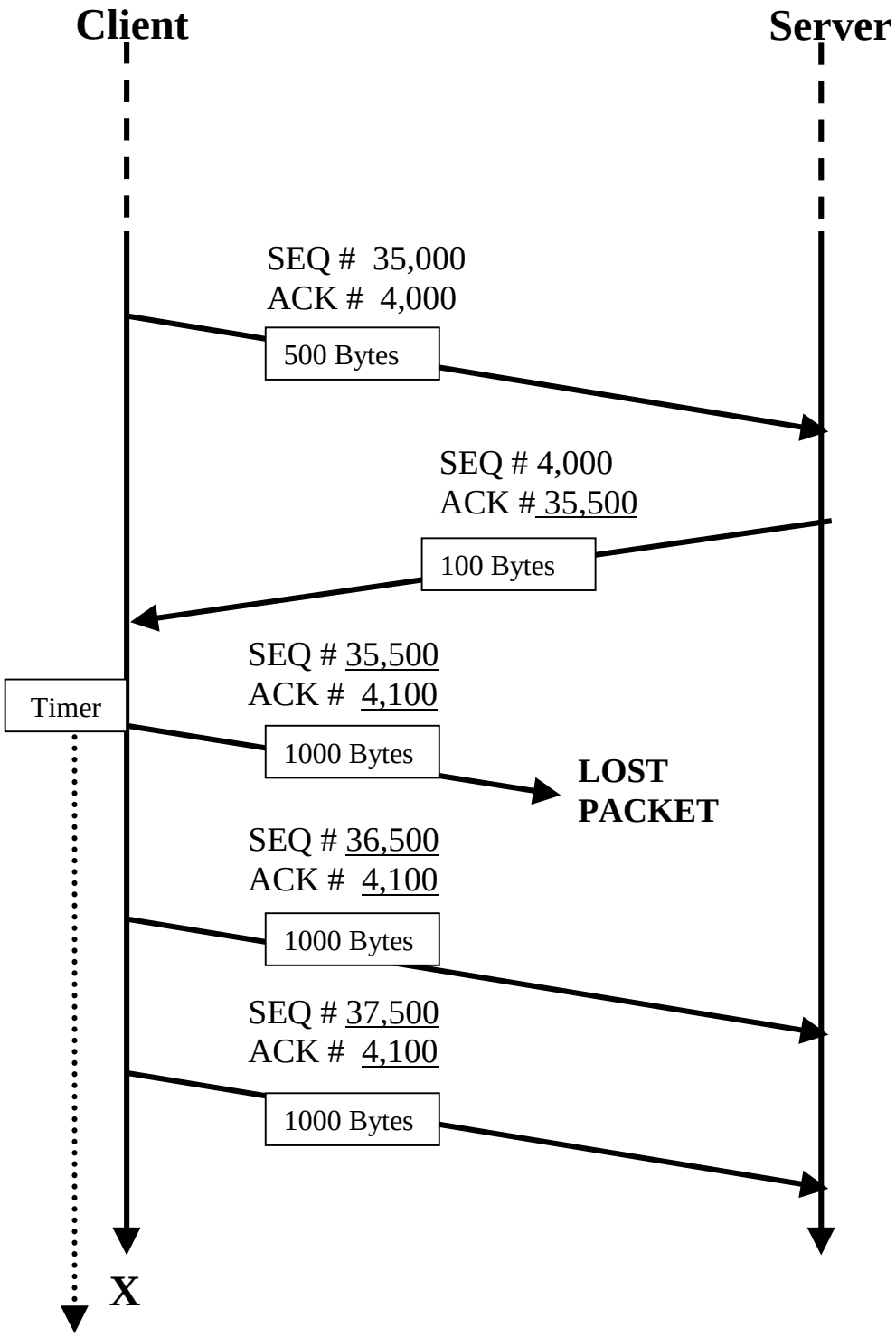
Opening a connection – TCP three-way handshake



1. Fill in the gaps above (sequence numbers and acknowledgement numbers). Note that if the SYN bit is set, TCP counts the segment as one data byte. Therefore the sequence number is incremented by one.
2. The server's window size is 3,000 bytes.

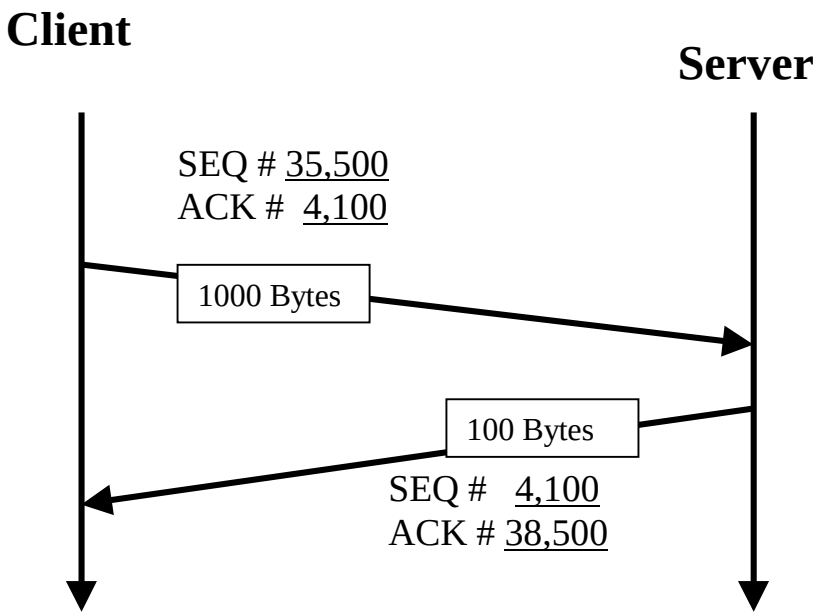
3. Assume the diagram below represents the same TCP connection between the client and server later on. Fill in the gaps below (sequence numbers and acknowledgement numbers).

4. At the time marked X below, why can the client not send the next 1,000 byte segment? The client can only send a maximum of 3,000 bytes to the server without receiving any acknowledgements. Otherwise it would exceed the server's window size.



5. What will happen when the timeout period expires on the client?
   The client will retransmit the packet which was lost (sequence number 35,500 and acknowledgement number 4,100). The server will store the segments which were out of sequence until the missing segment is received. Finally, the server will resequence the segments. The server will acknowledge all segments received as shown in the diagram below.

6. Assume the diagram below is later on in the same TCP connection between the client and server and that the timeout period has just expired. Fill in the gaps below (sequence numbers and acknowledgement numbers).



TCP/IP Solution 2

The attached pages contain a trace of Etherenet Frames between a PC and a Router.

---------- Frame 1 ----------

Source: PC
Destination: Router

```
TCP:  ----- TCP header -----
TCP:
TCP:  Source port             = 1417
TCP:  Destination port        = 23 (Telnet)
TCP:  Initial sequence number = 91338341
TCP:  Data offset             = 24 bytes
TCP:  Flags                   = 02
TCP:                ..0. .... = (No urgent pointer)
TCP:                ...0 .... = (No acknowledgment)
TCP:                .... 0... = (No push)
TCP:                .... .0.. = (No reset)
TCP:                .... ..1. = SYN
TCP:                .... ...0 = (No FIN)
TCP:  Window                  = 8192
TCP:  Checksum                = 9EC5 (correct)
TCP:
TCP:  Options follow
TCP:  Maximum segment size    = 1460
TCP:
```

```
ADDR  HEX                                                ASCII
0000  AA 00 04 00 F4 43 00 A0  C9 61 84 02 08 00 45 00  .....C...a....E.
0010  00 2C DB 9A 40 00 80 06  07 47 A3 21 E9 A5 A3 21  .,..@....G.!...!
0020  E8 01 05 89 00 17 05 71  B6 65 00 00 00 00 60 02  .......q.e....‘.
0030  20 00 9E C5 00 00 02 04  05 B4 7E 7E              .........~~
```

Frame 1 is the first frame from the PC to the router.
(1) Why is the data offset 24 bytes? Options are used, to specify maximum segment size.
(2) What is a more common size for the data offset field? 20 bytes
(3) What is the purpose of this TCP segment? Synchronisation
(4) What is the window size in bytes? 8,192
(5) What is the maximum segement size? 1,460

Take a note of the sequence number.

{ Note: the HEX output gives all the information about the frame
The first 14 bytes is the Ethernet Header AA 00 04 00 F4 43 00 A0 C9 61 84 02 08 00
The next 20 bytes is the IP header 45 00 00 2C DB 9A 40 00 80 06 07 47 A3 21 E9 A5 A3 21 E8 01
The next 24 bytes is the TCP header 05 89 00 17 05 71 B6 65 00 00 00 00 60 02 20 00 9E C5 00 00 02 04 05 B4 }

---------- Frame 2 ----------

Source: Router
Destination: PC

```
TCP:  ----- TCP header -----
TCP:
TCP:  Source port              = 23 (Telnet)
TCP:  Destination port         = 1417
TCP:  Initial sequence number = 1162329827
TCP:  Acknowledgment number    = 91338342
TCP:  Data offset              = 24 bytes
TCP:  Flags                    = 12
TCP:                ..0. .... = (No urgent pointer)
TCP:                ...1 .... = Acknowledgment
TCP:                .... 0... = (No push)
TCP:                .... .0.. = (No reset)
TCP:                .... ..1. = SYN
TCP:                .... ...0 = (No FIN)
TCP:  Window                   = 4288
TCP:  Checksum                 = AD65 (correct)
TCP:
TCP:  Options follow
TCP:  Maximum segment size     = 536
TCP:
```

```
ADDR  HEX                                                ASCII
0000  00 A0 C9 61 84 02 AA 00  04 00 F4 43 08 00 45 00  ...a.......C..E.
0010  00 2C 00 00 00 00 FF 06  A3 E1 A3 21 E8 01 A3 21  .,.........!...!
0020  E9 A5 00 17 05 89 45 47  BE E3 05 71 B6 66 60 12  ......EG...q.f‘.
0030  10 C0 AD 65 00 00 02 04  02 18 00 00              ...e........
```

Frame 2 is from the router to the PC.

(6) What is the purpose of this TCP segment? Synchronisation and acknowledgement

(7) What is the maximum segement size? 536

(8) What is the Window size? 4,288

(9) How many maximum size segments can the PC send without receiving an acknowledgement? Eight (4,288 divided by 536)

Take a note of the sequence number and the acknowledgement number.

---------- **Frame 3** ----------

Source: PC
Destination: Router

```
TCP:  ----- TCP header -----
TCP:
TCP:  Source port              = 1417
TCP:  Destination port         = 23 (Telnet)
TCP:  Sequence number          = 91338342
TCP:  Acknowledgment number    = 1162329828
TCP:  Data offset              = 20 bytes
TCP:  Flags                    = 10
TCP:                ..0. .... = (No urgent pointer)
TCP:                ...1 .... = Acknowledgment
TCP:                .... 0... = (No push)
TCP:                .... .0.. = (No reset)
TCP:                .... ..0. = (No SYN)
TCP:                .... ...0 = (No FIN)
TCP:  Window                   = 8576
TCP:  Checksum                 = B0C6 (correct)
TCP:  No TCP options
TCP:
```

```
ADDR  HEX                                                ASCII
0000  AA 00 04 00 F4 43 00 A0  C9 61 84 02 08 00 45 00  .....C...a....E.
0010  00 28 DC 9A 40 00 80 06  06 4B A3 21 E9 A5 A3 21  .(..@....K.!...!
0020  E8 01 05 89 00 17 05 71  B6 66 45 47 BE E4 50 10  .......q.fEG..P.
0030  21 80 B0 C6 00 00 7E 7E  7E 7E 7E 7E              !.....~~~~~~
```

Frame 3 is from the router to the PC.
(10)Why is the data offset 20 bytes? No options
(11)What is the purpose of this TCP segment? Acknowledgement

Take a note of the sequence number and the acknowledgement number

---------- Frame 4 ----------

Source: Router
Destination: PC

```
Telnet:----- Telnet -----
Telnet:
Telnet:IAC Will Echo
Telnet:IAC Will Suppress go-ahead
Telnet:IAC Do Terminal-type
Telnet:IAC Do Negotiate about window size
Telnet:
Telnet:
```

```
ADDR  HEX                                                ASCII
0000  00 A0 C9 61 84 02 AA 00  04 00 F4 43 08 00 45 C0  ...a.......C..E.
0010  00 34 00 01 00 00 FF 06  A3 18 A3 21 E8 01 A3 21  .4.........!...!
0020  E9 A5 00 17 05 89 45 47  BE E4 05 71 B6 66 50 18  ......EG...q.fP.
0030  10 C0 AE 56 00 00 FF FB  01 FF FB 03 FF FD 18 FF  ...V............
0040  FD 1F                                            ..
```

Frame 4 is from the router to the PC. Note that the analyser has decoded the Telnet info but not the TCP info. Therefore you require the additional following information:
Sequence number is 1162329828 (Hex 45 47 BE E4).
Acknowledgement number is 91338342 (Hex 05 71 B6 66).
Length of data is 12 bytes (the last 24 hex numbers in the sequence above).

Take a note of the sequence number, the acknowledgement number and the length of data.

---------- Frame 5 ----------

Source: Router
Destination: PC

```
Telnet:----- Telnet -----
Telnet:
Telnet:<0D0A0D0A>User Access Verification<0D0A0D0A>Password:
Telnet:
```

```
ADDR  HEX                                                ASCII
0000  00 A0 C9 61 84 02 AA 00  04 00 F4 43 08 00 45 C0
...a.......C..E.
0010  00 52 00 02 00 00 FF 06  A2 F9 A3 21 E8 01 A3 21
.R.........!...!
0020  E9 A5 00 17 05 89 45 47  BE F0 05 71 B6 66 50 18
......EG...q.fP.
0030  10 C0 17 C8 00 00 0D 0A  0D 0A 55 73 65 72 20 41
..........User A
0040  63 63 65 73 73 20 56 65  72 69 66 69 63 61 74 69  ccess
Verificati
0050  6F 6E 0D 0A 0D 0A 50 61  73 73 77 6F 72 64 3A 20
on....Password:
```

Frame 5 is from the router to the PC. Note that the analyser has decoded the Telnet info but not the TCP info. Therefore you require the additional following information:
Sequence number is 1162329840 (Hex 45 47 BE F0).
Acknowledgement number is 91338342 (Hex 05 71 B6 66).
Length of data is 42 bytes (the last 84 hex numbers in the sequence above).

Take a note of the sequence number, the acknowledgement number and the length of data.

---------- Frame 6 ----------

Source: PC
Destination: Router

```
TCP:  ----- TCP header -----
TCP:
TCP:  Source port              = 1417
TCP:  Destination port         = 23 (Telnet)
TCP:  Sequence number          = 91338342
TCP:  Acknowledgment number    = 1162329882
TCP:  Data offset              = 20 bytes
TCP:  Flags                    = 10
TCP:                ..0. .... = (No urgent pointer)
TCP:                ...1 .... = Acknowledgment
TCP:                .... 0... = (No push)
TCP:                .... .0.. = (No reset)
TCP:                .... ..0. = (No SYN)
TCP:                .... ...0 = (No FIN)
TCP:  Window                   = 8522
TCP:  Checksum                 = B0C6 (correct)
TCP:  No TCP options
TCP:
```

```
ADDR  HEX                                                ASCII

0000  AA 00 04 00 F4 43 00 A0  C9 61 84 02 08 00 45 00
.....C...a....E.
0010  00 28 DD 9A 40 00 80 06  05 4B A3 21 E9 A5 A3 21
.(..@....K.!...!
0020  E8 01 05 89 00 17 05 71  B6 66 45 47 BF 1A 50 10
.......q.fEG..P.
0030  21 4A B0 C6 00 00 7E 7E  7E 7E 7E 7E             !J....~~~~~~
```

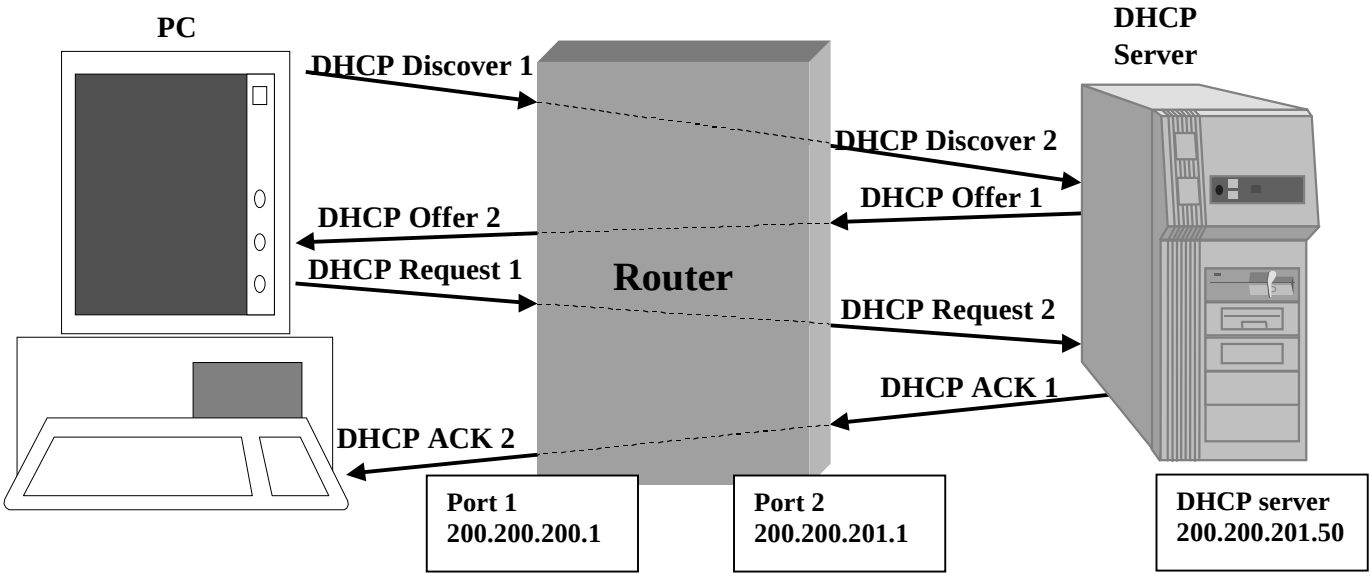
Take a note of the sequence number and the acknowledgement number.

	PC to Router		Router to PC	
Frame No.	Sequence number	Acknowledgement Number	Sequence number	Acknowledgement Number
1	91338341	N/A		
2			1162329827	91338342
3	91338342	1162329828		
4			1162329828	91338342
5			1162329840	91338342
6	91338342	1162329882		

1. **Fill in the acknowledgement and sequence numbers in the table above.**
2. **Why is the sequence number of frame 5 twelve greater than the sequence number of frame 4? Frame 4 included 12 bytes of TCP data**
3. **Frame 5 was sent before an acknowledgement was received for frame four. Under what circumstances would this be impossible? (a) If the timer which was started when frame 4 was transmitted has timed-out (b) If the receiver's buffer was full, that is, the window size has been reached.**
4. **Why is the sequence number of frame 6 the same as the sequence number of frame 3? No data was sent in frame 3. It was only an acknowledgement packet. Therefore, the sequence number is not incremented.**
5. **What does the acknowledgement number of frame 6 tell us? The PC successfully received frame 5, which contained 42 bytes of data.**

SOLUTION 3 - DHCP

Fill in the gaps in the sequence below.



		IP address
DHCP discover 1	**Source** **Destination**	**0.0.0.0** **255.255.255.255**
DHCP discover 2	**Source** **Destinatio**	**200.200.201.1** **200.200.201.50**
DHCP offer 1	**Source** **Destination**	*200.200.201.50* *200.200.201.1*
DHCP offer 2	**Source** **Destination**	*200.200.200.1* *255.255.255.255*
DHCP Request 1	**Source** **Destination**	*0.0.0.0* *255.255.255.255*
DHCP Request 2	**Source** **Destination**	*200.200.201.1* *200.200.201.50*
DHCP ACK 1	**Source** **Destination**	*200.200.201.50* *200.200.201.1*
DHCP ACK 2	**Source** **Destination**	*200.200.200.1* *255.255.255.255*

VARIABLE LENGTH SUBNET MASK SUGGESTED SOLUTION

Subnet mask of 255.255.255.0
1 X 254 host network 200.100.150.0

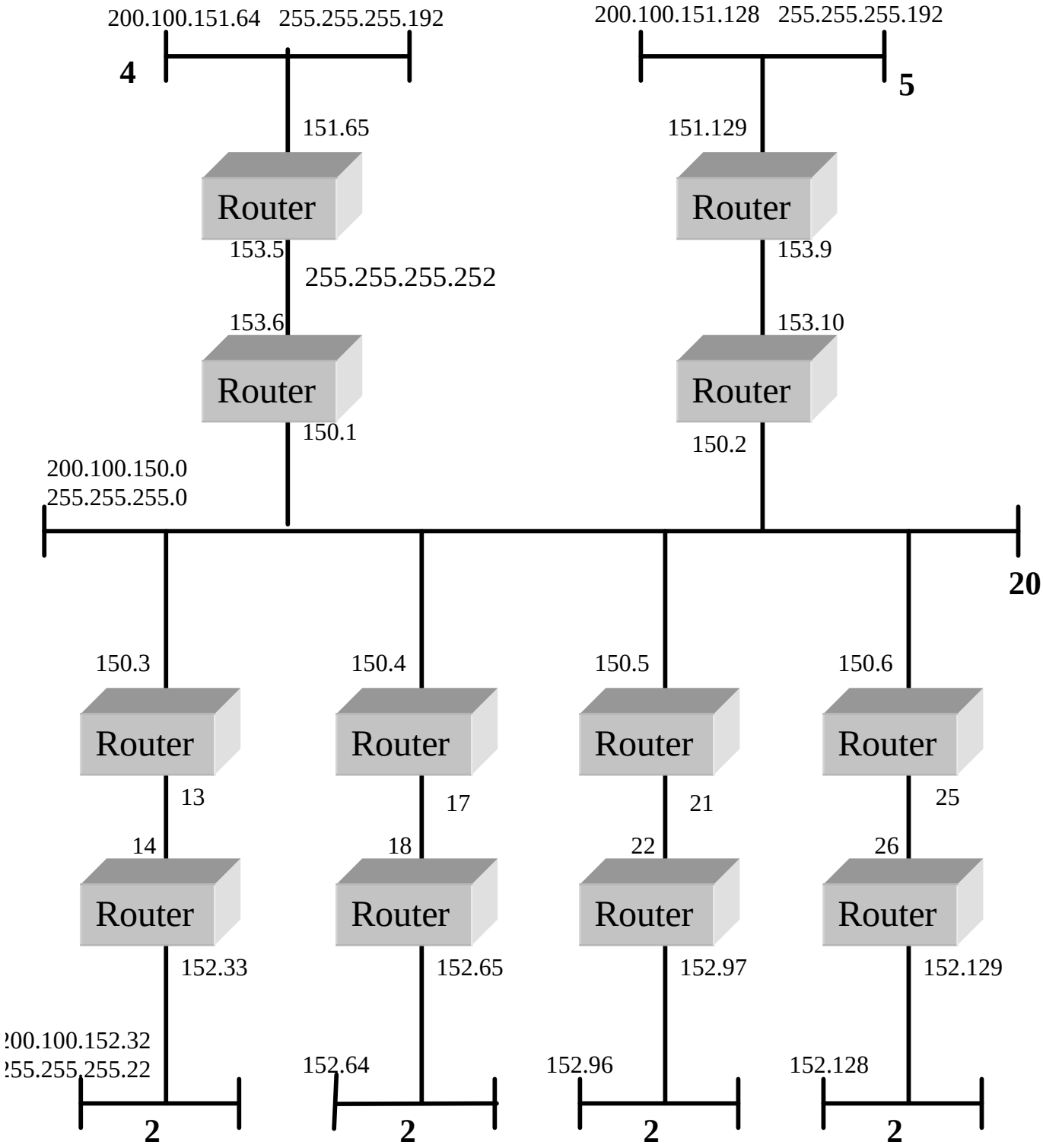
Subnet mask of 255.255.255.192
2 X 62 host networks 200.100.151.64
200.100.151.128

Subnet mask of 255.255.255.224

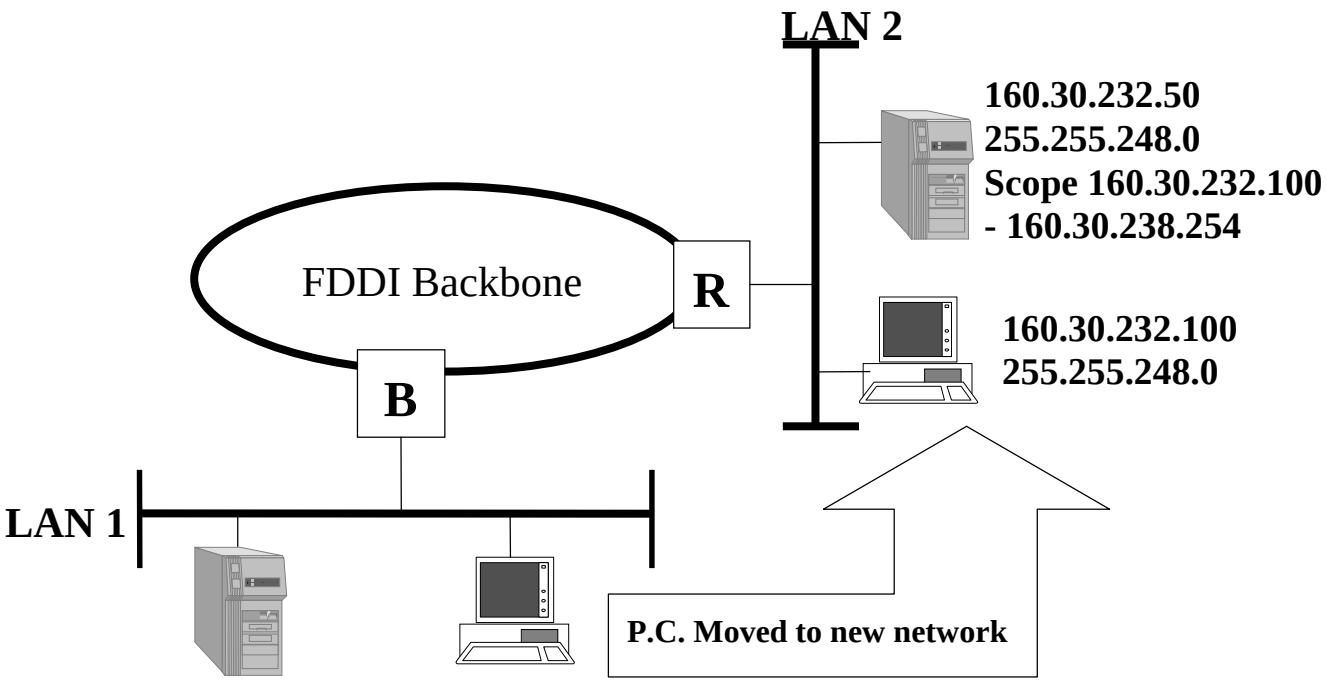
6 X 30 host networks 200.100.152.32
200.100.152.64
200.100.152.96
200.100.152.128
200.100.152.160
200.100.152.192

Subnet mask of 255.255.255.252
62 X 2 host networks 200.100.153.4
200.100.153.8
200.100.153.12
200.100.153.16
200.100.153.20
200.100.153.24
etc…..

Note: All IP addresses on this diagram start with 200.100.



Solution 5 - DHCP and proxy ARP



1. Why did the PC not change IP address?

To give up an IP address, the PC must receive a NACK (negative acknowledgement) packet from the DHCP server.
Although the PC's address is not in the server's scope, it is a valid address for that network as far as the server is concerned. This is because, according to the server's subnet mask (255.255.0.0), the network is 160.30.0.0, so any address that starts with 160.30 is valid. Therefore the server will not send a NACK.
The PC does not require a positive acknowledgement and will continue to run with the invalid address until the lease expires.

2. What must be done to ensure that for future moves PCs will change address automatically?

The solution is as shown in the diagram above.
The DHCP server must be configured to recognise that the only valid network addresses for LAN 2 are in the range 160.30.232.1 to 160.30.239.254.
This is achieved by giving the server a subnet mask of 255.255.248.0

Ericsson Radio System AB
MV/ERA/GDP/K
S-126 25 Stockholm, Sweden
Telephone: + 46 8 719 9222
http://internal-training.ericsson.se/