

## CCNA Extra Knowledge

Topic	Page
MISC Basic Networking Notes	236
The LLC – Logical Link Control (IEEE 802.2)	236
Late Collision	236
Duplex and Simple Communications	237
Manually Configured (Static) MAC Address	237
MISC Data Link Layer Notes	238
MISC Physical Layer Notes	239
Multiplexing Methods	241
Ethernet Autonegotiation and Duplex Mismatch	242
ISO-TP: OSI Transport Layer Protocols	243
The TCP Connection States	244
HTTP and TCP TIME_WAIT State	247
Dissecting TCP and IP Header Fields	247
TCP Selective Acknowledgment (SACK) and Fast Retransmission	250
Dive into Proxy ARP	251
Local Proxy ARP	251
Determining Cisco Router Memory Size	253
The Myth of the Running Configuration file	253
Spanning Tree Protocol Port ID	254
Router-on-a-Stick VLAN Trunking Encapsulation and IP Address Configuration	254
<b>switchport trunk encapsulation</b> and <b>switchport mode trunk</b> Configuration	255
The Problem of Router-on-a-Stick Configuration on Ethernet Interface	256
Switching Technologies	257
Dissecting Subnet Zero	257
Default Route (Gateway of Last Resort) Configurations	259
The <b>passive-interface</b> Router Subcommand	259
The <b>permanent</b> keyword in the Static Route Configuration	260
RIPv1 and VLSM	262
Fun with NATs and ACLs (Firewalls)	263
NAT with Virtual Interface	266
NAT Attack Session – <b>ip nat inside source</b> and <b>ip nat outside source</b>	267
Complex ACL	269
The Access Control List <b>established</b> Keyword	270
The Access Control List <b>fragments</b> Keyword	271
Switch Port Access Control Lists	272
Advanced Access Control List Configurations	273
Optional PPP Commands	278
Unidirectional PPP PAP Authentication	278
Unidirectional PPP CHAP Authentication	283
Frame Relay DLCIs	285
IEEE 802.11 Standards and Specifications	286
IEEE 802.11 Frame Types	287
IEEE 802.11 Types and Subtypes	292
IEEE 802.1X Port-Based Authentication	293
VPN and IPsec Basics	295
Troubleshooting ISDN	303

## MISC Basic Networking Notes

- The most common network user applications on today's networks are **email, web browsers, instant messaging, collaboration, and databases**.
- Below are the 3 categories of network applications:
  - i) **Batch applications** are started by a human complete with no other interaction, eg: FTP and TFTP.
  - ii) **Interactive applications** include database updates and queries. A person requests data from the server and waits for a reply. Response time depends more on the server than the network.
  - iii) **Real-time applications** include VoIP and video. Network bandwidth is critical as these applications are time critical. Quality of Service (QoS) and sufficient network bandwidth are mandatory for these applications.

## The LLC – Logical Link Control (IEEE 802.2)

- LLC was originated from the **High-Level Data Link Control (HDLC)** and uses a subset of the HDLC specification. It is the upper data link sublayer that provides an interface for upper layers to deal with any type of MAC lower sublayer, eg: 802.3 Ethernet and 802.5 Token Ring in order to achieve **physical media independence**. It is used for managing the data link communication, defining **Service Access Points (SAPs)** to identify and encapsulate network layer protocols.  
**Note:** LLC is the same for various physical media, eg: Ethernet, Token Ring, and WLAN.

- LLC defines 3 types of operations (or services) for data communication:

<b>LLC Type 1</b>	<b>Connectionless and unreliable.</b> Allows network layer protocols to run on it. Generally used by network layer protocols that using a transport layer.
<b>LLC Type 2</b>	<b>Connection-oriented and reliable.</b> Allows the LLC sublayer to provide connection establishment, data acknowledgement, error recovery, and flow control (windowing or sliding window). A retransmission timer is started when an endpoint sends frames to its peer. The frames will be retransmitted if an acknowledgment is not received within the timeout interval. Generally used in LAN environments where network and transport layer protocols are not involved.
<b>LLC Type 3</b>	<b>Connectionless and reliable.</b>

- Connection-oriented LLC (**LLC2**) uses a 2-byte Control field (in the 802.2 LLC header).

## Late Collision

- Late collision is a type of collisions found in the CSMA/CD environment where a collision error is detected after the first 64 bytes of the frame have been sent. It occurs when 2 end systems encounter a collision upon frame transmission even they have performed collision detection. This condition can occur when the network is so large that the frame propagation from one end to another takes longer than the time used to perform collision detection.
- Late collisions are not retransmitted by the NIC (Layer 2). They are left for the upper layers (eg: TCP) to detect and recover the loss of data.

## Duplex VS Simplex Communications

<b>Duplex</b>	2 directly connected devices can communicate with another <b>in both directions</b> .
<b>Half-Duplex</b>	Allows communication in both directions, but only 1 direction at a time. 1 way (lane) with 2 directions. Ex: walkie-talkie – both speakers cannot speak and be heard at the same time. One must use “Over” to indicate the end of a speech before another one can speak.
<b>Full-Duplex</b>	Allows communication in both directions at the same time. 2 ways (lanes) with 2 directions (a two-lane road with one lane for each direction). Ex: Telephone – it allows both callers to speak and be heard at the same time.
<b>Simplex</b>	2 directly connected devices can communicate with another <b>only in 1 direction</b> . 1 way (lane) and 1 direction. Ex: Fiber optic strands (a pair of TX and RX strands is used to connect 2 devices).

## Manually Configured (Static) MAC Address

- A **static address** for a device can be restricted to a specific port, with a specific VLAN. A static address will not be removed from the MAC address table when an interface link is down – it is bound to the assigned interface and non-movable. When a static address is seen on another interface, the address will be ignored and will not be written to the address table. A static address cannot be (dynamically) learnt on another port until it is being removed from the running-config.
- The syntax for configuring a static MAC address is  
**mac-address-table static {mac-addr} vlan {vlan-id} interface {type num}**  
**mac-address-table static {mac-addr} {type num} vlan {vlan-id}** (Cisco IOS Release 12.0)
- The example below configures a static MAC entry for the NIC with 1111.1111.1111. Connection of the NIC to interfaces other than Fa0/1 will have connectivity but no accessibility.

```
Switch(config)#mac-address-table static 1111.1111.1111 vlan 1 int Fa0/1
```

## MISC Data Link Layer Notes

- Ethernet is an example of **connectionless** networks, which means that the destination end systems are not contacted prior to communication, nor is there any mechanism for the sender to know whether a frame has reached its intended destination.
- Every time a transparent bridge learns a MAC address, it would timestamp the entry. When the bridge sees a frame from this MAC address, it refreshes the timestamp. If the bridge does not hear from this source for a specific amount of time (aging timer), the bridge deletes the entry from its bridging table. The default MAC address table aging timer is 5 minutes.
- **Content-Addressable Memory (CAM)** is a special type of computer memory that implements the lookup-table function in a single clock cycle (single operation) using dedicated comparison circuitry. CAMs are hardware-based search engines that are much faster than algorithmic approaches for search-intensive applications. CAMs are especially popular in network devices, which require high-speed table lookup for packet switching and forwarding.
- The default ARP table aging time is 4 hours (14400 seconds); while the default MAC address table aging time is 5 minutes (300 seconds). Aging time is the period that an entry is being kept.
- The **arp timeout** *{seconds}* interface subcommand and **mac-address-table aging-time** *{seconds}* global configuration command modify the ARP table aging timer and MAC address table aging timer respectively.
- A **balanced** line is a transmission line that consists of 2 connectors of the same type which there is no master-slave relationship – the DTE and DCE are treated as equals. Each station may initialize, supervise, recovery from errors, and send frames at any time.
- A **balun** can be used to convert balanced signal to unbalanced signal.
- **Alignment error** is an error that occurs in Ethernet networks, where a received frame has extra bits – a number that is not divisible by 8. Alignment errors are generally caused by collisions and eventually frame damage.
- **Bit-oriented protocols** are data link layer protocols that transmit frames regardless of the frame content. **Byte-oriented protocols** use a specific character from the user character set to mark the boundaries of frames. As compared with byte-oriented protocols, bit-oriented protocols are more efficient and support full-duplex operation. As a result, byte-oriented protocols have been superseded or replaced by bit-oriented protocols.
- Broadcasts and IP networks are not limited to VLANs, even though it is very tempting to think so. If 2 switches are connected with a crossover cable, with one configured with VLAN 10 on all ports, and another configured with VLAN 20 on all ports; hosts that connected to the switches are able to communicate as they as on the same IP network!

## MISC Physical Layer Notes

- The maximum point-to-point transmitting distance for a full-duplex mode 100BaseTX/FX multimode fiber converter is 2KM; under half-duplex mode, this is limited to 412m.
- The maximum point-to-point transmitting distance for a full-duplex mode 100BaseTX/FX single-mode fiber converter is 120KM. It is not recommended for a 100BaseTX/FX single-mode fiber converter to run under half-duplex mode with single-mode fiber.
- To fully utilize the advantage of long-distance transmission with fiber optics, the 100BaseTX/FX converters should be deployed together with switches instead of hubs, as switches support full-duplex mode, which allows point-to-point connection up to 2KM for multimode fiber and up to 120KM for single-mode fiber.
- 1000BaseSX Gigabit ports on 10/100 and Gigabit switches only support point-to-point connections of up to 280m. Gigabit multimode-SX-to-single-mode-LX converters have the potential to extend the distance to 10KM (the maximum point-to-point transmitting distance is 10KM + 280m = 10280m).

- Below describes the differences between stacking and cascading:

<b>Stacking</b>	Connecting multiple switches or hubs together with an SCSI cable. The cable plugs into the rear port of each switch or hub. The physical switches form a single logical switch.
<b>Cascading</b>	Connecting multiple switches or hubs together with UTP cables. The cables plugs into the front port of each switch or hub.

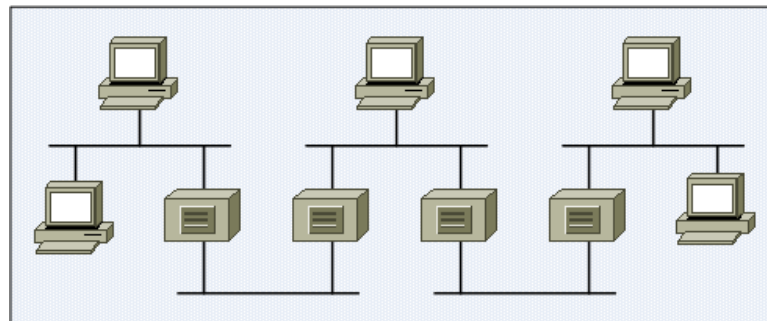
Stacking is the better solution when the devices are near to each other; while cascading is more suitable for linking devices that are further apart or differing makes (vendor, model, etc).

- **Note:** The backplane speed of a stack is limited to 32Gbps. For comparison, some larger modular switches can support 720Gbps on their backplanes.
- 2 pairs of wires (1 pair for **transmission**, another pair for collision detection and **reception**) in 100BaseTX UTP 8-pin connector are used for full-duplex transmission:

Pin Number	Signal
1	TX+
2	TX-
3	RX+
4	-
5	-
6	RX-
7	-
8	-

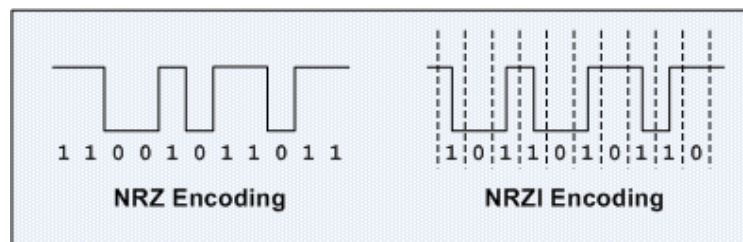
- **DOCSIS** – Data over Cable Service Interface Specification.

- The **5-4-3** repeater deployment rule – No more than 5 network segments can be connected end-to-end using 4 repeaters, but only 3 segments can have hosts on them. This rule only applicable to bus topologies that use repeaters or hubs, and not applicable to extended star topologies that use switches. This rule which is no longer applicable for the context of today’s networks, was the source of pain for those who didn’t ever notice or understand it.



**Figure A6-1:** The 5-4-3 Repeater Deployment Topology

- **Signal Quality Error (SQE)** is designed to fix the problem in earlier versions of Ethernet where a NIC does not know if a transceiver is connected. The SQE test is used to test the collision detection circuit between a transceiver and a NIC. After data is successfully transmitted, the Ethernet transceiver would assert the SQE signal on the collision detection circuit of the NIC. The NIC treats this as a verification that the transceiver will inform it when a collision occurs.
- The SQE test is no longer being used in modern Ethernet networks, as most NICs already have an integrated transceiver and hence the test for the collision detection circuit is unnecessary.
- NRZ and NRZI encoding schemes are used for transmitting digital data. Both signals are not self-clocking.
- **Non-Return to Zero (NRZ)** encoding is used in low speed synchronous and asynchronous links. With NRZ, a binary 1 bit signal is sent as a high voltage value and a binary 0 signal is sent as a low voltage value – there is no encoding at all! The receiver may lose synchronization when using NRZ due to long runs of consecutive bits with the same value (no changes in voltage).
- With **Non-Return to Zero Inverted (NRZI)** encoding, a 0 is encoded as no change or transition in the voltage level, while a 1 is encoded when there is a change or transition in the voltage level – either **from low to high** or **from high to low**.



**Figure A6-2:** The NRZ and NRZI Encoding Schemes

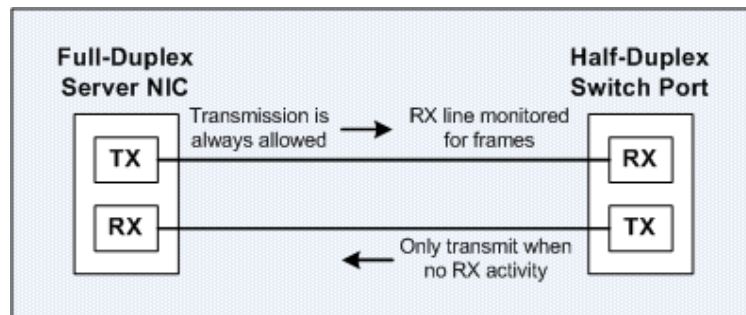
- Parity checking is a method of error checking in data transmission. An extra bit – the parity bit is added to each character or data word so that the sum of the bits will be either an odd number (**odd parity**) or an even number (**even parity**).

## Multiplexing Methods

- Multiplexing is a technology that is used to combine and send multiple digital signals simultaneously across a single physical transmission channel (wire, fiber, or link).
- Multiplexing occurs on the physical layer, as it combines multiple signals across a single physical channel.
- **Time-Division Multiplexing (TDM)** is a technique for assigning bandwidth on a single wire to data from several channels based on pre-assigned time slots. Bandwidth is allocated to each channel based on time slots, regardless of whether there is data to be transferred. Bandwidth will be wasted when there is no data to transfer. The SDH/SONET is an example application of TDM. **Note: Statistical Time-Division Multiplexing (STDM)** is an advanced version of TDM.
- **Frequency-Division Multiplexing (FDM)** is a form of signal multiplexing where multiplex baseband signals are modulated on different frequency carrier waves and combined together to create a composite signal. Data channels are allocated bandwidth based on the signal frequency of the traffic. Ex: FM radio.
- The concept corresponding to FDM in the optical domain is known as **Wavelength-Division Multiplexing (WDM)**. **Wavelength-Division Multiplexing (WDM)** and **Dense WDM (DWDM)** are technologies that are being used in fiber-optic communications which multiplex multiple optical carrier signals of a single fiber optic by using different wavelengths (colours) of laser light to carry different signals. Data channels are allocated bandwidth based on wavelength (inverse of frequency).
- With **Statistical-Division Multiplexing (SDM)**, the link sharing is adapted to the instantaneous traffic demands of the data streams that are translated over each channel (bandwidth is dynamically allocated to data channels). SDM is an alternative to multiplexing methods that create a fixed sharing of a link, eg: TDM and FDM.
- **Asynchronous Time-Division Multiplexing (ATDM)** is differs from normal TDM in which the time slots are assigned when necessary rather than pre-assigned to certain transmitters.

## Ethernet Autonegotiation and Duplex Mismatch

- Ethernet autonegotiation uses FLPs (**Fast Link Pulses**) and NLPs (**Normal Link Pulses**) bursts (or signals) to negotiate the speed and duplex settings as well as other autonegotiation parameters between 2 NICs to allow them to operate at the highest possible performance mode.
- Autonegotiation is an **active** method that negotiates the speed and duplex settings; while auto-sensing is a rather **passive** method that only detects the speed of operation – **it is impossible to passively determine the duplex mode**.
- Autonegotiation is a protocol; hence it only works if it is running on both sides of a link. Setting a switch port to a specific speed and duplex disables autonegotiation for duplex mode. As a result, the end system connecting to the switch port is not able to see autonegotiation parameters, and hence connects only at half-duplex – **duplex mismatch** is occurred.
- Duplex mismatch normally produce poor performance and data link errors (eg: input errors, CRC errors). Always remember to hardcode the speed and duplex settings at both ends or configure autonegotiation at both ends in order to avoid duplex mismatch!  
**Note:** The end system performs auto-sensing when the autonegotiation process fails.
- In half-duplex operation, the RX line is being monitored for frames. If the RX line is receiving a frame, no frames are sent until the RX line is clear. A collision would occur when a frame is received on the RX line while a frame is being sent on the TX line.
- In full-duplex operation, the RX line is not being monitored, and the TX line is always considered available. Collisions do not occur as the TX and RX lines are completely independent.



**Figure A6-3:** Common Autonegotiation Failure Scenario

- When one side of a link operates in full-duplex, and another side operates in half-duplex, a large number of collisions will occur on the half-duplex side; due to the full-duplex side transmits frames without checking its RX line, and chances are it will be sending frames constantly. The **chances of collision is high**, as when the half-duplex side transmit a frame after it sees the RX line is not receiving any frame, the full-duplex side also transmit a frame as it never check its RX line. The half-duplex side does not expect a frame to be received when it is sending a frame – a collision occurs.
- Additionally, the half-duplex side would have a hard-time for getting a chance to transmit, resulting in **poor performance** for communication with the device.
- Gigabit Ethernet uses a more robust autonegotiation mechanism and has low chances to fail. Therefore Gigabit Ethernet interfaces should always be set to autonegotiation.

## ISO-TP: OSI Transport Layer Protocols

- The OSI suite defines the following 5 transport layers protocols:

<b>TP0</b> (Transport Protocol Class 0)	Performs <b>segmentation</b> and <b>reassemble</b> functions.
<b>TP1</b> (Transport Protocol Class 1)	Performs <b>segmentation</b> , <b>reassemble</b> , and <b>error recovery</b> . It segments and sequences PDUs and retransmits them or reinitiates the connection if an excessive number of PDUs are unacknowledged.
<b>TP2</b> (Transport Protocol Class 2)	Performs <b>segmentation</b> , <b>reassemble</b> , and <b>multiplexing</b> .
<b>TP3</b> (Transport Protocol Class 3)	Performs <b>segmentation</b> , <b>reassemble</b> , <b>error recovery</b> , and <b>multiplexing</b> . It segments and sequences PDUs and retransmits them or reinitiates the connection if an excessive number of PDUs are unacknowledged.
<b>TP4</b> (Transport Protocol Class 4)	Same with TP3. It is the most <b>commonly used</b> among OSI transport protocols. <b>Similar to TCP</b> in the TCP/IP suite.

- The protocols **increase in complexity** from Class 0 to 4.

## The TCP Connection States

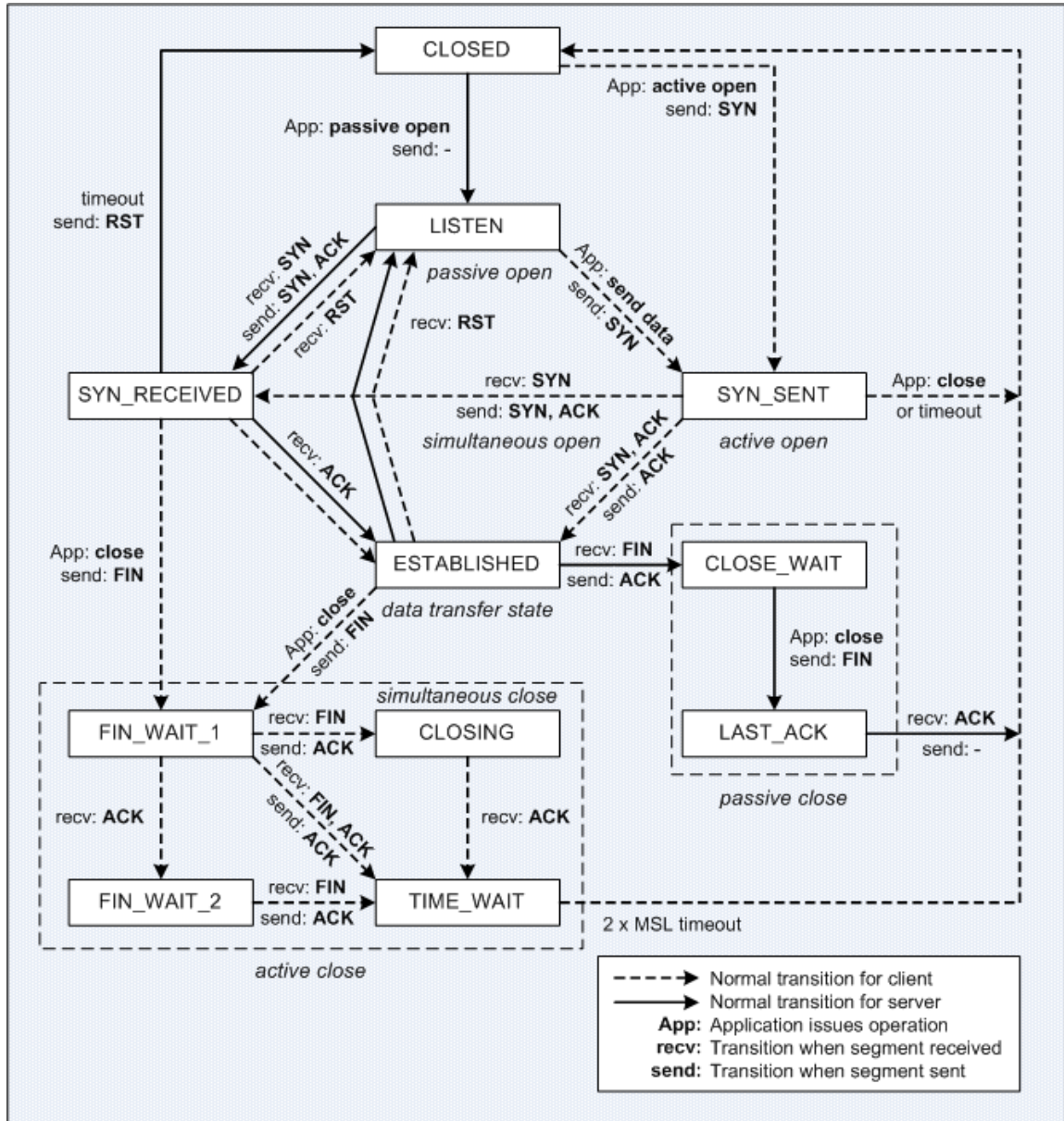


Figure A6-4: The TCP Finite State Machine

- In this context of discussion, a **client** is the peer that requests a connection; and a **server** is the peer that accepts a connection.
- **Handshake** is a series of information exchanged between devices to ensure both ends are synchronized for data transmission and operation.

<b>Active open</b>	A client socket initiates a connection by sending a SYN segment which contains the server's port number and the client's <b>Initial Sequence Number</b> (ISN). The client socket enters into the SYN_SENT state.
<b>Passive open</b>	Indicates a server socket is opened and listening or waiting for incoming SYN.
<b>Active close</b>	Initiated by the client socket when it finished receiving data. The client socket sends a FIN segment and enters into the FIN_WAIT_1 or FIN_WAIT_2 state. The connection is now a half-closed connection, where the client no longer sends data but still able to receive data from the server. The server socket enters into the passive close state upon receiving the FIN segment from the client.
<b>Passive close</b>	Occurs when the server socket receives the FIN segment from the client active close. The server socket enters into the CLOSE_WAIT state. The server socket is waiting for the server application to close. The server sends its FIN when the server application is closed.
<b>Simultaneous open</b>	Occurs when both client and server applications send a SYN to each other to establish a TCP connection. However, the possibly is small, as the server required to know the port of the client host that the SYN should destined to. TCP simultaneous open is normally found in <b>P2P applications</b> .
<b>Simul. close</b>	Occurs when both client and server applications perform active close.

- A connection goes through the following states during its lifetime:

<b>State</b>	<b>Description</b>
LISTEN	The server socket waits for a connection request from any remote client host.
SYN_SENT	The client socket waits for an acknowledgment after a connection request is sent.
SYN_RECEIVED	The server socket receives a connection request and pending to reply an acknowledgment to the client. The SYN Flood DoS attack would typically cause a lot of TCP connections in this state.
ESTABLISHED	Represents an established connection. Data communication is allowed. The normal state for the data transfer phase of a connection.
FIN_WAIT_1	The client socket has initiated active close and wait for an acknowledgment from the server after a connection termination request (FIN segment) is sent.
FIN_WAIT_2	The client socket has received an acknowledgment for its connection termination request but is still waiting for the connection termination request (FIN segment) from the server.
CLOSE_WAIT	The remote client socket is closed. The server socket enters into passive close state and waits for a connection termination request from the server application.
CLOSING	The client socket waiting for a connection termination acknowledgment from the remote server socket.
LAST_ACK	The server application and socket are closed. Waiting for the <b>final ACK</b> for the connection termination request from the remote client socket.
TIME_WAIT	The client socket waiting for enough time to pass to make sure the stray packets destined to the closed socket are flushed out from the network.

**Note:** Application layer protocol (eg: HTTP and Telnet) servers tend to initiates active close than clients, which reverse the client and server roles in TCP. TCP assumes that clients tend to initiate active close (by first sending a segment with the FIN bit set).

- **Q:** How does a client application enter the ESTABLISHED state from the CLOSED state?  
**A:** The client application calls the connect() socket function, which causes TCP to send an empty segment with the SYN bit set and enters into the SYN\_SENT state. The server then replies with an empty segment with the SYN and ACK bits set. When the client receives the SYN/ACK segment, it replies with an ACK segment, and reports a successful connection establishment to the client application.
  
- **Q:** What is the normal TCP shutdown sequence?  
**A:** TCP is a bidirectional protocol – the connection is shutdown in 2 identical phases, one for each direction. When the server finished sending data (the application protocol has finished using the connection, but TCP still has some work to perform), it sends a segment with the FIN bit set, which the client replies with a segment with ACK bit set. This sequence happens again when the server waits for the FIN segment from the client, and replies with an ACK segment to the client.  
**Note:** This is the normal TCP shutdown sequence observed in application layer protocols (eg: HTTP and Telnet), which reverse the client and server roles in TCP, as TCP assumes that clients tend to initiate active close (by first sending a segment with the FIN bit set).
  
- **Q:** What is the usage of the RST bit?  
**A:** It represents an abnormal close, which happens under several circumstances. The 2 common RST occurrences are “connection refused” and “connection terminated by remote host”. The 1st case happens when trying to connect to a non-open port on a remote host, while the 2nd case happens when the remote host interrupts the connection, the application crashes, or there is a malicious insertion of a segment with the correct IP and TCP information (eg: source and destination IP addresses, source and destination port numbers, sequence number) and RST bit set.
  
- **Q:** What’s wrong when connections keep getting into the FIN\_WAIT\_x state?  
**A:** Either the application or remote host is not closing the connection properly. Since FIN\_WAIT states often last up to 10 minutes, it is well worth the effort to find and fix the root cause.
  
- **Q:** What’s wrong when there are many connections in the TIME\_WAIT state?  
**A:** Nothing wrong. TIME\_WAIT is absolutely normal. Every socket that gets closed normally goes through this state after it is closed. This state is a safety mechanism that catches stray packets that are destined for a closed connection. Since the maximum time that such stray packets can exist is 2 times the **maximum segment lifetime**, hence the TIME\_WAIT state lasts for 2 x MSL. However, there is no easy way to estimate MSL on the fly, so protocol stacks normally hard-code a value (15 – 60 sec) for it. Hence, TIME\_WAIT usually lasts 30 – 120 sec.

**Glossary:**

<b>Stray packets</b>	Duplicate packets that are still in the network when a connection is closed.
<b>MSL</b>	The maximum length of time a TCP segment can exist (or alive) in a network.
<b>Stack</b>	The software implementation of a network protocol suite (eg: TCP/IP).

## HTTP and TCP TIME\_WAIT State

- Due to the nature of TCP/IP, it is possible that after an active close has commenced, there are duplicate packets that traversing around the network and trying to reach their destination sockets. If a new socket binds to the same port before these old packets are flushed out from the network, old and new data could be mixed up.
- In HTTP, active close always initiated by the server. A server socket enters the TIME\_WAIT state when it receives the last FIN from the client and replies with an ACK.
- Application protocol (eg: HTTP and Telnet) servers tend to initiates active close than clients, which reversed the client and server roles in TCP, as TCP assumes that clients tend to initiate active close. If that is the case, TIME\_WAITs won't be existed in a busy web server as they do. When the active close is initiated by clients, the TIME\_WAIT state and the responsibility of keeping old and new data from intermixing would tend to be on the client sockets.

## Dissecting TCP and IP Header Fields

4-bit	<b>Nibble</b>
8-bit	<b>Byte / Octet</b>
32-bit	<b>Word</b>

- **Header Length** or **Data Offset** (4 bits) specifies the number of 32-bit words in the TCP header, which indirectly indicates where the application data begins.
- **Unused** or **Reserved** (6 bits) bits are reserved for future use. All must be 0s.
- **Flags** or **Control Bits** (6 bits):

URG	Indicates that the Urgent Data Pointer field is significant.
ACK	Indicates that the Acknowledgment Number field is significant.
PSH	Push function.
RST	Reset the connection.
SYN	Synchronize the sequence numbers.
FIN	No more data from sender.

- Whenever a receiving TCP sees the PUSH flag, it will pass the data to the receiving process without waiting for more data – the receiving buffer that contains data associated with a PUSH flag will be passed to application for processing even if the buffer is not fully filled.
- **Checksum** (16 bits) is the 16-bit one's complement of the one's complement sum of the header, options, and application data. This field is filled with 0s when computing the checksum.  
**Note:** The Header Checksum field in the IPv4 header is a checksum that is calculated based on all the fields in the IPv4 header **only**; hence only the IPv4 header is being checked for errors. The Header Checksum field is filled with 0s when computing the checksum.

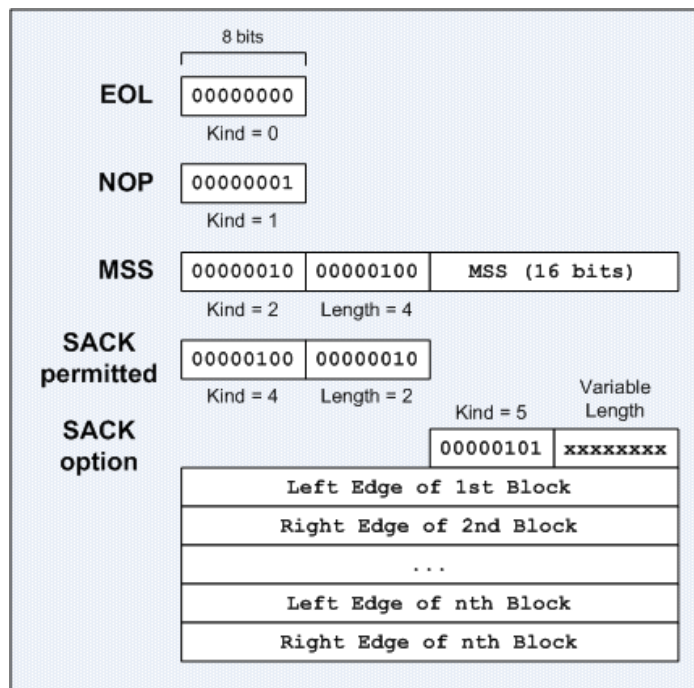
- **Urgent Data Pointer** (16 bits) indicates a positive offset from the current sequence number in the segment. It points to the sequence number of the octet following the urgent data (the end of the urgent data). This field is only being interpreted when the URG control bit in a segment is set. TCP must inform the application layer how much urgent data remains to be read from the connection whenever it receives a segment with an urgent data pointer.
- **Options** (variable) may exist at the end of the TCP header. Their lengths are multiple of 8 bits. The TCP checksum covers all the TCP options.
- Below lists some common TCP options:

Kind (8-bit)	Length (8-bit)	Description
0	-	<b>EOL</b> – End of Option List. Indicates the end of the option list. Used at the end of all options, not the end of each option.
1	-	<b>NOP</b> – No Operation. May be used between options to align the beginning of a subsequent option to a word boundary.
2	4	<b>MSS</b> – Maximum Segment Size. Indicates the maximum receive segment size at the TCP that sent out this segment. This option must only be sent in the initial connection request – connection request segments with SYN control bit set. Any segment size is allowed if this option is not used.
4	2	<b>SACK permitted</b> . The <b>Selective Acknowledgment (SACK)</b> extension uses 2 forms of TCP options: an enabling option – <b>SACK permitted</b> , that may be sent in a SYN segment to indicate the use of SACK option in an established connection; and the <b>SACK option</b> , which may be sent over an established connection after the SACK permitted is granted operation.
5	Variable	<b>SACK options</b> . They are used to convey extended acknowledgment information over an established connection. It is normally sent by a receiver to inform the sender about the <b>non-contiguous blocks of received data</b> , which mostly caused by the lost of a TCP segment. The receiver will wait for the retransmissions of the missing block of data to fill in the gap. A TCP client includes the SACK options along with the ACK segment to the TCP server whenever there is queued and unacknowledged data. SACK optimizes <b>TCP retransmissions with selective retransmission</b> , which skips the retransmission of selectively acknowledged segments. The SACK option can be sent only when the TCP client has received the SACK permitted option in the SYN segment from the TCP server, which indicates that the server supports SACK.

- The SACK option fields:

<b>Left Edge</b>	The first sequence number of this queued and unacknowledged data block.
<b>Right Edge</b>	The last sequence number of this queued and unacknowledged data block.

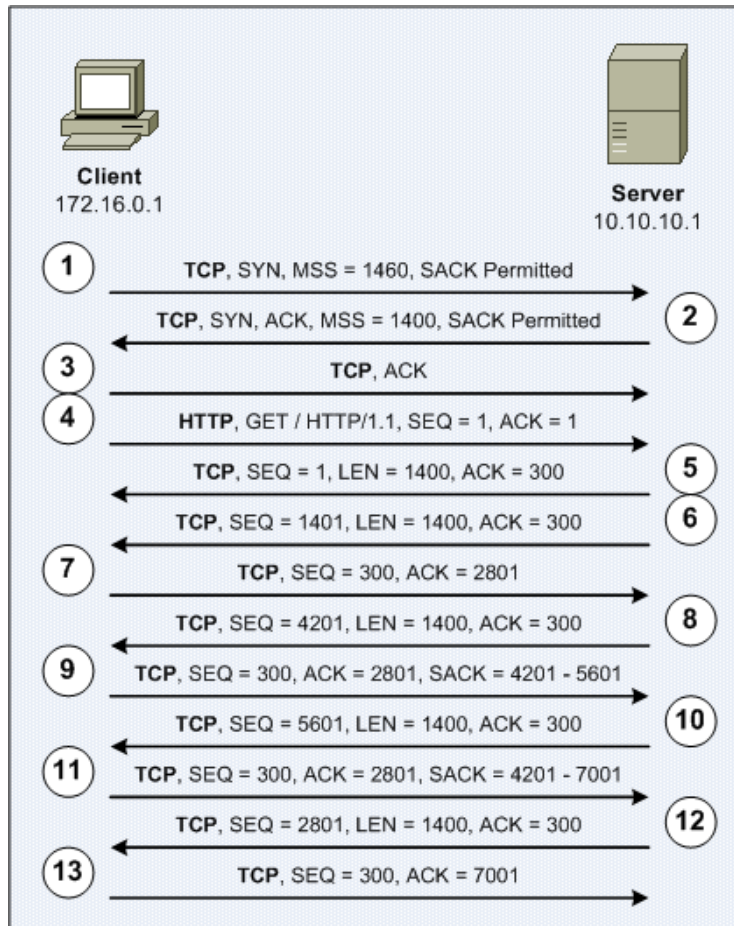
**Reference:** RFC 2018 – TCP Selective Acknowledgment Options.



**Figure A6-5:** TCP Options

- **Padding** (variable) which is comprised of 0s is used to ensure the TCP header is 32-bit aligned.
- The **Type of Service (TOS)** field in the IP header was intended to use for **TOS routing**, in which a router would have separate routing tables for different TOS values. When forwarding a packet, the router would first choose a routing table based on the packet's TOS, followed by normal routing table lookup. TOS routing has rarely been implemented in the Internet. Only 2 routing protocols – OSPF and IS-IS have ever supported the calculation of separate paths based on TOS.
- The first 3 bits in the TOS field in the IP header were being used for IP Precedence, in which values from 0 – 7 can be used to specify the transmission priority of packets at each router hop. IP Precedence does not affect the path of a packet as with TOS. IP Precedence is being phased out in favor of DSCP, but is supported by many applications and routers.
- **Differentiated Services Code Point (DSCP)** is a modification of the TOS field. 6 bits of the field are being reallocated for use as the DSCP field. DSCP is not compatible with IP Precedence.
- An application can modify the handling of IP packets by extending the IP header with IP options. IP options are rarely used for regular IP packets, as most routers are heavily optimized for forwarding IP packets without IP options. The use of IP options introduces a potential DoS vulnerability against routers due to the additional processing workload of packets with IP options.
- Most IP options (eg: the *record-route* and *timestamp* options) are used for statistics collection and do not affect the forwarding path of packets. However, the *strict-source route* and *loose-source route* options can be utilized by the originator of a packet to control the forwarding path the packet.
- **IP source routing** is often considered as a security hole, as even with security is being provided through address filtering, the final destination of a packet might be buried in the IP options field. As a result, most routers are configured to discard packets containing source routing options with the **no ip source-route** global configuration command.

## TCP Selective Acknowledgment (SACK) and Fast Retransmission



**Figure A6-6:** TCP Selection Acknowledgment (SACK)

- In the above scenario, there are total of 5 segments to be sent from 10.0.0.1 to 172.16.0.1: **1)** 1 – 1400, **2)** 1401 – 2800, **3)** 2801 – 4200, **4)** 4201 – 5600, and **5)** 5601 – 7000.
- Below describes the steps involved in the scenario:
  - i) Steps 1, 2, and 3 show the TCP three-way handshake connection establishment phase. Both end systems agree to use SACK. The agreed MSS is 1460 bytes.
  - ii) 10.10.10.1 transmits the 1st and 2nd segments to 172.16.0.1 (Steps 5 and 6). 172.16.0.1 acknowledges the segments (Step 7).
  - iii) 10.10.10.1 transmits the 4th and 5th segments to 172.16.0.1 (Step 8 and 10). The 3rd segment is lost. With **selective acknowledgment**, 172.16.0.1 selectively acknowledges the out-of-order segments, instead of cumulatively acknowledging the last in-order segment received (Step 9 and 11).
  - iv) With **fast retransmission**, which skips the retransmission of selectively acknowledged segments, 10.10.10.1 retransmits the 3rd segment to 172.16.0.1 (Step 12).
  - v) Finally, 172.16.0.1 cumulatively acknowledges the receipt of all segments (Step 13).

## Dive into Proxy ARP

- Proxy ARP allows a host with not routing capability to reach remote subnets without the default gateway configuration. The hosts assume the network they reside as a flat network in which they can reach any hosts after the ARP resolution process. Proxy ARP is defined in RFC 1027.
- Below are some of the disadvantages of using Proxy ARP:
  - i) It increases the amount of ARP traffic on the network.
  - ii) A host requires larger ARP table for handling IP-to-MAC address mappings.
  - iii) Security threat – spoofing, where a host claims to be another for intercepting packets.

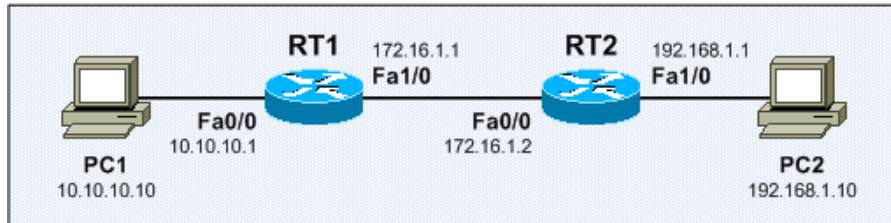


Figure A6-7: Sample Proxy ARP Network

- Below shows the ARP table on PC1 when RT1 providing Proxy ARP service and PC1 is not configured with any default gateway:

```
PC1#sh arp
Protocol  Address          Age (min)  Hardware Addr  Type   Interface
Internet  10.10.10.10      -          cc00.0f28.0000  ARPA   FastEthernet0/0
Internet  10.10.10.1       0          cc01.0f28.0000  ARPA   FastEthernet0/0
Internet  192.168.1.10    0          cc01.0f28.0000  ARPA   FastEthernet0/0
Internet  172.16.1.2      0          cc01.0f28.0000  ARPA   FastEthernet0/0
PC1#
RT1#sh ip redirects
Default gateway is not set

Host          Gateway          Last Use      Total Uses    Interface
ICMP redirect cache is empty
RT1#
```

## Local Proxy ARP

- Below shows the output of the **show ip interface {type num}** command which shows that local proxy ARP is disabled. This section discusses what Local Proxy ARP is and its usage.

```
Router#sh ip int fa1/0
FastEthernet1/0 is up, line protocol is up
Internet address is 10.10.10.1/24
Broadcast address is 255.255.255.255
Address determined by setup command
MTU is 1500 bytes
Helper address is not set
Directed broadcast forwarding is disabled
Outgoing access list is not set
Inbound access list is not set
Proxy ARP is enabled
Local Proxy ARP is disabled
--- output omitted ---
```

- Local Proxy ARP is used when there is a need to perform proxy ARP for hosts in local network. Normally network devices do not perform local proxy ARP as they would assume the host with the requested IP address would answer the ARP request itself.

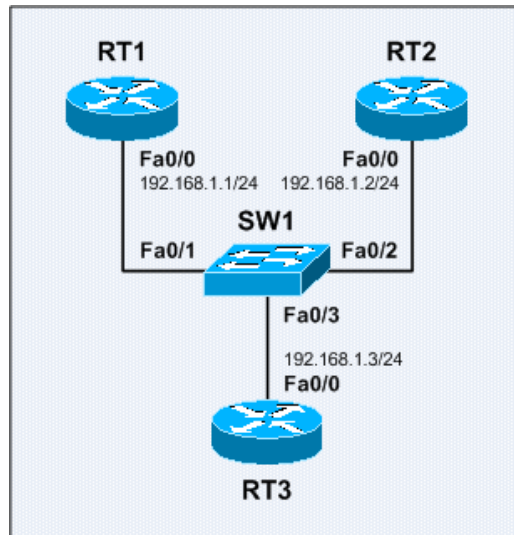


Figure A6-8: Sample Local Proxy ARP Network

- The **switchport protected** interface subcommand is configured on SW1 Fa0/1 and Fa0/2. The **ip local-proxy-arp** interface subcommand is configured on RT3 Fa0/0.
- Below shows that RT3 is performing proxy ARP for RT2:

```
RT1#ping 192.168.1.2
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 17.0.0.2, timeout is 2 seconds:
!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/3/4 ms
RT1#sh arp
Protocol  Address          Age (min)  Hardware Addr  Type   Interface
-----  -
Internet 192.168.1.1      -          1111.1111.1111 ARPA   FastEthernet0/0
Internet 192.168.1.2      6          3333.3333.3333 ARPA   FastEthernet0/0
Internet 192.168.1.3      5          3333.3333.3333 ARPA   FastEthernet0/0
```

- The **switchport protected** interface subcommand configures a switch port as a protected port. **Protected ports** have the following features:
  - A protected port does not forward any traffic (including unicast, multicast, and broadcast) to other protected ports – traffic cannot be forwarded between protected ports at Layer 2; all traffic between protected ports must be forwarded through a Layer 3 device.
  - Traffic between a protected port and a non-protected port is forwarded as usual.**Note:** This applies to a single switch, and cannot be extended across multiple switches.
- Protected ports are being used in environments where traffic must not be forwarded between ports on the same switch so that an end system does not see the traffic from another end system. Below are some other sample usages of protected ports:
  - A hotel environment where the ports for each room should not be able to communicate with each other, but they need to communicate with the gateway.
  - A DMZ zone where the ports for each server should not be able to communicate with each other in order to prevent further damages of other servers when a server is owned.
  - A collocation environment where servers from different customers should not be able to communicate with each other.

## Determining Cisco Router Memory Size

- The **show version** EXEC command is able to tell how much **Dynamic RAM** (DRAM) and **packet memory** (separate SRAM or shared memory) are installed in a Cisco router.
- The Cisco 4000, 4500, 4700, and 7500 series routers have separate DRAM and packet memory. The example below shows a router with **64M** of DRAM and **2M** of SRAM (packet memory):  
cisco RSP4 (R5000) processor with **65536K/2072K** bytes of memory  
The example below shows a router with **256M** of DRAM and **8M** of SRAM (packet memory):  
cisco RSP8 (R7000) processor with **262144K/8216K** bytes of memory.
- The Cisco 1600, 2500, 2600, 3600, and 7200 series routers use a fraction of their DRAM as packet memory. Hence both numbers need to be added to find out the real amount of DRAM. The example below shows a router with  $29696K + 3072K = 32768K = \mathbf{32M}$  of DRAM.  
cisco 2611 (MPC860) processor (revision x) with **29696K/3072K** bytes of memory  
The example below shows a router with  $93184K + 5120K = 98304K = \mathbf{98M}$  of DRAM.  
cisco 2621XM (MPC860P) processor (revision) with **93184K/5120K** bytes of memory
- DRAM is mainly used for system processing, eg: storing routing tables, routing protocols data, network accounting information, and running the Cisco IOS software; while packet memory is used for packet buffering of the router network interfaces and CPU cache memory functions. The packet memory on Cisco 4000, 4500, 4700, and 7500 series routers is the total physical I/O memory (also known as **SRAM** or **Fast Memory**); while the packet memory on Cisco 1600, 2500, 2600, 3600, and 7200 series routers is the amount of **shared memory**, which is a portion of the DRAM.
- The terms I/O memory (iomem), shared memory, Fast Memory, and PCI memory are all refer to **Packet Memory**, which is either a separate physical RAM stick or module, or shared DRAM.
- **Static RAM** (SRAM) is a type of semiconductor memory that does not need to be periodically refreshed as **Dynamic RAM** (DRAM). Memory refresh is the process of periodically reading information from an area of computer memory, and rewriting the read information back to the same area immediately without any modification. SRAM is still volatile, as the data stored in it is eventually lost when the memory is not powered.
- SRAM is more expensive, but faster and far less power hungry when idle (no periodical refresh) compared to DRAM. Additionally, due to a more complex internal structure, SRAM is less dense than DRAM and hence not used for high-capacity, low-cost applications, eg: PC main memory.

## The Myth of the Running Configuration file

- The **reload** privileged command is the only way to restore the configuration of a device into its last reboot state. Replacing the running-config with the startup-config or a configuration file from a TFTP server will be **unsuccessful** – configuration is **merged** instead of being overwritten.
- The Configuration Replace and Configuration Rollback feature which introduced in Cisco IOS Release 12.3(7)T and 12.2(25)S provides the capability to replace the current running configuration with any configuration file. The **configure replace** privileged command can be used to overwrite the running configuration.

## Spanning Tree Protocol Port ID

- When 2 switches are interconnected with multiple cross cables, the root port of the non-root bridge switch (SW2) is the port that connects to the **lowest Port ID** of the root bridge (SW1).

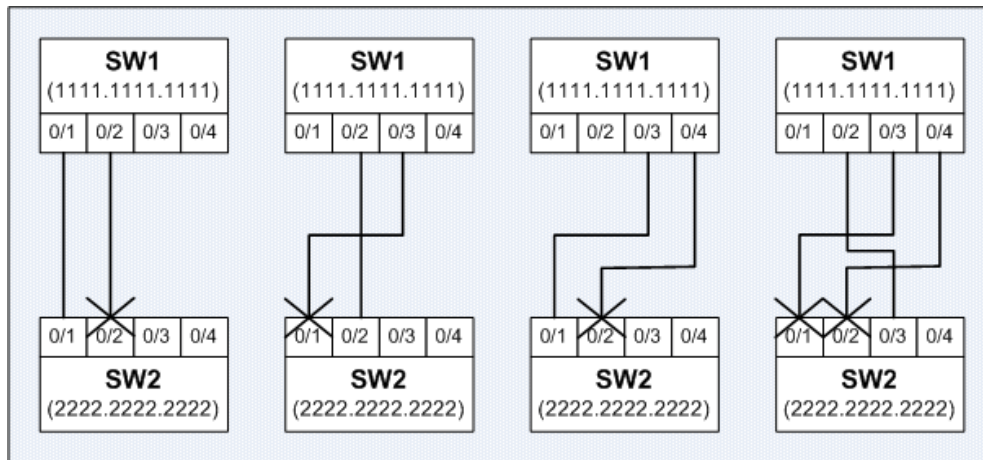


Figure A6-8: Spanning Tree Protocol Port ID

## Router-on-a-Stick VLAN Trunking Encapsulation and IP Address Configuration

- Below shows the configuration sequence of VLAN trunking encapsulation and IP address on a router subinterface in Router-on-a-Stick setup. The VLAN trunking encapsulation must be configured prior to configuring an IP address.

```

RT1#conf t
Enter configuration commands, one per line. End with CNTL/Z.
RT1(config)#int fa0/0.1
RT1(config-subif)#ip add 192.168.1.1 255.255.255.0

% Configuring IP routing on a LAN subinterface is only allowed if that
subinterface is already configured as part of an IEEE 802.10, IEEE 802.1Q,
or ISL VLAN.

RT1(config-subif)#
RT1(config-subif)#encapsulation ?
  dot1Q  IEEE 802.1Q Virtual LAN
  isl    Inter Switch Link - Virtual LAN encapsulation
  sde    IEEE 802.10 Virtual LAN - Secure Data Exchange

RT1(config-subif)#encapsulation dot1Q 1
RT1(config-subif)#ip add 192.168.1.1 255.255.255.0
RT1(config-subif)#^Z
RT1#
RT1#sh int fa0/0.1
FastEthernet0/0.1 is up, line protocol is up
Hardware is AmdFE, address is cc01.0520.0000 (bia cc01.0520.0000)
Internet address is 192.168.1.1/24
MTU 1500 bytes, BW 100000 Kbit, DLY 100 usec,
  reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation 802.1Q Virtual LAN, Vlan ID 1.
ARP type: ARPA, ARP Timeout 04:00:00
RT1#
  
```

## switchport trunk encapsulation and switchport mode trunk Configuration

- Below shows the configuration sequence of VLAN trunking on a Fast Ethernet interface. The VLAN trunking encapsulation must be configured (with the **switchport trunk encapsulation** interface subcommand) prior to configuring a trunking interface with the **switchport mode trunk** interface subcommand.

```
Switch#sh run int fa0/1
Building configuration...

Current configuration : 68 bytes
!
interface FastEthernet0/1
 switchport mode dynamic desirable
end

Switch#
Switch#sh int fa0/1 trunk

Port          Mode          Encapsulation  Status        Native vlan
Fa0/12        desirable     negotiate       other         1

Port          Vlans allowed on trunk
Fa0/12        none

Port          Vlans allowed and active in management domain
Fa0/12        none

Port          Vlans in spanning tree forwarding state and not pruned
Fa0/12        none
Switch#
Switch#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Switch(config)#int fa0/1
Switch(config-if)#switchport mode trunk
Command rejected: An interface whose trunk encapsulation is "Auto" can not
be configured to "trunk" mode.
Switch(config-if)#
Switch(config-if)#switchport trunk encapsulation ?
 dot1q      Interface uses only 802.1q trunking encapsulation when trunking
 isl        Interface uses only ISL trunking encapsulation when trunking
 negotiate  Device will negotiate trunking encapsulation with peer on
           interface

Switch(config-if)#switchport trunk encapsulation dot1q
Switch(config-if)#switchport mode trunk
Switch(config-if)#^Z
Switch#
```

## The Problem of Router-on-a-Stick Configuration on Ethernet Interface

- RT1 with router-on-a-stick configuration on an Ethernet interface was unable to communicate with PC1 resides in **Native VLAN** due to confusion of using the main interface and subinterface. **Note:** This problem does not happen on Fast Ethernet interfaces.
- Below shows the configuration on RT1 which experienced the above describe problem scenario:

```
!  
interface Ethernet0/0  
  no ip address  
  full-duplex  
!  
interface Ethernet0/0.1  
  encapsulation dot1Q 1 native  
  ip address 192.168.1.1 255.255.255.0  
!  
interface Ethernet0/0.2  
  encapsulation dot1Q 2  
  ip address 192.168.2.1 255.255.255.0  
!  
interface Ethernet0/0.3  
  encapsulation dot1Q 3  
  ip address 192.168.3.1 255.255.255.0  
!
```

- Below shows the root cause of the problem – RT1 unable to communicate with PC1:

```
RT1#debug arp  
ARP packet debugging is on  
RT1#  
RT1#ping 192.168.1.2 rep 1  
  
Type escape sequence to abort.  
Sending 1, 100-byte ICMP Echos to 192.168.1.2, timeout is 2 seconds:  
.  
Success rate is 0 percent (0/1)  
RT1#  
RT1#sh arp  
Protocol  Address      Age (min)  Hardware Addr  Type   Interface  
Internet  192.168.1.1    -          cc01.01c4.0000  ARPA   Ethernet0/0.1  
Internet  192.168.1.2    0          Incomplete     ARPA  
RT1#  
RT1#sh log | in ARP  
00:07:31: IP ARP: sent req src 192.168.1.1 cc01.01c4.0000,  
00:07:31: IP ARP rep filtered src 192.168.1.2 cc02.01c4.0000, dst  
192.168.1.1 cc01.01c4.0000 wrong cable, interface Ethernet0/0  
RT1#
```

- Below shows the output of the **show arp** command after implemented a workaround – configure the native VLAN IP address on the main interface instead of the subinterface:

```
RT1#sh arp  
Protocol  Address      Age (min)  Hardware Addr  Type   Interface  
Internet  192.168.1.1    -          cc01.01c4.0000  ARPA   Ethernet0/0  
Internet  192.168.1.2    0          cc02.01c4.0000  ARPA   Ethernet0/0
```

## Switching Technologies

- The load balancing method chosen for packet forwarding depends on the type of switching performed by a router. Cisco IOS supports the following 2 common switching methods:

<b>Process Switching</b>	Performs <b>packet-by-packet</b> load balancing. Utilizes CPU resource for looking up the next hop in the routing table for every packet that needs to be forwarded.
<b>Fast Switching</b>	Performs <b>destination-by-destination</b> or <b>session-by-session</b> load balancing. Uses a route cache for determining the destination IP address and the next hop IP address, which is <b>less CPU intensive</b> than process switching.

- The first packet to any host is always process switched. If fast switching is enabled on the outbound interface, the router would create a **cache entry** for the destination host after it forwarded the first packet. The cache entry is then used to forward all subsequent packets destined to the same host. Route cache lookup is much faster than routing table lookup.
- Cisco IOS offers **other technologies and methods of switching** IP packets depend on the hardware platform and IOS version.

## Dissecting Subnet Zero

- Traditionally, it was strongly recommended not to use subnet zero (all-0s) and broadcast subnet for addressing (RFC 950 – Internet Standard Subnetting Procedure). This is to avoid the confusion of having a network and a subnet with indistinguishable addresses.
- Below shows a Class B network – 172.16.0.0/16 is subnetted with /18 subnet mask:

Subnet Address	Subnet Mask	Broadcast Address	Valid Host Range
172.16.0.0	255.255.192.0	172.16.63.255	172.16.0.1 – 172.16.63.254
172.16.64.0	255.255.192.0	172.16.127.255	172.16.64.1 – 172.16.127.254
172.16.128.0	255.255.192.0	172.16.191.255	172.16.128.1 – 172.16.191.254
172.16.192.0	255.255.192.0	172.16.255.255	172.16.192.1 – 172.16.255.254

- As shown in the example above, 172.16.16.16 is resides in the 172.16.0.0 subnet (subnet zero). This subnet address is same with the network address – a network and subnet with indistinguishable addresses.
- **RFC 1878 – Variable Length Subnet Table For IPv4** states that the practice of excluding all-0s and all-1s subnets is **obsolete**.
- **MISC Note:** Broadcast address is used in both logical (L3) and hardware (L2) addressing. With logical addressing, the host address will be all 1s. With hardware addressing, the hardware address will be all 1s in binary (all Fs in hexadecimal).

- Sample Subnet Zero configuration:

```

Router(config)#no ip subnet-zero
Router(config)#int fa0/0
Router(config-if)#ip address 10.0.0.1 255.255.255.0      (1)
Bad mask /24 for address 10.0.0.1
Router(config-if)#ip address 10.1.0.1 255.255.255.0      (2)
Router(config-if)#ip address 172.16.0.1 255.255.255.0    (3)
Bad mask /24 for address 172.16.0.1
Router(config-if)#ip address 172.16.1.1 255.255.255.0    (4)
Router(config-if)#ip address 192.168.0.1 255.255.255.192 (5)
Bad mask /26 for address 192.168.0.1
Router(config-if)#ip address 192.168.0.65 255.255.255.192 (6)
Router(config-if)#exit
Router(config)#
Router(config)#ip subnet-zero
Router(config)#int fa0/0
Router(config-if)#ip address 10.0.0.1 255.255.255.0
Router(config-if)#ip address 172.16.0.1 255.255.255.0
Router(config-if)#ip address 192.168.0.1 255.255.255.192
Router(config-if)#

```

- Subnet Zero explanation:

<b>1</b>	10.0.0.1/24. Class A. Default subnet mask is /8.				
	10	0	0	1	
	00000000	<b>00000000</b>	<b>00000000</b>	00000001	Subnet Zero
<b>2</b>	10.1.0.1/24. Class A. Default subnet mask is /8.				
	10	1	0	1	
	00000000	<b>00000001</b>	<b>00000000</b>	00000001	Not Subnet Zero
<b>3</b>	172.16.0.1/24. Class B. Default subnet mask is /16.				
	172	16	0	1	
	10101100	00010000	<b>00000000</b>	00000001	Subnet Zero
<b>4</b>	172.16.1.1/24. Class B. Default subnet mask is /16.				
	172	16	1	1	
	10101100	00010000	<b>00000001</b>	00000001	Not Subnet Zero
<b>5</b>	192.168.0.1/26. Class C. Default subnet mask is /24.				
	192	168	0	1	
	11000000	10101000	00000000	<b>00</b> 000001	Subnet Zero
<b>6</b>	192.168.0.65/26. Class C. Default subnet mask is /24.				
	192	168	0	65	
	11000000	10101000	00000000	<b>01</b> 000001	Not Subnet Zero

- **Note:** The **no ip subnet-zero** global configuration command only affects the subnet zero; it does not affect the broadcast (all-1s) subnet.

## Default Route (Gateway of Last Resort) Configurations

- The **ip default-gateway** *{ip-addr}* command should only be used when **ip routing is disabled** on the Cisco device (eg: switch); or used to reach the TFTP server during IOS upgrade, as a limited-function IOS (RXBOOT) with **no routing capability** will be loaded.
- The **ip default-network** *{classful-net-addr}* and **ip route 0.0.0.0 0.0.0.0** *{classful-net-addr | ip-addr}* commands can be used when **ip routing is enabled** on the Cisco device. The major difference between these 2 commands is that configuring a static default route only defines a default route for a single router, while **ip default-network** *{classful-net-addr}* will propagate the default route to other routers via a routing protocol.

## The passive-interface Router Subcommand

- The **passive-interface** router subcommand on most routing protocols (eg: RIP) would restrict outgoing routing updates only. The router would still receive and process incoming routing updates from other routers which reside on the passive interface subnet of the router.
- In EIGRP, the **passive-interface** *{type num}* router subcommand suppresses the exchange of Hello packets between 2 routers, resulting in their neighbor relationship will never be formed and hence will never exchange routing updates. This suppresses not only outgoing routing updates, but also incoming routing updates.

## The permanent keyword in the Static Route Configuration

- Without the **permanent** keyword in a static route statement, an inactive interface will cause the directly connected network and all the associated static routes removed from the routing table.
- Adding the **permanent** keyword to a static route statement will keep the static route remains in the routing table no matter what happens, even if the interface goes down (or shutdown) or the directly connected network is removed from the routing table.
- The advantage of this option is that static routes do not need to be processed for insertion into / removal from the routing table upon interface status change, hence saving processing resources. The processing time for static routing insertion into / removal from the routing table is 1 second. Prior to Cisco IOS Release 12.0, this processing time was 5 seconds.
- **Note:** Configuring static routes with the **permanent** keyword could make a subnet that doesn't even exist to be shown in the routing table!
- Below shows the behavior of the routing table before and after using the **permanent** keyword:

```
Router#sh run | in ip route
ip route 172.16.1.0 255.255.255.0 10.10.10.2
Router#
Router#sh ip route

Gateway of last resort is not set

      172.16.0.0/24 is subnetted, 1 subnets
S       172.16.1.0 [1/0] via 10.10.10.2
      10.0.0.0/30 is subnetted, 1 subnets
C       10.10.10.0 is directly connected, Serial1/0
Router#
Router#debug ip routing
IP routing debugging is on
Router#
00:03:34: is_up: 1 state: 4 sub state: 1 line: 0
00:03:42: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial1/0, changed
state to down
00:03:42: is_up: 0 state: 4 sub state: 1 line: 0
00:03:42: RT: interface Serial1/0 removed from routing table
00:03:42: RT: del 10.10.10.0/30 via 0.0.0.0, connected metric [0/0]
00:03:42: RT: delete subnet route to 10.10.10.0/30
00:03:42: RT: delete network route to 10.0.0.0
00:03:43: RT: del 172.16.1.0/24 via 10.10.10.2, static metric [1/0]
00:03:43: RT: delete subnet route to 172.16.1.0/24
00:03:43: RT: delete network route to 172.16.0.0
Router#
Router#sh ip route

Gateway of last resort is not set

Router#
```

```

Router#conf t
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#no ip route 172.16.1.0 255.255.255.0
Router(config)#ip route 172.16.1.0 255.255.255.0 10.10.10.2 permanent
Router(config)#^Z
Router#
00:05:22: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial1/0, changed
state to up
00:05:22: is_up: 1 state: 4 sub state: 1 line: 0
00:05:22: RT: add 10.10.10.0/30 via 0.0.0.0, connected metric [0/0]
00:05:22: RT: interface Serial1/0 added to routing table
00:05:23: RT: add 172.16.1.0/24 via 10.10.10.2, static metric [1/0]
Router#
Router#sh ip route

Gateway of last resort is not set

      172.16.0.0/24 is subnetted, 1 subnets
S       172.16.1.0 [1/0] via 10.10.10.2
      10.0.0.0/30 is subnetted, 1 subnets
C       10.10.10.0 is directly connected, Serial1/0
Router#
Router#
00:06:04: is_up: 1 state: 4 sub state: 1 line: 0
00:06:12: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial1/0, changed
state to down
00:06:12: is_up: 0 state: 4 sub state: 1 line: 0
00:06:12: RT: interface Serial1/0 removed from routing table
00:06:12: RT: del 10.10.10.0/30 via 0.0.0.0, connected metric [0/0]
00:06:12: RT: delete subnet route to 10.10.10.0/30
00:06:12: RT: delete network route to 10.0.0.0
Router#
Router#sh ip route

Gateway of last resort is not set

      172.16.0.0/24 is subnetted, 1 subnets
S       172.16.1.0 [1/0] via 10.10.10.2
Router#
Router#conf t
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#int s1/0
Router(config-if)#shut
Router(config-if)#^Z
Router#
00:06:35: is_up: 0 state: 6 sub state: 1 line: 0
00:06:37: %LINK-5-CHANGED: Interface Serial1/0, changed state to
administratively down
00:06:37: is_up: 0 state: 6 sub state: 1 line: 0
Router#
Router#sh ip route

Gateway of last resort is not set

      172.16.0.0/24 is subnetted, 1 subnets
S       172.16.1.0 [1/0] via 10.10.10.2
Router#

```

## RIPv1 and VLSM

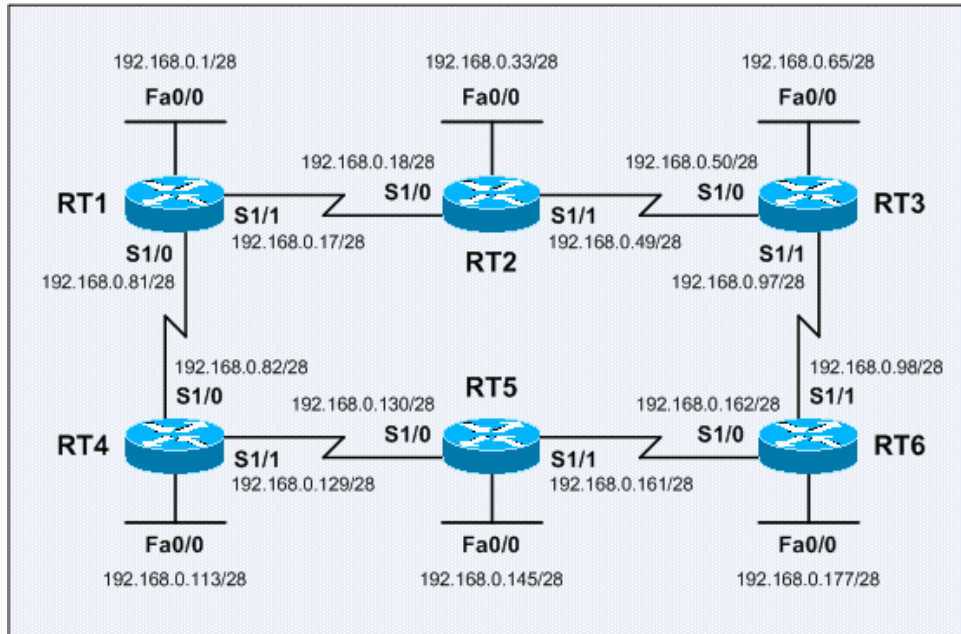


Figure A6-9: Sample RIPv1 and VLSM Network

- The network above is subnetted using a Class C address block. In the network above, there are a total of 12 networks (6 LANs and 6 point-to-point WANs). The 255.255.255.240 (/28) subnet mask is used to support a maximum of 16 networks with 14 usable IP addresses on each LAN.
- RIPv1 does not support VLSM information, so all networks must have the same subnet mask.
- This sample setup shows that even RIP does not support VLSM, such network can be setup when all subnets are using the same subnet mask.
- Below shows the routing table on RT1:

```

RT1#sh ip route
      192.168.0.0/28 is subnetted, 12 subnets
C       192.168.0.0 is directly connected, FastEthernet0/0
C       192.168.0.16 is directly connected, Serial1/1
R       192.168.0.32 [120/1] via 192.168.0.18, 00:00:01, Serial1/1
R       192.168.0.48 [120/1] via 192.168.0.18, 00:00:01, Serial1/1
R       192.168.0.64 [120/2] via 192.168.0.18, 00:00:01, Serial1/1
C       192.168.0.80 is directly connected, Serial1/0
R       192.168.0.96 [120/2] via 192.168.0.18, 00:00:01, Serial1/1
R       192.168.0.112 [120/1] via 192.168.0.82, 00:00:12, Serial1/0
R       192.168.0.128 [120/1] via 192.168.0.82, 00:00:12, Serial1/0
R       192.168.0.144 [120/2] via 192.168.0.82, 00:00:13, Serial1/0
R       192.168.0.160 [120/2] via 192.168.0.82, 00:00:12, Serial1/0
R       192.168.0.176 [120/3] via 192.168.0.18, 00:00:01, Serial1/1
                               [120/3] via 192.168.0.82, 00:00:12, Serial1/0
RT1#
  
```

## Fun with NATs and ACLs (Firewalls)

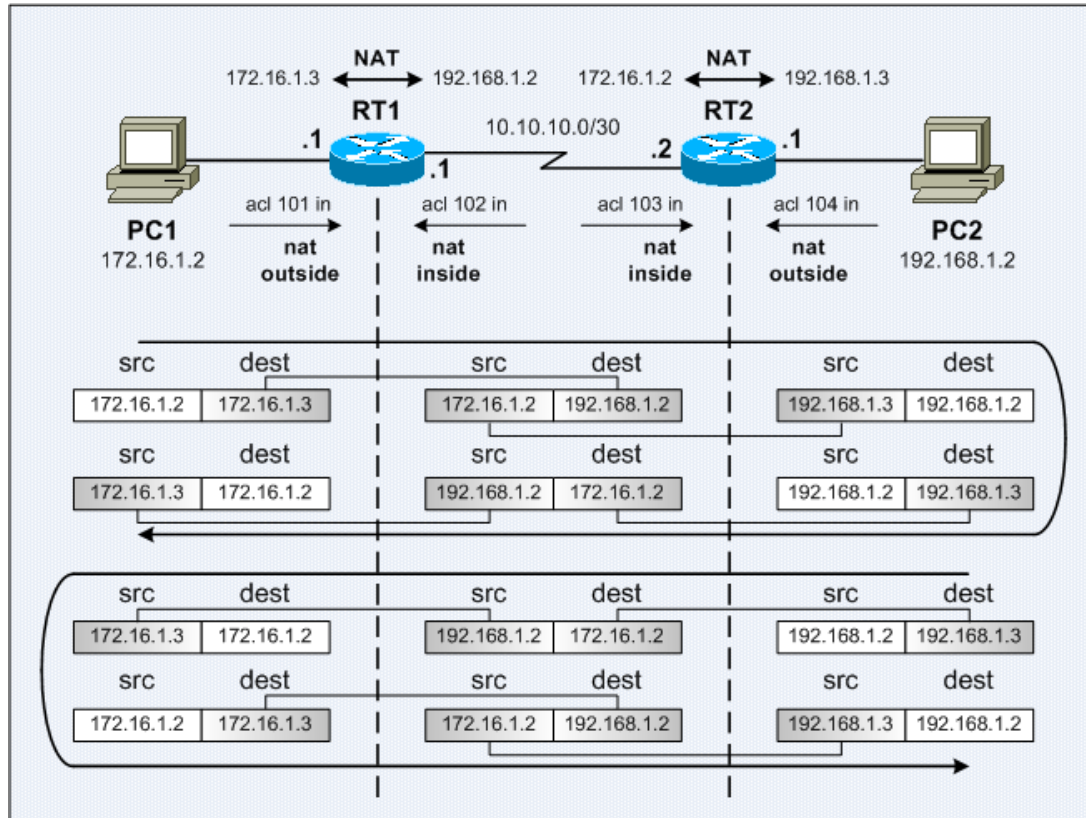


Figure A6-10: Sample NATs and ACLs Network

- NAT configuration on RT1 for PC1 to remote access 192.168.1.2 via NAT IP 172.16.1.3:  
RT1 (config) # **ip nat inside source static 192.168.1.2 172.16.1.3**
- NAT configuration on RT2 for PC2 to remote access 172.16.1.2 via NAT IP 192.168.1.3:  
RT2 (config) # **ip nat inside source static tcp 172.16.1.2 23 192.168.1.3 23**
- Extended IP Access Lists configuration on RT1 and RT2 for PC1 to remote access (Telnet) PC2:

PC1 to PC2:

```
RT1 (config) # access-list 101 permit tcp host 172.16.1.2 host 172.16.1.3 eq 23
RT2 (config) # access-list 103 permit tcp host 172.16.1.2 host 192.168.1.2 eq 23
```

PC2 back to PC1:

```
RT2 (config) # access-list 104 permit tcp host 192.168.1.2 eq 23 host 192.168.1.3
RT1 (config) # access-list 102 permit tcp host 192.168.1.2 eq 23 host 172.16.1.2
```

- Extended IP Access Lists configuration on RT1 and RT2 for PC2 to remote access (Telnet) PC1:

PC2 to PC1:

```
RT2 (config) # access-list 104 permit tcp host 192.168.1.2 host 192.168.1.3 eq 23
RT1 (config) # access-list 102 permit tcp host 192.168.1.2 host 172.16.1.2 eq 23
```

PC1 back to PC2:

```
RT1 (config) # access-list 101 permit tcp host 172.16.1.2 eq 23 host 172.16.1.3
RT2 (config) # access-list 103 permit tcp host 172.16.1.2 eq 23 host 192.168.1.2
```

- Below shows the NAT operations on RT1 and RT2 and ACL hit counts when PC1 accesses PC2:

```

RT1#debug ip nat
IP NAT debugging is on
RT1#
00:02:30: NAT: s=172.16.1.2, d=172.16.1.3->192.168.1.2 [14748]
00:02:32: NAT: s=172.16.1.2, d=172.16.1.3->192.168.1.2 [14748]
00:02:36: NAT: s=172.16.1.2, d=172.16.1.3->192.168.1.2 [14748]
00:02:38: NAT*: s=192.168.1.2->172.16.1.3, d=172.16.1.2 [19343]
00:02:38: NAT*: s=172.16.1.2, d=172.16.1.3->192.168.1.2 [14749]
00:02:38: NAT*: s=172.16.1.2, d=172.16.1.3->192.168.1.2 [14750]
--- output omitted ---
RT1#
RT1#sh access-list
Extended IP access list 101
  10 permit tcp host 172.16.1.2 host 172.16.1.3 eq telnet (28 matches)
  20 permit tcp host 172.16.1.2 eq telnet host 172.16.1.3
Extended IP access list 102
  10 permit tcp host 192.168.1.2 eq telnet host 172.16.1.2 (14 matches)
  20 permit tcp host 192.168.1.2 host 172.16.1.2 eq telnet
RT1#
RT1#sh ip nat statistics
Total active translations: 1 (1 static, 0 dynamic; 0 extended)
Outside interfaces:
  FastEthernet0/0
Inside interfaces:
  FastEthernet1/0
Hits: 36 Misses: 0
Expired translations: 0
Dynamic mappings:
RT1#
-----
RT2#
00:02:31: NAT: s=172.16.1.2->192.168.1.3, d=192.168.1.2 [14748]
00:02:37: NAT*: s=192.168.1.2, d=192.168.1.3->172.16.1.2 [19343]
00:02:37: NAT*: s=172.16.1.2->192.168.1.3, d=192.168.1.2 [14749]
--- output omitted ---
RT2#sh access-list
Extended IP access list 103
  10 permit tcp host 172.16.1.2 host 192.168.1.2 eq telnet (25 matches)
  20 permit tcp host 172.16.1.2 eq telnet host 192.168.1.2
Extended IP access list 104
  10 permit tcp host 192.168.1.2 eq telnet host 192.168.1.3 (14 matches)
  20 permit tcp host 192.168.1.2 host 192.168.1.3 eq telnet
RT2#
-----
PC1#telnet 172.16.1.3
Trying 172.16.1.3 ... Open

User Access Verification

Password:
PC2>who
      Line          User           Host(s)        Idle           Location
  *  0 con 0
  *  66 vty 0
      Interface     User           Mode           Idle           Peer Address
PC2>

```

- Below shows the NAT operations on RT2 and RT1 and ACL hit counts when PC2 accesses PC1:

```

RT2#debug ip nat
IP NAT debugging is on
RT2#
00:04:42: NAT: s=192.168.1.2, d=192.168.1.3->172.16.1.2 [50878]
00:04:42: NAT*: s=172.16.1.2->192.168.1.3, d=192.168.1.2 [2634]
00:04:42: NAT*: s=192.168.1.2, d=192.168.1.3->172.16.1.2 [50879]
00:04:42: NAT*: s=192.168.1.2, d=192.168.1.3->172.16.1.2 [50880]
--- output omitted ---
RT2#
RT2#sh access-list
Extended IP access list 103
 10 permit tcp host 172.16.1.2 host 192.168.1.2 eq telnet
 20 permit tcp host 172.16.1.2 eq telnet host 192.168.1.2 (17 matches)
Extended IP access list 104
 10 permit tcp host 192.168.1.2 eq telnet host 192.168.1.3
 20 permit tcp host 192.168.1.2 host 192.168.1.3 eq telnet (25 matches)
RT2#
RT2#sh ip nat statistics
Total active translations: 1 (1 static, 0 dynamic; 0 extended)
Outside interfaces:
  FastEthernet0/0
Inside interfaces:
  FastEthernet1/0
Hits: 40 Misses: 0
Expired translations: 0
Dynamic mappings:
RT2#
-----
RT1#
00:04:43: NAT: s=192.168.1.2->172.16.1.3, d=172.16.1.2 [50878]
00:04:43: NAT*: s=172.16.1.2, d=172.16.1.3->192.168.1.2 [2634]
00:04:43: NAT*: s=192.168.1.2->172.16.1.3, d=172.16.1.2 [50879]
00:04:43: NAT*: s=172.16.1.2, d=172.16.1.3->192.168.1.2 [2635]
--- output omitted ---
RT1#sh access-list
Extended IP access list 101
 10 permit tcp host 172.16.1.2 host 172.16.1.3 eq telnet
 20 permit tcp host 172.16.1.2 eq telnet host 172.16.1.3 (17 matches)
Extended IP access list 102
 10 permit tcp host 192.168.1.2 eq telnet host 172.16.1.2
 20 permit tcp host 192.168.1.2 host 172.16.1.2 eq telnet (25 matches)
RT1#
-----
PC2#telnet 192.168.1.3
Trying 192.168.1.3 ... Open

User Access Verification

Password:
PC1>
PC1>who
   Line          User           Host(s)        Idle           Location
   * 0 con 0      idle           idle           00:00:35
   * 66 vty 0    idle           idle           00:00:00 172.16.1.3

   Interface    User           Mode           Idle           Peer Address
PC1>

```

## NAT with Virtual Interface

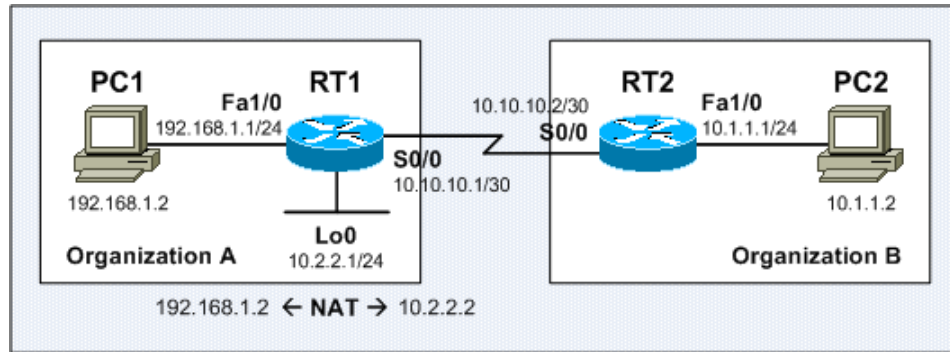


Figure A6-11: NAT with Virtual Interface

- In this scenario, PC1 and PC2 are belong to 2 different organizations and must be assigned with IP addresses in 192.168.1.0/24 and 10.1.1.0/24 subnets respectively due to some reasons, eg: network overlapping and not feasible to have subnet other than the mentioned subnets.
- Organization A has no issue introducing routing to 10.1.1.0/24 into its network; but Organization B unable to introduce routing to 192.168.1.0/24 due to its network design and IP addressing plan.
- The solution is assign 10.2.2.1/24 to RT1 Lo0, which is supposed to be assigned to RT1 Fa1/0; followed by translating 192.168.1.2 to 10.2.2.2 – the IP address which Organization B expects.
- NAT configuration on RT1:

```
!  
interface Loopback0  
 ip address 10.2.2.1 255.255.255.0  
 ip nat outside  
!  
interface Serial0/0  
 ip address 10.10.10.1 255.255.255.0  
 ip nat outside  
!  
interface FastEthernet1/0  
 ip address 192.168.1.1 255.255.255.0  
 ip nat inside  
!  
router eigrp 100  
 network 10.0.0.0  
 no auto-summary  
!  
ip nat inside source static 192.168.1.2 10.2.2.2  
!
```

- The NAT debug messages on RT1 shows the translation between 192.168.1.2 and 10.2.2.2:

```
RT1#debug ip nat  
IP NAT debugging is on  
RT1#  
00:07:18: NAT*: s=192.168.1.2->10.2.2.2, d=10.1.1.2 [10]  
00:07:18: NAT*: s=10.1.1.2, d=10.2.2.2->192.168.1.2 [10]  
00:07:18: NAT*: s=192.168.1.2->10.2.2.2, d=10.1.1.2 [11]  
00:07:18: NAT*: s=10.1.1.2, d=10.2.2.2->192.168.1.2 [11]
```

- RT2 routing table showing that 192.168.1.0/24 network is not introduced into Organization B:

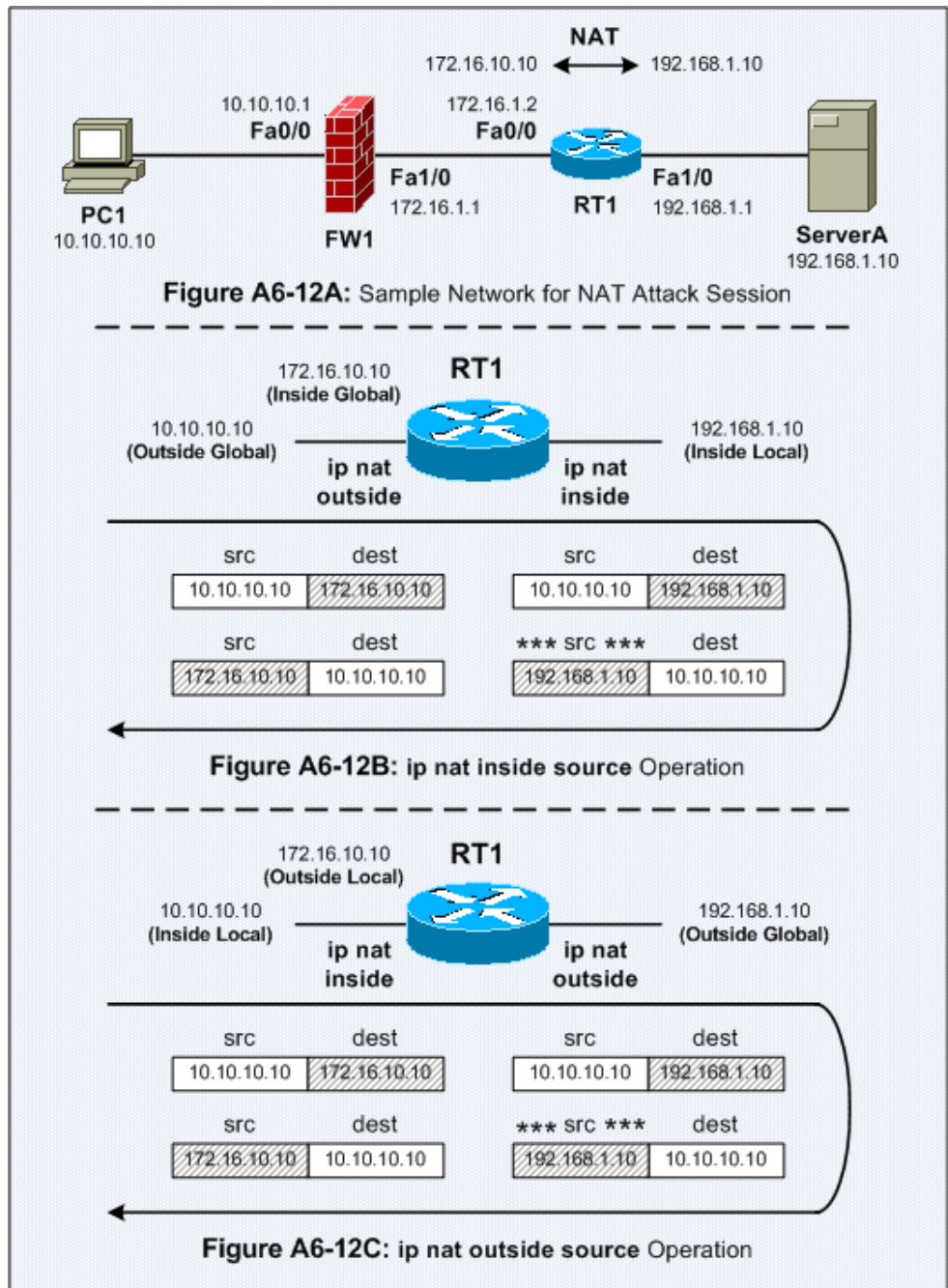
```

RT2#sh ip route
Gateway of last resort is not set

10.0.0.0/24 is subnetted, 3 subnets
D 10.2.2.0 [90/2297856] via 10.10.10.1, 00:10:46, Serial0/0
C 10.10.10.0 is directly connected, Serial0/0
C 10.1.1.0 is directly connected, FastEthernet1/0
RT2#

```

**NAT Attack Session – ip nat inside source and ip nat outside source**



**Figure A6-12: NAT Attack Session**

- This section shows how to achieve the same result – ServerA communicates with PC1 using 172.16.10.10 NAT IP instead of 192.168.1.10 real IP with **inside source** and **outside source** NAT configurations. This is a knowledge attack session which strengthens the understanding of NAT terminologies and operations.
- The **\*\*\*** indicates where the NAT operations are initiated according to the **ip nat** commands.
- **Note:** A router does not require a physical nor logical interface to reside in the NAT IP address subnet for the operation of NAT – 172.16.10.0/24 in this case. As compared to the previous section – NAT with Virtual Interface, which the router has a logical loopback interface and advertising the NAT IP subnet (10.2.2.0/24) using EIGRP.
- RT1 configuration for **ip nat inside source** operation:

```

!
interface FastEthernet0/0
 ip address 172.16.1.2 255.255.255.0
 ip nat outside
!
interface FastEthernet1/0
 ip address 192.168.1.1 255.255.255.0
 ip nat inside
!
ip nat inside source static 192.168.1.10 172.16.10.10
!

```

- Below shows the NAT debug messages on RT1 for the configuration above:

```

RT1#debug ip nat
IP NAT debugging is on
RT1#
00:04:29: NAT*: s=10.10.10.10, d=172.16.10.10->192.168.1.10 [15]
00:04:29: NAT*: s=192.168.1.10->172.16.10.10, d=10.10.10.10 [15]
00:04:29: NAT*: s=10.10.10.10, d=172.16.10.10->192.168.1.10 [16]
00:04:29: NAT*: s=192.168.1.10->172.16.10.10, d=10.10.10.10 [16]
00:04:29: NAT*: s=10.10.10.10, d=172.16.10.10->192.168.1.10 [17]
00:04:29: NAT*: s=192.168.1.10->172.16.10.10, d=10.10.10.10 [17]

```

- RT1 configuration for **ip nat outside source** operation:

```

!
interface FastEthernet0/0
 ip address 172.16.1.2 255.255.255.0
 ip nat inside
!
interface FastEthernet1/0
 ip address 192.168.1.1 255.255.255.0
 ip nat outside
!
ip nat outside source static 192.168.1.10 172.16.10.10
ip route 172.16.10.10 255.255.255.255 FastEthernet1/0
!

```

## Complex ACL

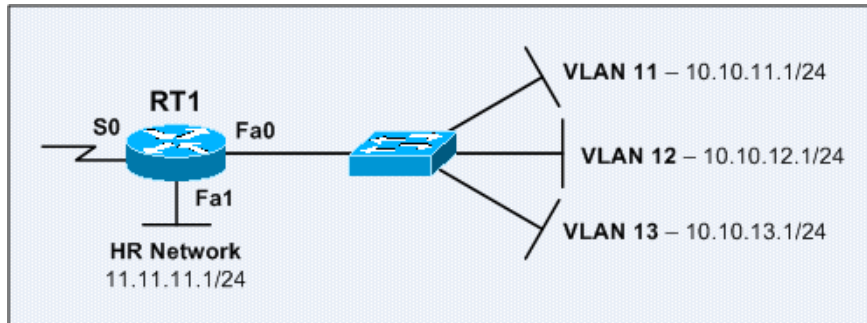


Figure A6-13: Sample Complex ACL Network

- The user requirement for the sample network above is to deny all access (Microsoft file sharing, Remote Desktop, ICMP ping, etc) from PCs in VLAN 11, 12, and 13 to the PCs in HR Network.
- Below shows a sample solution by applying an inbound access list to RT1 Fa1 interface:

```
access-list 101 deny tcp 11.11.11.0 0.0.0.255 eq 135 10.10.11.0 0.0.0.255
access-list 101 deny tcp 11.11.11.0 0.0.0.255 eq 139 10.10.11.0 0.0.0.255
access-list 101 deny tcp 11.11.11.0 0.0.0.255 eq 445 10.10.11.0 0.0.0.255
access-list 101 deny udp 11.11.11.0 0.0.0.255 eq 137 10.10.11.0 0.0.0.255
access-list 101 deny udp 11.11.11.0 0.0.0.255 eq 138 10.10.11.0 0.0.0.255
access-list 101 deny tcp 11.11.11.0 0.0.0.255 eq 3389 10.10.11.0 0.0.0.255
access-list 101 deny icmp 11.11.11.0 0.0.0.255 10.10.11.0 0.0.0.255 echo-reply
! -----
access-list 101 deny tcp 11.11.11.0 0.0.0.255 eq 135 10.10.12.0 0.0.0.255
access-list 101 deny tcp 11.11.11.0 0.0.0.255 eq 139 10.10.12.0 0.0.0.255
access-list 101 deny tcp 11.11.11.0 0.0.0.255 eq 445 10.10.12.0 0.0.0.255
access-list 101 deny udp 11.11.11.0 0.0.0.255 eq 137 10.10.12.0 0.0.0.255
access-list 101 deny udp 11.11.11.0 0.0.0.255 eq 138 10.10.12.0 0.0.0.255
access-list 101 deny tcp 11.11.11.0 0.0.0.255 eq 3389 10.10.12.0 0.0.0.255
access-list 101 deny icmp 11.11.11.0 0.0.0.255 10.10.12.0 0.0.0.255 echo-reply
! -----
access-list 101 deny tcp 11.11.11.0 0.0.0.255 eq 135 10.10.13.0 0.0.0.255
access-list 101 deny tcp 11.11.11.0 0.0.0.255 eq 139 10.10.13.0 0.0.0.255
access-list 101 deny tcp 11.11.11.0 0.0.0.255 eq 445 10.10.13.0 0.0.0.255
access-list 101 deny udp 11.11.11.0 0.0.0.255 eq 137 10.10.13.0 0.0.0.255
access-list 101 deny udp 11.11.11.0 0.0.0.255 eq 138 10.10.13.0 0.0.0.255
access-list 101 deny tcp 11.11.11.0 0.0.0.255 eq 3389 10.10.13.0 0.0.0.255
access-list 101 deny icmp 11.11.11.0 0.0.0.255 10.10.13.0 0.0.0.255 echo-reply
! -----
access-list 101 permit ip any any
!
interface FastEthernet1
 ip address 11.11.11.1 255.255.255.0
 ip access-group 101 in
!
```

## The Access Control List **established** Keyword

- The **established** keyword is only applicable to TCP access list entries to match TCP segments that have the ACK **and** / **or** RST control bit set (regardless of the source and destination ports), which assumes that a TCP connection has already been established. The **non-matching** cases are initial TCP connection-establishment segments which have only the SYN bit set.
- A typical usage is to distinguish the connections originating inside from connections originating elsewhere. Figure A6-14 shows a scenario which allows internal systems to initiate Telnet connections to any Internet site (outside network), but not the other way around. A simple solution is to block incoming packets that don't have the ACK or RST bits set by using the **established** keyword, which permits return traffic for connections that are established and initiated from the inside, and denies connections initiated from outside to inside.

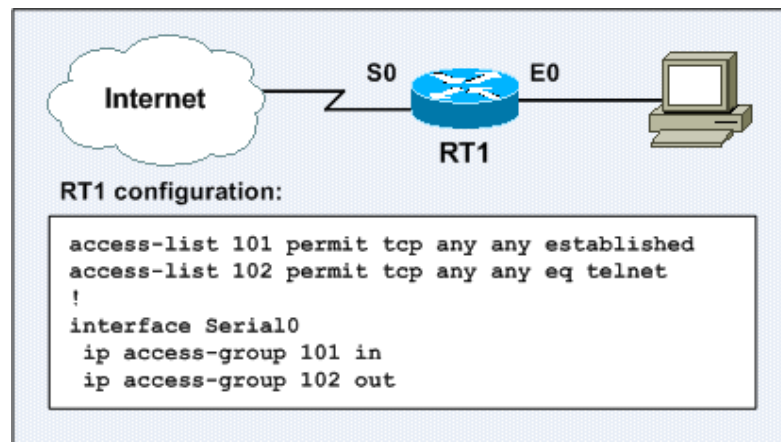


Figure A6-14: Sample ACL established Network I

- **Note:** This method of blocking unwanted traffic originating outside the network can be circumvented – it is possible to forge a packet with the appropriate bits set.
- Another usage is to allow connections to be initiated from client systems only, but not from the server to the others. This can prevent abuse from the server and tighten the server to offer only the necessary services.

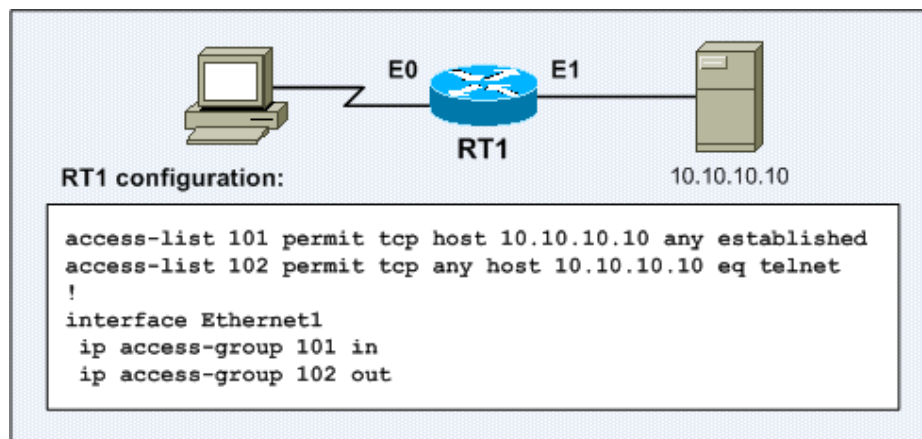


Figure A6-15: Sample ACL established Network II

- The **access-list 101 permit tcp any any established** is equivalent to **access-list 101 permit tcp any any ack rst**.

## The Access Control List **fragments** Keyword

- The **fragments** keyword indicates that an access list entry is **only** applied to **non-initial packet fragments** (L3). The fragment is either permitted or denied accordingly. The default behavior is without the **fragments** keyword.
- The **fragments** keyword cannot be configured for an access list entry that contains any L4 information. Ex: **access-list 101 permit tcp host 1.1.1.1 host 2.2.2.2 eq 80 fragments** is invalid.
- Without the **fragments** keyword (default), an access list entry that contains only L3 information (eg: **access-list 101 permit ip host 10.10.10.1 host 10.10.10.2**) is matched with all types of packets – non-fragmented packets, initial and non-initial packet fragments.  
**Note:** The **fragments** keyword indicates that an access list entry will be matched with non-initial fragments.
- Without the **fragments** keyword (default), an access list entry that contains L3 and L4 information (eg: **access-list 101 permit tcp host 10.10.10.1 host 10.10.10.2 eq telnet**) is matched with non-fragmented packets and initial packet fragments and is either permitted or denied accordingly.
  - o When a **permit** statement (without the **fragments** keyword) is matched with non-initial packet fragments, the non-initial fragments are permitted.
  - o When a **deny** statement (without the **fragments** keyword) is matched with non-initial packet fragments, the next access list entry is processed, and the fragments are not being denied!

**Summary:** The **deny** statements (without the **fragments** keyword) are handled differently for non-initial packet fragments than the **permit** statements (without the **fragments** keyword).

- Do not simply add the **fragments** keyword to every access list entry, as the 1st fragment (initial fragment) will not be matched with an access list **permit** or **deny** entry that contains the **fragments** keyword – the packet is compared to the next access list entry, until it is either permitted or denied by an access list entry that does not contain the **fragments** keyword. Therefore, 2 **deny** access list entries are needed for every **deny** entry. The 1st **deny** entry of the pair will not include the **fragments** keyword to be applied for initial fragments; the 2nd **deny** entry of the pair will include the **fragments** keyword to be applied for non-initial fragments.
- **Note:** Packet fragments are considered individual packets and each is counted individually as a packet in access list accounting.
- When there are multiple **deny** access list entries for a particular host with different L4 ports, only a single **deny** access list entry with the **fragments** keyword for the host is required. All the packet fragments are handled in the same manner by the access list.
- The fragment control feature affect policy routing if the policy routing is based on the **match ip address** command and the access list has entries that match L4 through L7 information. It is possible for non-initial fragments to be matched with the access list and are policy routed, even if the 1st fragment was not policy routed (or the reverse).
- This feature provides a better match capability between initial and non-initial fragments and hence allows the configuration of advanced policy routing.

## Switch Port Access Control Lists

- Switch port ACLs can only be applied as **inbound** lists with **extended named access lists** to **L2 switch interfaces**.
- MAC extended access lists perform filtering based on the source and destination MAC addresses, as well as the optional EtherType information.

```
Switch#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Switch(config)#mac access-list ?
    extended  Extended Access List

Switch(config)#mac access-list extended ?
    WORD      access-list name

Switch(config)#mac access-list extended example01
Switch(config-ext-macl)#deny any host ?
    H.H.H 48-bit destination MAC address

Switch(config-ext-macl)#deny any host 1111.1111.1111
Switch(config-ext-macl)#permit any any
Switch(config-ext-macl)#^Z
Switch#
Switch#sh access-list
Extended MAC access list example01
    deny  any host 1111.1111.1111
    permit any any
Switch#
Switch#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Switch(config)#int fa0/1
Switch(config-if)#mac access-group example01 ?
    in  Apply to Ingress

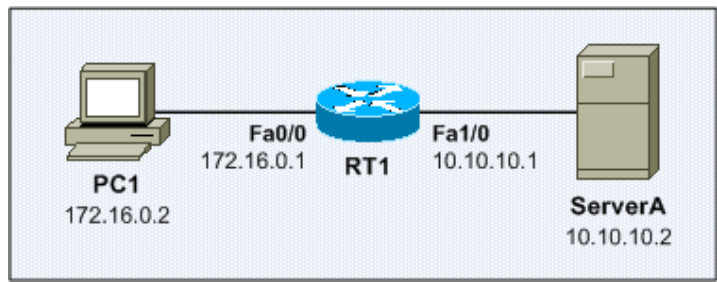
Switch(config-if)#mac access-group example01 in
Switch(config-if)#^Z
Switch#
Switch#sh mac access-group int fa0/1
Interface FastEthernet0/1:
    Inbound access-list is example01
Switch#
```

- The question is do we really want to deny MAC addresses? Deny access based on the EtherType field in the Ethernet frame header is usually the better option.
- Blocking 0x0800 would mean blocking all IP traffic, which could be handy in the future when forcing everyone to run IPv6!

- MAC access lists can filter traffic based on various EtherType:

```
Switch(config-ext-macl)#deny any any ?
<0-65535>      An arbitrary EtherType in decimal, hex, or octal
aarp           EtherType: AppleTalk ARP
amber          EtherType: DEC-Amber
appletalk      EtherType: AppleTalk/EtherTalk
cos            CoS value
dec-spanning   EtherType: DEC-Spanning-Tree
decnet-iv      EtherType: DECnet Phase IV
diagnostic     EtherType: DEC-Diagnostic
dsm            EtherType: DEC-DSM
etype-6000     EtherType: 0x6000
etype-8042     EtherType: 0x8042
lat            EtherType: DEC-LAT
lavc-sca       EtherType: DEC-LAVC-SCA
lsap           LSAP value
mop-console    EtherType: DEC-MOP Remote Console
mop-dump       EtherType: DEC-MOP Dump
msdos          EtherType: DEC-MSDOS
mumps          EtherType: DEC-MUMPS
netbios        EtherType: DEC-NETBIOS
vines-echo     EtherType: VINES Echo
vines-ip       EtherType: VINES IP
xns-idp        EtherType: XNS IDP
<cr>
```

### Advanced Access Control List Configurations



**Figure A6-16:** Network Setup for Advanced Access Control List Configurations

- **Dynamic access lists** or **Lock-and-Key Security** is a type of ACL traffic filtering security feature that dynamically filters IP protocol traffic. Users that would like to traverse through the router are normally blocked by the extended ACL until they authenticates themselves to the router through a Telnet session with a username and password. After the user is authenticated, the Telnet connection is dropped, and a dynamic ACL entry is appended to the existing extended ACL to temporary allow the user to access resources that are blocked behind the router for certain duration until the specified timeout expires.

- Below shows a sample Dynamic Access Lists configuration on RT1:

```
[1] Router(config)#username remote password cisco123
[1] Router(config)#username remote autocommand access-enable host timeout 10
[2] Router(config)#access-list 101 permit tcp any host 172.16.0.1 eq telnet
[3] Router(config)#access-list 101 dynamic remote-access01 timeout 15 permit ip
172.16.0.0 0.0.0.255 10.10.10.0 0.0.0.255
Router(config)#int fa0/0
Router(config-if)#ip access-group 101 in
Router(config-if)#exit
Router(config)#line vty 0 4
[4] Router(config-line)#login local
```

**Note:** Configuring the **autocommand access-enable host timeout 10** line command under **line vty 0 4** which is normally found in other configuration examples is not as flexible as configuring the **autocommand** for particular users with the **username** privileged commands.

- Below describes the configuration steps for the sample dynamic access list configuration:
  - 1) Create a user authentication method on the router. This can either be local authentication or remote security database using RADIUS or TACACS+ server. This sample configuration defines a user named **remote** with a password of **cisco123** and a line of command (**access-enable host timeout 10** in this case) which will be issued automatically (due to the **autocommand** keyword) after the user is authenticated via the Telnet session to the router.

**Note:** The **access-enable EXEC** command creates a temporary access-list entry.
  - 2) Define an extended ACL to allow only Telnet access to the router (for authentication), but block all other traffic.
  - 3) Create a dynamic ACL that applies to the extended access list 101 after the user is authenticated via the Telnet session to the router.
  - 4) Since this sample configuration is using local authentication, the router needs to be configured to locally authenticate when a user connects to its VTY ports.
- **Reflexive access lists** are dynamic access lists that allow traffic based on the detection of traffic in the opposite direction as well as upper-layer session information. They often permit outbound traffic which originated from an inside network but deny inbound traffic which originated from an outside network.
- Reflexive ACLs cannot be defined with standard or numbered ACLs; they can only be defined with **extended named ACLs** and can be used along with other standard or extended ACLs. They are temporary entries that are created when a new IP session begins and are removed when the session ends (when the last segment with FIN or RST is seen or the idle timeout expires). Reflexive ACLs are not applied directly to an interface, but are “nested” within an extended named IP ACL that is applied to an interface.

- Below shows a sample Reflexive Access Lists configuration on RT1 as well as the output of the **show access-list EXEC** command on RT1:

```

!
ip access-list extended Telnet-In
  evaluate RACL-1
  deny ip any any
ip access-list extended Telnet-Out
  permit tcp host 172.16.0.2 host 10.10.10.2 eq telnet reflect RACL-1
  deny ip any any
!
interface FastEthernet1/0
ip address 10.10.10.1 255.255.255.0
ip access-group Telnet-In in
ip access-group Telnet-Out out
!
-----
RT1#sh access-list
Reflexive IP access list RACL-1
Extended IP access list Telnet-In
  10 evaluate RACL-1
  20 deny ip any any
Extended IP access list Telnet-Out
  10 permit tcp host 172.16.0.2 host 10.10.10.2 eq telnet reflect RACL-1
  20 deny ip any any
RT1#

```

- **Time-based access lists** provide time-oriented access control. A certain time of day and week is specified and the period is often identified with a **time range reference name**. The time range name will be used as a reference in extended ACL configuration.
- Below shows a sample Time-based Access Lists configuration that defines **no Internet access during office hours** – Monday to Friday, 9am to 6pm.

```

Router#conf t
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#time-range no-http
Router(config-time-range)#?
Time range configuration commands:
  absolute  absolute time and date
  default   Set a command to its defaults
  exit      Exit from time-range configuration mode
  no        Negate a command or set its defaults
  periodic  periodic time and date

Router(config-time-range)#periodic ?
Friday      Friday
Monday      Monday
Saturday    Saturday
Sunday      Sunday
Thursday    Thursday
Tuesday     Tuesday
Wednesday   Wednesday
daily       Every day of the week
weekdays   Monday thru Friday
weekend     Saturday and Sunday

Router(config-time-range)#periodic weekdays 09:00 to 18:00
Router(config-time-range)#exit

```

```

Router(config)#ip access-list extended Time_no-http
Router(config-ext-nacl)#deny tcp any any eq www time-range no-http
Router(config-ext-nacl)#permit tcp any any
Router(config-ext-nacl)#exit
Router(config)#int fa0/0
Router(config-if)#ip access-group Time_no-http in
Router(config-if)#^Z
Router#
Router#sh clock
10:00:01.835 UTC Sun Jan 6 2008
Router#sh time-range
time-range entry: no-http (inactive)
    periodic weekdays 9:00 to 18:00
    used in: IP ACL entry
Router#sh access-list
Extended IP access list Time_no-http
    deny tcp any any eq www time-range no-http (inactive)
    permit tcp any any
Router#
-----
Router#sh clock
10:00:01.139 UTC Mon Jan 7 2008
Router#sh time-range
time-range entry: no-http (active)
    periodic weekdays 9:00 to 18:00
    used in: IP ACL entry
Router#sh access-list
Extended IP access list Time_no-http
    deny tcp any any eq www time-range no-http (active)
    permit tcp any any
Router#

```

- **Context-Based Access Control (CBAC)** is a Cisco IOS Firewall feature which provides advanced traffic filtering functionality and secure access control on a per-application basis. CBAC is often referred to as **Stateful IOS Firewall Inspection**.
- CBAC provides multiple levels of network protection with the following functions:

<p><b>Traffic Filtering</b></p>	<p>Intelligently filters TCP and UDP segments based on application layer protocol information (eg: FTP connection information). Without CBAC, traffic filtering is limited to access list implementations which only able to examine the information at the network and transport layers. Due to the capability of learning the state information of the sessions and control them, CBAC supports filtering for application layer protocols that involve multiple channels created as a result of negotiations in the control channel. Many multimedia protocols and other protocols (eg: FTP, RPC, SQL*Net) involve multiple channels in their communications. CBAC can be configured to inspect application layer traffic for sessions and connections that are originated from either side of a firewall and permit the specified traffic through it; hence it can be used for intranet, extranet, and Internet perimeters of a network.</p>
---------------------------------	---

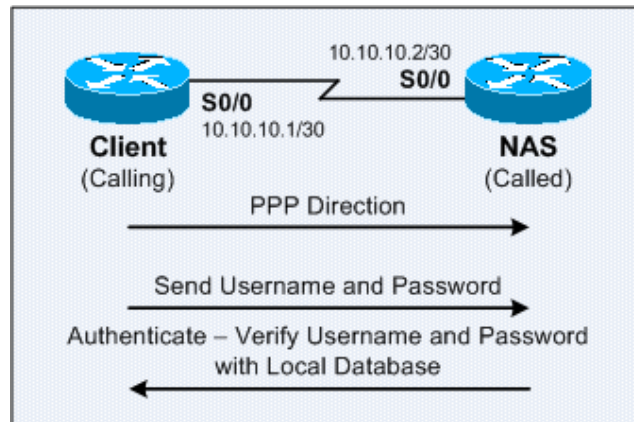
<b>Traffic Inspection</b>	<p>Inspects traffic that passes through a firewall to discover and manage state information for TCP and UDP sessions. The state information is used to create temporary openings in the access lists to allow return traffic for permissible sessions. With the capabilities of inspecting and maintaining TCP and UDP session information, CBAC is able to detect and prevent certain types of network attacks, eg: SYN Flooding. CBAC helps to protect against DoS attacks with the following approaches:</p> <ul style="list-style-type: none"> <li>i) Inspects TCP sequence numbers in to see if they are within expected ranges and drops the suspicious packets.</li> <li>ii) Drops half-open connections, which require firewall processing and memory resources to maintain.</li> <li>iii) Detects unusually high rates of new connections and issue alert messages.</li> </ul> <p>Besides that, CBAC can also help to protect against certain DoS attacks which involve fragmented IP packets. Even though the firewall prevents an attacker from establishing actual connections to an end system, the attacker can disrupt services on the end system by sending many non-initial IP fragments, which can eventually tie up resources on the target end system as it tries to reassemble the incomplete packets.</p>
<b>Alerts and Audit Trails</b>	<p>Generates real-time alerts and audit trails for tracking suspicious activities and network transactions (eg: time stamps, source and destination hosts, ports, total number of transmitted bytes, etc). With CBAC inspection rules, alerts and audit trail information can be configured on a per-application protocol basis.</p>
<b>Intrusion Prevention</b>	<p>Provides a limited amount of intrusion detection to protect against specific SMTP attacks. With intrusion detection, SYSLOG messages are reviewed and monitored for specific attack signatures. When CBAC detects an attack, it resets the offending connections and generates a SYSLOG message to a SYSLOG server.</p>

- CBAC only effective for the specified protocols. If a particular is not specified for CBAC, the existing access lists will determine how the traffic for the particular protocol is being processed. No temporary openings will be created for protocols that are not specified for CBAC inspection.
- CBAC can only provide protection against certain types of attacks, but not all types of attacks. CBAC should not be considered as a perfect defense; there is no such thing as a perfect defense. CBAC detects and prevents most types of popular network attacks.
- **Authentication Proxy** requires the Cisco IOS Firewall feature set as well. It is able to authenticate inbound and/or outbound users and grant the access to the resources that are blocked behind a firewall. By launching a browser to access resources behind the firewall, the firewall would respond to the HTTP session and redirect the user to an authentication page which a valid username and password must be supplied for authentication purpose. An authentication entry will be created for the user. There is no intervention by the authentication proxy for subsequent HTTP sessions through the firewall as long as a valid authentication entry exists for the user.

## Optional PPP Commands

- The **compress predictor** interface subcommand enables the PPP Predictor compression algorithm; while the **compress stac** interface subcommand enables the PPP Stacker compression algorithm.
- The **ppp quality {percent}** interface subcommand ensures a PPP link meets a percentage of quality during the optional link-quality determination phase in the PPP link establishment phase. This is to determine whether the link quality is sufficient to bring up any Layer 3 protocols. The link will not be established if it does not meet the percentage level.  
**Note:** PPP link establishment is handled by Link Control Protocol (LCP).
- The **ppp authentication {pap | chap} {chap | pap}** interface subcommand defines a PPP link to use the 1st authentication protocol, but will try 2nd authentication protocol if the 1st authentication protocol fails or rejected by the other side.
- The **ppp sent-username {username} password {password}** interface subcommand is mandate in PAP configuration.
- In PPP authentication configuration, passwords are case sensitive but usernames are **not** case-sensitive.

## Unidirectional PPP PAP Authentication



**Figure A6-17:** Unidirectional PPP PAP Authentication

- Unidirectional PPP PAP authentication configuration on Client:

```
interface Serial0/0
 ip address 10.10.10.1 255.255.255.252
 encapsulation ppp
 ppp direction callout
 ppp pap sent-username Client password 0 cisco
```

**Note:** The **ppp authentication pap** interface subcommand is not required on Client.

- Unidirectional PPP PAP authentication configuration on NAS:

```
username Client password 0 cisco
!
interface Serial10/0
 ip address 10.10.10.2 255.255.255.252
 encapsulation ppp
 ppp authentication pap [callin]
 ppp direction callin
```

**Note:** The **callin** keyword of the **ppp authentication pap** interface subcommand is optional

- A router configured with the **ppp authentication pap** interface subcommand will use PAP to verify the identity of the peer, which means that the peer must present its username and password to the local device for verification. The local device would use the local username-based authentication system to verify and authenticate its peer.
- The function of the **username {remote-username} password {passwd}** statement is different for PAP and CHAP. With PAP, it is only used to verify that an incoming username and password; whereas CHAP uses it to generate the response to a challenge and verify a response.
- For one-way PAP authentication, the **username {remote-hostname} password {passwd}** statement is only required on the called device to verify the username and password sent by the calling device; whereas for two-way PAP authentication, it is required on both devices.
- A router configured with the **ppp authentication pap callin** interface subcommand configured will only authenticate the peer during incoming calls – it will not authenticate the peer for outgoing calls.
- The **ppp pap sent-username {local-username} password {passwd}** interface subcommand is configured on the calling device to authenticate itself to a remote called device. The remote device must have the same set of **username – password** statement configured.
- The **ppp direction {callin | callout | dedicated}** interface subcommand is introduced in Cisco IOS Release 12.2T. This command is useful when a router is connected to an interface type where there is no inherent call direction, eg: a back-to-back or leased-line connection.

- Below shows the output of the PPP authentication debug messages for a successful unidirectional PAP authentication on Client:

```

Client#sh debug
PPP:
  PPP authentication debugging is on
  PPP protocol negotiation debugging is on

Client#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Client(config)#int s1/0
Client(config-if)#no shut
Client(config-if)#
00:07:14: %LINK-3-UPDOWN: Interface Serial0/0, changed state to up
00:07:14: Se0/0 PPP: Using configured call direction
00:07:14: Se0/0 PPP: Treating connection as a callout
00:07:14: Se0/0 PPP: Session handle[5A000003] Session id[25]
00:07:14: Se0/0 PPP: Phase is ESTABLISHING, Active Open
00:07:14: Se0/0 PPP: Authorization required
00:07:14: Se0/0 PPP: No remote authentication for call-out
00:07:14: Se0/0 LCP: O CONFREQ [Closed] id 25 len 10
00:07:14: Se0/0 LCP:   MagicNumber 0x0013D707 (0x05060013D707)
00:07:14: Se0/0 LCP: I CONFREQ [REQsent] id 83 len 14
00:07:14: Se0/0 LCP:   AuthProto PAP (0x0304C023)
00:07:14: Se0/0 LCP:   MagicNumber 0x0113BFC6 (0x05060113BFC6)
00:07:14: Se0/0 LCP: O CONFACK [REQsent] id 83 len 14
00:07:14: Se0/0 LCP:   AuthProto PAP (0x0304C023)
00:07:14: Se0/0 LCP:   MagicNumber 0x0113BFC6 (0x05060113BFC6)
00:07:14: Se0/0 LCP: I CONFACK [ACKsent] id 25 len 10
00:07:14: Se0/0 LCP:   MagicNumber 0x0013D707 (0x05060013D707)
00:07:14: Se0/0 LCP: State is Open
00:07:14: Se0/0 PPP: No authorization without authentication
00:07:15: Se0/0 PPP: Phase is AUTHENTICATING, by the peer
00:07:15: Se0/0 PAP: Using hostname from interface PAP
00:07:15: Se0/0 PAP: Using password from interface PAP
00:07:15: Se0/0 PAP: O AUTH-REQ id 25 len 17 from "Client"
00:07:15: Se0/0 PAP: I AUTH-ACK id 25 len 5
00:07:15: Se0/0 PPP: Phase is FORWARDING, Attempting Forward
00:07:15: Se0/0 PPP: Queue IPCP code[1] id[1]
00:07:15: Se0/0 PPP: Phase is ESTABLISHING, Finish LCP
00:07:15: Se0/0 PPP: Phase is UP
00:07:15: Se0/0 IPCP: O CONFREQ [Closed] id 1 len 10
00:07:15: Se0/0 IPCP:   Address 10.10.10.1 (0x03060A0A0A01)
00:07:15: Se0/0 CDPCP: O CONFREQ [Closed] id 1 len 4
00:07:15: Se0/0 PPP: Process pending ncp packets
00:07:15: Se0/0 IPCP: Redirect packet to Se1/0
00:07:15: Se0/0 IPCP: I CONFREQ [REQsent] id 1 len 10
00:07:15: Se0/0 IPCP:   Address 10.10.10.2 (0x03060A0A0A02)
00:07:15: Se0/0 IPCP: O CONFACK [REQsent] id 1 len 10
00:07:15: Se0/0 IPCP:   Address 10.10.10.2 (0x03060A0A0A02)
00:07:15: Se0/0 CDPCP: I CONFREQ [REQsent] id 1 len 4
00:07:15: Se0/0 CDPCP: O CONFACK [REQsent] id 1 len 4
00:07:15: Se0/0 IPCP: I CONFACK [ACKsent] id 1 len 10
00:07:15: Se0/0 IPCP:   Address 10.10.10.1 (0x03060A0A0A01)
00:07:15: Se0/0 IPCP: State is Open
00:07:15: Se0/0 IPCP: Add link info for cef entry 10.1.1.2
00:07:15: Se0/0 IPCP: Install route to 10.1.1.2
00:07:15: Se0/0 CDPCP: I CONFACK [ACKsent] id 1 len 4
00:07:15: Se0/0 CDPCP: State is Open
00:07:16: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/0, changed
state to up
Client(config-if)#

```

- Below shows the output of the PPP authentication debug messages for a successful unidirectional PAP authentication on NAS:

```

NAS#
00:07:08: Se0/0 LCP: I CONFREQ [Listen] id 25 len 10
00:07:08: Se0/0 LCP:   MagicNumber 0x0013D707 (0x05060013D707)
00:07:08: Se0/0 PPP: Authorization required
00:07:08: Se0/0 LCP: O CONFREQ [Listen] id 83 len 14
00:07:08: Se0/0 LCP:   AuthProto PAP (0x0304C023)
00:07:08: Se0/0 LCP:   MagicNumber 0x0113BFC6 (0x05060113BFC6)
00:07:08: Se0/0 LCP: O CONFACK [Listen] id 25 len 10
00:07:08: Se0/0 LCP:   MagicNumber 0x0013D707 (0x05060013D707)
00:07:09: Se0/0 LCP: I CONFACK [ACKsent] id 83 len 14
00:07:09: Se0/0 LCP:   AuthProto PAP (0x0304C023)
00:07:09: Se0/0 LCP:   MagicNumber 0x0113BFC6 (0x05060113BFC6)
00:07:09: Se0/0 LCP: State is Open
00:07:09: Se0/0 PPP: Phase is AUTHENTICATING, by this end
00:07:09: Se0/0 PAP: I AUTH-REQ id 25 len 17 from "Client"
00:07:09: Se0/0 PAP: Authenticating peer Client
00:07:09: Se0/0 PPP: Phase is FORWARDING, Attempting Forward
00:07:09: Se0/0 PPP: Phase is AUTHENTICATING, Unauthenticated User
00:07:09: Se0/0 PPP: Sent PAP LOGIN Request
00:07:09: Se0/0 PPP: Received LOGIN Response PASS
00:07:09: Se0/0 PPP: Phase is FORWARDING, Attempting Forward
00:07:09: Se0/0 PPP: Phase is AUTHENTICATING, Authenticated User
00:07:09: Se0/0 PPP: Sent LCP AUTHOR Request
00:07:09: Se0/0 PPP: Sent IPCP AUTHOR Request
00:07:09: Se0/0 LCP: Received AAA AUTHOR Response PASS
00:07:09: Se0/0 IPCP: Received AAA AUTHOR Response PASS
00:07:09: Se0/0 PAP: O AUTH-ACK id 25 len 5
00:07:09: Se0/0 PPP: Phase is UP
00:07:09: Se0/0 IPCP: O CONFREQ [Closed] id 1 len 10
00:07:09: Se0/0 IPCP:   Address 10.10.10.2 (0x03060A0A0A02)
00:07:09: Se0/0 PPP: Sent CDPCP AUTHOR Request
00:07:09: Se0/0 PPP: Process pending ncp packets
00:07:09: Se0/0 CDPCP: Received AAA AUTHOR Response PASS
00:07:09: Se0/0 CDPCP: O CONFREQ [Closed] id 1 len 4
00:07:09: Se0/0 IPCP: I CONFREQ [REQsent] id 1 len 10
00:07:09: Se0/0 IPCP:   Address 10.10.10.1 (0x03060A0A0A01)
00:07:09: Se0/0 AAA/AUTHOR/IPCP: Start. Her address 10.1.1.1, we want
0.0.0.0
00:07:09: Se0/0 PPP: Sent IPCP AUTHOR Request
00:07:09: Se0/0 AAA/AUTHOR/IPCP: Reject 10.1.1.1, using 0.0.0.0
00:07:09: Se0/0 AAA/AUTHOR/IPCP: Done. Her address 10.1.1.1, we want
0.0.0.0
00:07:09: Se0/0 IPCP: O CONFACK [REQsent] id 1 len 10
00:07:09: Se0/0 IPCP:   Address 10.10.10.1 (0x03060A0A0A01)
00:07:09: Se0/0 CDPCP: I CONFREQ [REQsent] id 1 len 4
00:07:09: Se0/0 CDPCP: O CONFACK [REQsent] id 1 len 4
00:07:09: Se0/0 IPCP: I CONFACK [ACKsent] id 1 len 10
00:07:09: Se0/0 IPCP:   Address 10.10.10.2 (0x03060A0A0A02)
00:07:09: Se0/0 IPCP: State is Open
00:07:09: Se0/0 CDPCP: I CONFACK [ACKsent] id 1 len 4
00:07:09: Se0/0 CDPCP: State is Open
00:07:09: Se0/0 IPCP: Add link info for cef entry 10.1.1.1
00:07:09: Se0/0 IPCP: Install route to 10.1.1.1
00:07:10: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/0, changed
state to up
NAS#

```

- Below shows the output of the PPP authentication debug messages for a failed unidirection PAP authentication (wrong **username password** statement configured on NAS) on Client and NAS:

```

Client(config-if)#no shut
Client(config-if)#
00:12:38: Se0/0 PPP: Phase is ESTABLISHING, Passive Open
00:12:38: Se0/0 LCP: State is Listen
00:12:38: Se0/0 LCP: State is Open
00:12:38: Se0/0 PPP: No authorization without authentication
00:12:38: Se0/0 PPP: Phase is AUTHENTICATING, by the peer
00:12:38: Se0/0 PAP: Using hostname from interface PAP
00:12:38: Se0/0 PAP: Using password from interface PAP
00:12:38: Se0/0 PAP: O AUTH-REQ id 40 len 17 from "Client"
00:12:39: Se0/0 PAP: I AUTH-NAK id 40 len 26 msg is "Authentication failed"
00:12:39: Se0/0 LCP: I TERMREQ [Open] id 134 len 4
00:12:39: Se0/0 LCP: O TERMACK [Open] id 134 len 4
00:12:39: Se0/0 PPP: Sending Acct Event[Down] id[29]
00:12:39: Se0/0 PPP: Phase is TERMINATING
00:12:41: Se0/0 LCP: TIMEout: State TERMSent
00:12:41: Se0/0 LCP: State is Closed
00:12:41: Se0/0 PPP: Phase is DOWN
-----
NAS#
00:12:30: Se0/0 PPP: Phase is ESTABLISHING, Passive Open
00:12:30: Se0/0 LCP: State is Listen
00:12:32: Se0/0 LCP: TIMEout: State Listen
00:12:32: Se0/0 PPP: Authorization required
00:12:32: Se0/0 LCP: State is Open
00:12:32: Se0/0 PPP: Phase is AUTHENTICATING, by this end
00:12:32: Se0/0 PAP: I AUTH-REQ id 40 len 17 from "Client"
00:12:32: Se0/0 PAP: Authenticating peer Client
00:12:32: Se0/0 PPP: Phase is FORWARDING, Attempting Forward
00:12:32: Se0/0 PPP: Phase is AUTHENTICATING, Unauthenticated User
00:12:32: Se0/0 PPP: Sent PAP LOGIN Request
00:12:32: Se0/0 PPP: Received LOGIN Response FAIL
00:12:32: Se0/0 PAP: O AUTH-NAK id 40 len 26 msg is "Authentication failed"
00:12:32: Se0/0 PPP: Sending Acct Event[Down] id[28]
00:12:32: Se0/0 PPP: Phase is TERMINATING
00:12:32: Se0/0 LCP: O TERMREQ [Open] id 134 len 4
00:12:33: Se0/0 LCP: I TERMACK [TERMSent] id 134 len 4
00:12:33: Se0/0 LCP: State is Closed
00:12:33: Se0/0 PPP: Phase is DOWN

```

## Unidirectional PPP CHAP Authentication

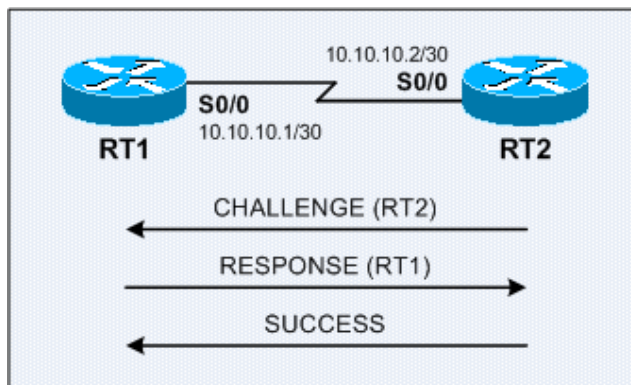


Figure A6-18: Unidirectional PPP CHAP Authentication

- This lab demonstrates the mechanisms of unidirectional (one-way) PPP CHAP authentication. Below describes the situation of the initial lab setup:
  - i) Both routers have no username and PPP authentication configurations.
  - ii) Both routers can ping each other.
  - iii) Both routers have enabled PPP authentication debugging with **debug ppp authentication**.
- The **username password** statement is required on **both devices** for both unidirectional and bidirectional CHAP authentication. In unidirectional CHAP authentication (a local device authenticating a remote device), it is first used by the remote device (RT1) to respond to the challenge generated by the local device (RT2), and then used by the local device (RT2) to verify the response from the remote device (RT1).
- Below enable PPP CHAP authentication on RT2. The debugging messages show RT2 challenges RT2 but RT1 is unable to response to RT2's challenge.

```
RT2(config)#int s0/0
RT2(config-if)#ppp authentication chap
RT2(config-if)#
00:15:52: Se0/0 CHAP: O CHALLENGE id 1 len 24 from "RT2"
00:15:53: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/0, changed
state to down
00:15:54: Se0/0 CHAP: O CHALLENGE id 2 len 24 from "RT2"
00:15:56: Se0/0 CHAP: O CHALLENGE id 3 len 24 from "RT2"
-----
RT1#
00:15:52: Se0/0 CHAP: I CHALLENGE id 1 len 24 from "RT2"
00:15:52: Se0/0 CHAP: Username RT2 not found
00:15:52: Se0/0 CHAP: Unable to authenticate for peer
00:15:52: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/0, changed
state to down
00:15:54: Se0/0 PPP: Using default call direction
00:15:54: Se0/0 PPP: Treating connection as a dedicated line
00:15:54: Se0/0 CHAP: I CHALLENGE id 2 len 24 from "RT2"
00:15:54: Se0/0 CHAP: Username RT1 not found
00:15:54: Se0/0 CHAP: Unable to authenticate for peer
00:15:56: Se0/0 CHAP: I CHALLENGE id 3 len 24 from "RT2"
00:15:56: Se0/0 CHAP: Username RT1 not found
00:15:56: Se0/0 CHAP: Unable to authenticate for peer
```

- Below configure the username and password on RT1. The debugging message show RT1 is able to response to RT2's challenge but RT2 is unable to validate RT1's response.

```

RT1 (config) #username RT2 password cisco123
RT1 (config) #
00:18:24: Se0/0 CHAP: I CHALLENGE id 67 len 24 from "RT2"
00:18:24: Se0/0 CHAP: O RESPONSE id 67 len 24 from "RT1"
00:18:24: Se0/0 CHAP: I FAILURE id 67 len 26 msg is "Authentication failure"
00:18:26: Se0/0 CHAP: I CHALLENGE id 68 len 24 from "RT2"
00:18:26: Se0/0 CHAP: O RESPONSE id 68 len 24 from "RT1"
00:18:26: Se0/0 CHAP: I FAILURE id 68 len 26 msg is "Authentication failure"
-----
RT2#
00:18:24: Se0/0 CHAP: O CHALLENGE id 67 len 24 from "RT2"
00:18:24: Se0/0 CHAP: I RESPONSE id 67 len 24 from "RT1"
00:18:24: Se0/0 CHAP: Unable to validate Response. Username RT1 not found
00:18:24: Se0/0 CHAP: O FAILURE id 67 len 26 msg is "Authentication failure"
00:18:26: Se0/0 CHAP: O CHALLENGE id 68 len 24 from "RT2"
00:18:26: Se0/0 CHAP: I RESPONSE id 68 len 24 from "RT1"
00:18:26: Se0/0 CHAP: Unable to validate Response. Username RT1 not found
00:18:26: Se0/0 CHAP: O FAILURE id 68 len 26 msg is "Authentication failure"

```

**Note:** The alternative configuration on RT1 is the **ppp chap hostname RT1** and **ppp chap password cisco123** interface subcommands.

- Below configure the username and password on RT2 for RT2 to validate RT1's response. Finally the unidirectional (one-way) PPP CHAP authentication is successful.

```

RT2 (config) #username RT1 password cisco123
RT2 (config) #
00:20:54: Se0/0 CHAP: O CHALLENGE id 127 len 24 from "RT2"
00:20:54: Se0/0 CHAP: I RESPONSE id 127 len 24 from "RT1"
00:20:54: Se0/0 CHAP: O SUCCESS id 127 len 4
00:20:54: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/0, changed
state to up
-----
RT1#
00:20:48: Se0/0 CHAP: I CHALLENGE id 127 len 24 from "RT2"
00:20:48: Se0/0 CHAP: O RESPONSE id 127 len 24 from "RT1"
00:20:48: Se0/0 CHAP: I SUCCESS id 127 len 4
00:20:49: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/0, changed
state to up
00:20:50: Se0/0 PPP: Using default call direction
00:20:50: Se0/0 PPP: Treating connection as a dedicated line

```

- LCP packets are sent during the PPP link establishment phase. These packets contain several Configuration Option fields that allow PPP devices to negotiate how they want the link to be established – the maximum datagram size the link can carry, authentication, quality monitoring, and compression. If no Configuration Option field is present, the default configurations are used.

## Frame Relay DLCIs

- DLCIs are 10 bits long. Below shows the range of Frame Relay DLCI. Practically, it allows up to 992 PVCs per Frame Relay physical link.

DLCI Range	Usage
0	LMI for ANSI Annex D and CCITT/ITU-T Q.933A type LMIs
1 - 15	Reserved for future use
16 - 1007	User traffic
1008 - 1022	Reserved for future use
1023	Reserved for LMI and CLLM messages for Cisco type LMI

**Note:** CLLM is referred to as **Consolidated Link Layer Management**.

- The LMI DLCI in the **show interface EXEC** command defines the type of LMI being used.

<b>ANSI</b>	LMI DLCI 0 LMI type is ANSI Annex D frame relay DTE
<b>Cisco</b>	LMI DLCI 1023 LMI type is CISCO frame relay DTE
<b>CCITT/ITU-T Q.933A</b>	LMI DLCI 0 LMI type is CCITT frame relay DTE

```
Router#sh int s0/0
Serial0/0 is up, line protocol is up
  Hardware is M4T
  Internet address is 10.10.10.10/24
  MTU 1500 bytes, BW 1544 Kbit, DLY 20000 usec,
    reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation FRAME-RELAY, crc 16, loopback not set
  Keepalive set (10 sec)
  LMI enq sent 75, LMI stat recvd 76, LMI upd recvd 0, DTE LMI up
  LMI enq recvd 0, LMI stat sent 0, LMI upd sent 0
  LMI DLCI 0 LMI type is ANSI Annex D frame relay DTE
--- output omitted ---
```

- The **ITU Telecommunication Standardization Sector (ITU-T)** coordinates standards for telecommunications on behalf of **International Telecommunication Union (ITU)**. The **Comité Consultatif International Téléphonique et Télégraphique (CCITT)** or International Telegraph and Telephone Consultative Committee was created in 1956. It was renamed to ITU-T in 1993.
- Multicasting support is an extension of the LMI specification that allows efficient distribution of routing information and ARP requests across a Frame Relay network. Multicasting uses the reserved DLCIs from 1019 to 1022.

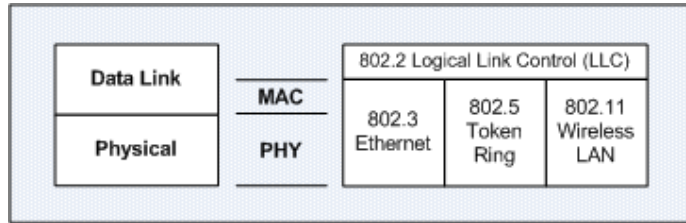
## IEEE 802.11 Standards and Specifications

- Below lists the IEEE 802.11 Standards / Specifications as well as their purpose:

<b>IEEE 802.11a</b>	54Mbps 5GHz standard.
<b>IEEE 802.11b</b>	Enhancements to 802.11 to support 5.5Mbps and 11Mbps.
<b>IEEE 802.11c</b>	Bridge operation procedures. Included in the IEEE 802.1d standard.
<b>IEEE 802.11d</b>	International (country-to-country) roaming extensions.
<b>IEEE 802.11e</b>	Enhancements to 802.11 – Quality of Service. Including packet bursting.
<b>IEEE 802.11F</b>	Inter-Access Point Protocol.
<b>IEEE 802.11g</b>	54Mbps 2.4GHz standard. Backward compatible with 802.11b.
<b>IEEE 802.11h</b>	Spectrum Managed 5GHz 802.11a – <b>Dynamic Frequency Selection (DFS)</b> and <b>Transmit Power Control (TPC)</b> .
<b>IEEE 802.11i</b>	<b>Wi-Fi Protected Access 2 (WPA2)</b> for enhanced security (authentication and encryption). Also referred to as <b>Robust Security Network (RSN)</b> .
<b>IEEE 802.11j</b>	Extensions for Japan and US public safety.
<b>IEEE 802.11k</b>	Enhancements to 802.11 – Radio Resource Management (RRM).
<b>IEEE 802.11m</b>	Maintenance of the standard, odds and ends.
<b>IEEE 802.11n</b>	Higher throughput improvements using <b>Multiple Input, Multiple Output (MIMO)</b> antennas.
<b>IEEE 802.11p</b>	<b>Wireless Access for the Vehicular Environment (WAVE)</b> for vehicular environments, eg: ambulances and passenger cars.
<b>IEEE 802.11r</b>	Fast roaming.
<b>IEEE 802.11s</b>	<b>Extended Service Set (ESS) Mesh Networking</b> .
<b>IEEE 802.11T</b>	<b>Wireless Performance Prediction (WPP)</b> .
<b>IEEE 802.11u</b>	Internetworking with non-802 networks, eg: cellular networks.
<b>IEEE 802.11v</b>	Wireless network management.
<b>IEEE 802.11w</b>	Protected Management Frames.
<b>IEEE 802.11y</b>	3650–3700 operation in the US.
<b>IEEE 802.11z</b>	Extensions to <b>Direct Link Setup (DLS)</b> .

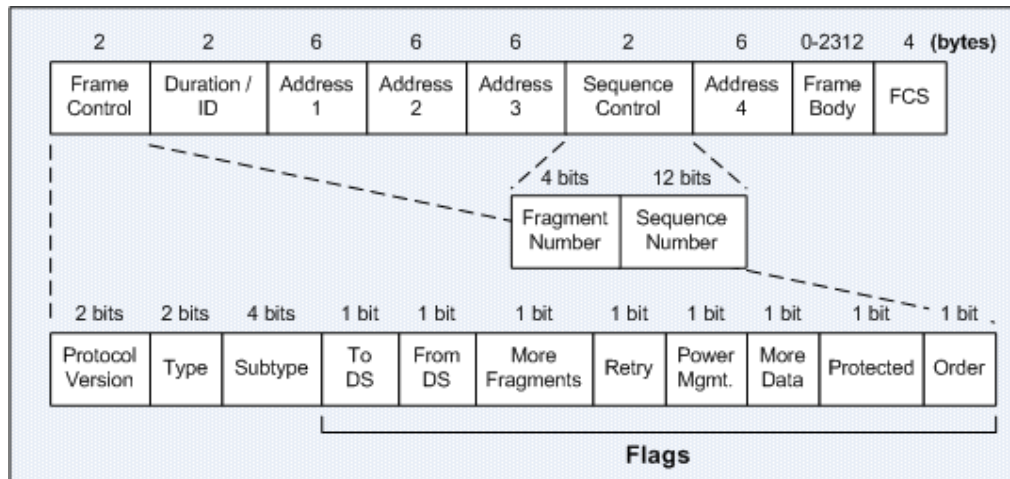
- The IEEE project naming convention uses upper-case letters (eg: 802.1Q) to identify **standalone** standards, and lower-case letters to identify **amendments** (previously known as supplements) to existing standards. There should never be 2 projects differing only in the case of letters!
- The REV notation (eg: 802.1Q-REV) is used to identify a revision of an existing standard, which has more extensive changes to the existing standard than an amendment. Previously, revisions also had their own project names.

## IEEE 802.11 Frame Types



**Figure A6-19:** 802.11 in the OSI Reference Model

- The IEEE 802.11 architecture resides in the Data Link **Media Access Control** (MAC) sublayer and the Physical layer in the OSI reference model.
- The **Basic Service Set** (BSS) and **Extended Services Set** (ESS) are the 2 available **infrastructure** modes of WLAN. Another mode of WLAN is **ad-hoc** mode.
- **Basic Service Set** (BSS) is the basic building block of a WLAN. It is often being referred to as the coverage area of an access point. An access point acts as a master to control the wireless stations within a BSS. A BSS is identified by an SSID. The most basic BSS is **Independent Basic Service Set** (IBSS) or **ad-hoc mode BSS**, which comprised of 2 wireless clients; whereas the most basic **infrastructure mode BSS** is comprised of an access point and a wireless client.
- The IEEE 802.11 WLAN specification defines various frame types than Ethernet for wireless communications, as well as managing and controlling wireless connections. The types of frames in the IEEE 802.11 specification are **management, control, and data frames**. Understanding the different IEEE 802.11 frame types is essential for analyzing and troubleshooting the operation of WLANs.
- Every IEEE 802.11 WLAN frame contains the MAC addresses of the source and destination wireless stations, a Frame Control field that indicates the 802.11 protocol version, frame type, and various indicators (eg: whether WEP is enabled, power management is active, etc), a Sequence Control field, the frame body, and the Frame Check Sequence for error detection.



**Figure A6-20:** 802.11 Frame Format

- Figure A6-20 shows the frame format of the IEEE 802.11 WLAN specification. Below describes the subfields and flags in the Frame Control field:

<b>Protocol Version</b>	Indicates the version of the 802.11 protocol. A receiving station uses this value to determine whether it supports the version of the protocol of the received frame.
<b>Type and Subtype</b>	Determine the function of the frame – management, control, or data. The type and subtype fields for each frame type determine the specific function to perform.
<b>To DS and From DS</b>	Indicates whether the frame is destined to or exiting from the distributed system (DS). All frames of wireless stations that are associated with an access point (infrastructure mode) will have one of the DS bits set. The interpretation of the Address fields depends on the setting of these bits.
<b>More Fragments</b>	Indicates whether there are more subsequent fragments for a particular management or data frame are to follow. Control frames are not fragmented, hence this bit is always set to 0 for control frames.
<b>Retry</b>	Indicates whether the management or data frame is being retransmitted.
<b>Power Management</b>	Indicates whether the sending wireless station is in active or power-saving mode.
<b>More Data</b>	Used to inform a wireless station which is in power-saving mode that the access point has more frames to send to it. Also used by an access points to indicate that additional broadcast or multicast frames are to follow. This bit is only being used in management and data frames; hence this bit is always set to 0 for control frames.
<b>Protected</b>	Indicates whether encryption and authentication are used for the frame. Control frames may not be encrypted; hence this bit is always set to 0 for control frames.
<b>Order</b>	Indicates that all received data frames must be processed in sequence.

- Below shows how to interpret the **To DS** and **From DS** bits:

	<b>From DS = 0</b>	<b>From DS = 1</b>
<b>To DS = 0</b>	All management, control, and data frames within an IBSS (ad-hoc).	Data frames arrived at a wireless station (from AP) in an infra. WLAN.
<b>To DS = 1</b>	Data frames transmitted from a wireless station (to AP) in an infra. WLAN.	Data frames on a wireless bridge (WDS, <b>Wireless Distribution System</b> ).

- The Duration/ID field is used in all control frames (except with the subtype of PS-Poll) to indicate the remaining duration needed to receive the next frame transmission.
- Wireless stations may want to save battery power by turning off antennas. When the subtype is PS-Poll, it contains the **association identity** (AID) of the waking transmitting station, which indicates which BSS the station belongs to. **Note:** PS is referred to as **Power Save**.

- An 802.11 frame may contain up to 4 Address fields. The general rule is that Address 1 indicates the receiver of a frame, Address 2 as the transmitter, and Address 3 for filtering by the receiver. Depends upon the type of frame, the 4 Address fields will contain a combination of the following address types:

<b>BSS Identifier (BSSID)</b>	Used to uniquely identify each BSS (WLAN). When the frame is from a wireless station in an infrastructure BSS, the BSSID is the MAC address of the access point; when the frame is from a wireless station in an IBSS (ad-hoc) mode, the BSSID is a locally administered MAC address generated with a 46-bit random number, and is generated by the wireless station that initiated the IBSS.
<b>Source Address (SA)</b>	Indicates the 48-bit MAC address of the source station that created and transmitted the frame (source of the transmission). Only 1 station can be the source of a frame.
<b>Destination Address (DA)</b>	Indicates the 48-bit MAC address of the destination station to receive the frame (recipient).
<b>Transmitter Address (TA)</b>	Indicates the 48-bit MAC address of the wireless interface that transmitted the frame onto the wireless medium. The TA is only being used in <b>wireless bridging</b> .
<b>Receiver Address (RA)</b>	Indicates the 48-bit MAC address of the (immediate) wireless station which should receive and process the frame. If it is a wireless station, the RA is the DA. For frames destined to a node on an Ethernet network connected to an access point, the RA is the wireless interface of the access point, and the DA may be a node attached to the Ethernet.

- Below shows the usage of the Address fields in data frames:

Function	To DS	From DS	Address 1 (RX)	Address 2 (TX)	Address 3	Address 4
IBSS	0	0	DA	SA	BSSID	-
From AP	0	1	DA	BSSID	SA	-
To AP	1	0	BSSID	SA	DA	-
WDS	1	1	RA	TA	DA	SA

**Note:** Address 1 indicates the receiver; while Address 2 indicates the transmitter.

- The Sequence Control field contains the following 2 subfields:

<b>Fragment Number</b>	Indicates the number of each frame of a fragmented upper-layer packet. The 1st fragment will have a fragment number of 0, and each subsequent fragment of a fragmented packet increments the fragment number incremented by one.
<b>Sequence Number</b>	Indicates the sequence number of each frame. It begins at 0 and incremented by 1 until 4095 and rollovers to zero and begins again ( <b>modulo-4096</b> ). All fragments of a fragmented packet as well as retransmitted frames will have the same sequence number.

- Below lists the IEEE 802.11 **management frames** that allow wireless stations to establish and maintain communications:

<b>Association Request</b>	The 802.11 association process allows an access point to synchronize and allocate resources for a wireless adapter. A wireless adapter begins the process by sending an Association Request frame to an access point. Upon receiving the Association Request frame, the access point is considered associated with the wireless adapter and would allocate an association ID and resources for the wireless adapter. An Association Request frame contains information such as the SSID of the WLAN the wireless client wishes to associate with and the supported data rates.
<b>Association Response</b>	An access point would send an Association Response frame containing an acceptance or rejection notice to the wireless adapter requesting association. An Association Response frame contains information, eg: the association ID and the supported data rates.
<b>Reassociation Request</b>	When a wireless adapter roams away from its currently associated access point after found another access point with a stronger beacon signal, the wireless adapter would send a Reassociation Request frame to the new access point. The new access point would then coordinate with the previous access point to forward the data frames meant for the wireless adapter that may still be in the buffer of the previous access point.
<b>Reassociation Response</b>	An access point sends a Reassociation Response frame containing an acceptance or rejection notice to a wireless adapter requesting reassociation. Similar to the Association Response frame, the Reassociation Response frame contains information regarding an association – the association ID and the supported data rates.
<b>Probe Request</b>	A wireless station sends a Probe Request frame when it would like to obtain information of another wireless station. Ex: A wireless adapter sends a Probe Request frame to determine the access points that are within range.
<b>Probe Response</b>	A wireless station receives a Probe Request frame would respond with a Probe Response frame that contains capability information, eg: the supported data rates.
<b>Beacon</b>	An access point sends Beacon frames periodically to announce its presence and the services it offers using SSID, timestamp, and other access point parameters to wireless adapters that are within range. Wireless adapters continuously scan all 802.11 radio channels for beacon frames to choose the best access point to associate with. Beacon frames are also used to logically separate WLANs.
<b>Disassociation</b>	A wireless station sends a Disassociation frame to another wireless station when it would like to terminate the association. Ex: A wireless adapter that is shutting down gracefully can send a Disassociation frame to notify its associated access point that it is powering off. The access point can then remove the wireless adapter from the association table and release the allocated memory resources.

<b>Authentication</b>	The 802.11 authentication process is where an access point accepts or rejects the identity of a wireless adapter. A wireless adapter begins the process by sending an Authentication frame that contains its identity to the access point. For open authentication, the access point responds with an Authentication frame as a response to indicate the acceptance or rejection; while for shared-key authentication, the access point responds with an Authentication frame containing challenge text, which the wireless client must response with an Authentication frame containing the encrypted version of the challenge text using the shared-key for the access point to verify its identity. WLAN authentication occurs at L2 and is authenticating devices instead of users. The authentication and association processes are occurred in sequence. <b>Note:</b> Authentication occurs first and then followed by association.
<b>Deauthentication</b>	A wireless station sends a Deauthentication frame to another wireless station in order to terminate a secure connection.

- Below lists the IEEE 802.11 **control frames** that assist the delivery of data frames between wireless stations:

<b>Request to Send (RTS)</b>	A station sends a RTS frame to another station as the 1st phase of the necessary 2-way handshake before transmitting a data frame.
<b>Clear to Send (CTS)</b>	A station response to a RTS frame with the CTS frame to provide the clearance for the source station to transmit a data frame. The CTS frame contains a time value which would cause all nearby stations (including hidden stations) to hold off data transmission for a certain period of time necessary for the source station to transmit its frames.
<b>Acknowledgement (ACK)</b>	A destination station would run an error checking process to detect the presence of errors upon received a data frame. The destination station would send an ACK frame to the source station if no errors are found. The source station will retransmit the frame if it doesn't receive an ACK for the frame for a certain period of time.

**Note:** Kindly refer to Page 174 for the discussion of the CSMA/CA and RTS/CTS mechanisms.

- Finally, **data frames** are used to carry upper layers data – packets.
- Below shows the wireless client association process:
  - i) Access points send out beacons announcing the SSID and supported data rates.
  - ii) A wireless client scans all channels and sends out Probe Request frames to all access points within range.
  - iii) All access points within range reply with a Probe Response frame, and the wireless client listens for the responses from the access points.
  - iv) The wireless client associates with the access point with the strongest signal. Authentication and other security information are sent to the access point.
  - v) The access point accepts the association request and associated with the wireless client.

**Note:** 802.1X authentication could occur straight after the association process is completed.
- The maximum Ethernet frame size is 1518 bytes whereas a wireless frame could be as large as 2346 bytes. Usually the WLAN frame size is limited to 1518 bytes as WLANs are often connected to and communicating with wired Ethernet networks.

## IEEE 802.11 Types and Subtypes

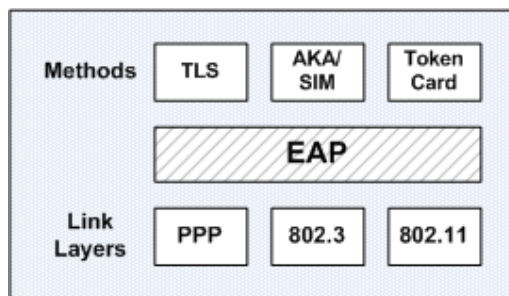
Type Value b3 b2	Type Description	Subtype Value B7 b6 b5 b4	Subtype Description
00	Management	0000	Association Request
00	Management	0001	Association Response
00	Management	0010	Reassociation Request
00	Management	0011	Reassociation Response
00	Management	0100	Probe Request
00	Management	0101	Probe Response
00	Management	0110-0111	Reserved
00	Management	1000	Beacon
00	Management	1001	ATIM
00	Management	1010	Disassociation
00	Management	1011	Authentication
00	Management	1100	Deauthentication
00	Management	1101-1111	Reserved
01	Control	0000-1001	Reserved
01	Control	1010	PS-Poll
01	Control	1011	RTS
01	Control	1100	CTS
01	Control	1101	ACK
01	Control	1110	CF End
01	Control	1111	CF End + CF-ACK
10	Data	0000	Data
10	Data	0001	Data + CF-ACK
10	Data	0010	Data + CF-Poll
10	Data	0011	Data + CF-ACK + CF-Poll
10	Data	0100	Null function (no data)
10	Data	0101	CF-ACK (no data)
10	Data	0110	CF-Poll (no data)
10	Data	0111	CF-ACK + CF-Poll (no data)
10	Data	1000-1111	Reserved
11	Reserved	0000-1111	Reserved

## IEEE 802.1X Port-Based Authentication

- The IEEE 802.1X standard defines a client-server-based access control and authentication protocol that restricts unauthorized devices from gaining access to a network through publicly accessible ports. The authentication server authenticates a client connects to a switch port before granting the available network services to the client.
- 802.1X allows only **Extensible Authentication Protocol over LAN (EAPOL)** traffic through the switch port which an unauthenticated client is connected to. Normal traffic can pass through the switch port after authentication is completed successfully.
- With 802.1X port-based authentication, network devices have the following **device roles**:

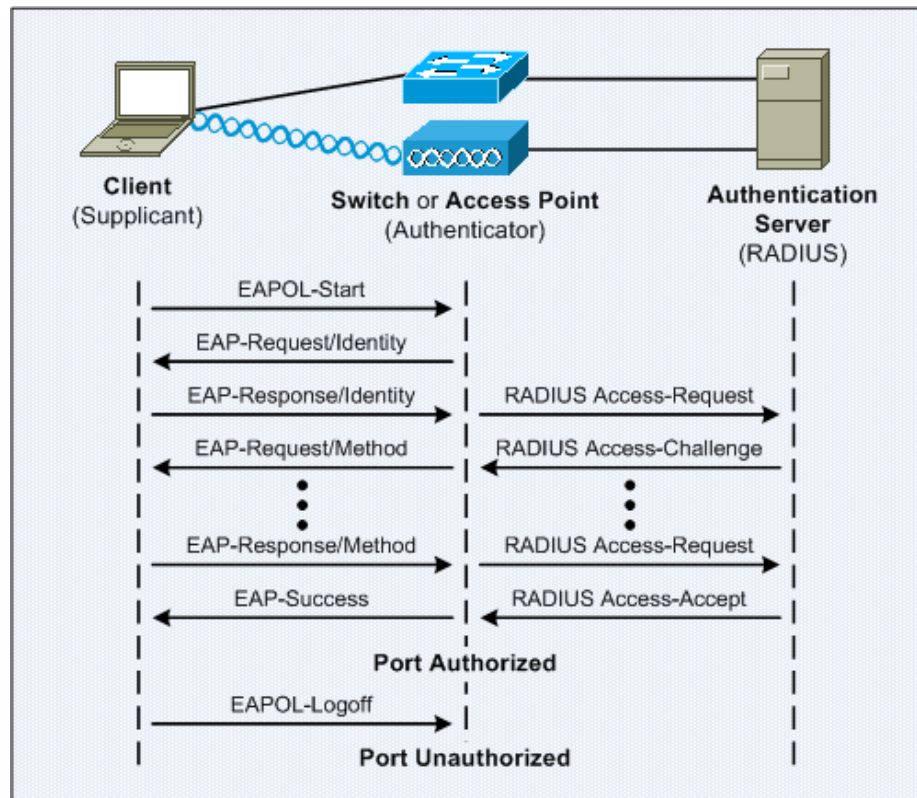
<b>Client (Supplicant)</b>	The device (workstation) that requires access to the LAN. Responds to the requests from the switch. Must be running 802.1X-compliant client software.
<b>Switch or Access Point (Authenticator)</b>	Controls the physical access to the network based on the authentication status of the client. Acts as a proxy between the client and authentication server, which requests identity information from the client, verify the information with the authentication server, and relays the response to the client. It is responsible for the <b>re-encapsulation</b> of the EAP and RADIUS frames for communication with a client and an authentication server respectively.
<b>Authentication Server</b>	Performs the authentication of the client. It validates the identity of the client and notifies the switch or access point whether the client is authorized to access the network. Currently, the RADIUS security system with EAP extensions is the only supported authentication server.

**Note:** RADIUS is referred to as **Remote Authentication Dial-In User Service**.



**Figure A6-21: The EAP Architecture**

- EAP is designed to run over any link layer and use any number of authentication methods.



**App06-22: 802.1X Authentication Message Exchange**

- Below describes a typical WLAN 802.1X authentication process:
  - 1) A wireless client becomes active on the medium. After the IEEE 802.11 WLAN Probe Request/Response, Authentication, and Association processes, the access point forces the port into an unauthorized state, which only 802.1X traffic is forwarded.
  - 2) The access point replies with an EAP-Request Identity message to the wireless client to obtain the client's identity. The wireless client's EAP-Response Identity message, which contains the client's identity, is forwarded to the authentication server.
  - 3) The authentication server authenticates the wireless client and sends an Access Accept (or Access Reject) message to the access point.
  - 4) Upon receiving the Access Accept message, the access point relays the EAP-Success message to the wireless client and transitions the client's port to an authorized state and normal traffic is forwarded.
- **Note:** Below lists the sequence of establishing WLAN connectivity with 802.1X authentication: **Probe Request, Probe Response, Authentication, Association, 802.1X authentication...**
- When 802.1X is enabled, ports are authenticated before any other L2 or L3 features are enabled.
- **Note:** 802.1X is also considered as an efficient and effective alternative solution to port security.

## VPN and IPsec Basics

- Virtual Private Networks (VPNs) allow the creation of private networks across a public domain (eg: Internet) for private and secured data transmission.
- VPNs are secure and inexpensive. There are 2 types of VPN implementation – using IPsec to create authentication and encryption services; and establishing tunnels via tunneling protocols.
- Tunneling is method of overcoming protocol restrictions by encapsulating or wrapping packets from a protocol in another protocol and transmits the encapsulated packet over a network that supports the encapsulating protocol.
- Below describes the most common tunneling protocols:

<b>Layer 2 Forwarding (L2F)</b>	A Cisco-proprietary tunneling protocol created for <b>Virtual Private Dial-Up Networks (VPDNs)</b> , where devices create secure connections to a corporate network using dial-up connections. L2F was later replaced by L2TP (backward compatible with L2F).
<b>Point-to-Point Tunneling Protocol (PPTP)</b>	Created by Microsoft for secure data transmission between remote networks and corporate network.
<b>Layer 2 Tunneling Protocol (L2TP)</b>	Created by Cisco and Microsoft to replace L2F and PPTP. L2TP merged both L2F and PPTP capabilities into one tunneling protocol.
<b>Generic Routing Encapsulation (GRE)</b>	Another Cisco-proprietary tunneling protocol. It forms (unencrypted) virtual point-to-point links which are able to encapsulate a variety of protocols inside IP tunnels. It can also be used to protect the private address space from being advertised to any public domain.

**Note:** There are many types of tunnels. VPNs are tunnels; GRE creates tunnels; Secure Shell (SSH) is also a form of tunnel.

- IPsec is a suite of industry standard network layer (L3) protocols and algorithms for securing communications over the IP-based networks by authenticating and/or encrypting IP packets. IPsec also includes the IKE protocol for establishing shared security information (**Security Association, SA**) between 2 network devices or end systems to support secure communication.  
**Note:** Other Internet security protocols, eg: SSL and SSH, are operate from the Transport layer up to the Application layer (L4 – L7).  
**Note:** The official term of “**IPsec**” as defined by the IETF is often wrongly written as “**IPSec**”.
- IPsec secures a path between a pair of gateways, a pair of hosts, or a gateway and a host.
- IPsec unable to encrypt non-IP traffic. For such a situation, a GRE tunnel would first be created to encrypt the non-IP traffic and followed by using IPsec to encrypt the GRE tunnel. Additionally, Multicast can only be passed on GRE tunnels.

- IPsec-based VPN is comprised of 2 parts – **Internet Key Exchange (IKE)** protocol and IPsec security protocols – **Authentication Header (AH)** and **Encapsulating Security Payload (ESP)**. Below describes the flow of IPsec events:
  - i) **IKE Phase 1: IKE Security Negotiation** – IKE negotiates how to protect IKE by establishing an authenticated and secure channel between 2 IKE peers called the **IKE Security Association**. IKE Phase 1 is consists of **Main Mode** or **Aggressive Mode**. The peer that initiates the session will propose or offer at least one or more configured ISAKMP policies which specify a combination of encryption algorithm, hash algorithm, authentication type, Diffie-Hellman group, and the lifetime. The remote peer will then try to find a matching configured policy that has the same parameters as the one being sent by its peer. If no matching policy is found, IKE will terminate the negotiation. If a policy is mutually agreed upon, IKE will complete the negotiation process and an **ISAKMP SA** will be created. Additionally, peers in an IPsec session must authenticate themselves among each other during IKE Phase 1 Main Mode exchange before IKE can proceed.
  - ii) **IKE Phase 2: IPsec Security Negotiation** – IKE negotiates how to protect IPsec by negotiating the IPsec security associations (SAs) and generating the keys for IPsec. IKE Phase 2 negotiation is done in only 1 mode – **Quick Mode**. The peer that initiates the session will propose or offer at least one or more configured transforms which specify a combination of authentication and/or encryption algorithm. The remote peer will then try to find a matching configured transform that has the same parameters as the one being sent by its peer. If no matching transform is found, IKE will terminate negotiation and an IPsec VPN will not be established. If a policy is mutually agreed upon, IKE will complete the negotiation process and an **IPsec SA** will be created.
  - iii) IPsec transfers the actual data in the VPN tunnel using the authentication and encryption methods agreed upon the IKE negotiation process.
  
- **Internet Key Exchange (IKE)** allows 2 VPN endpoints verify the identity of each other (using pre-shared keys or RSA) in IKE Phase 1, and negotiate the methods (security policies) for secured data transmission in IKE Phase 2. IKE manages VPN connections by defining a set of **Security Associations (SAs)** for each connection. Each SA has its own SAID.
  
- In a VPN, before a communication path is considered secure, the VPN endpoints must be authenticated. IPsec uses the following authentication methods to authenticate peers:
 

<b>Pre-Shared Keys</b>	Secret key values that are manually configured on each peer.
<b>RSA signatures</b>	Use the exchange of digital certificates to authenticate the peers.
  
- IKE is a hybrid protocol that uses part of Oakley and part of SKEME inside the ISAKMP framework; hence IKE is formerly known as ISAKMP/Oakley. IKE typically uses ISAKMP for **key exchange** and **establishment of SAs**, although other methods can be used.
  
- IKE establishes both ISAKMP and IPsec SAs for an IPsec VPN session. IKE first negotiates an ISAKMP SA with the peer. It is possible to configure multiple policy statements with different parameters, and then allow the peers to negotiate and establish a mutual agreement.
  
- An IPsec SA defines the security algorithms or parameters associated with particular connection – the IPsec protocol (AH, ESP, or both), the session keys used for data encryption, etc. IPsec SAs are unidirectional (simplex); hence there is always more than 1 IPsec SA per IPsec connection. In cases where only either AH or ESP is used, 2 SAs will be created for each connection – one for each the incoming and outgoing traffic. In cases where AH and ESP are used in conjunction, 4 SAs will be created.

- The **Internet Security Association and Key Management Protocol (ISAKMP)** defines the procedures for authenticating peers, IKASAMP and IPsec SAs establishment, negotiation, modification, and deletion; key generation, and threat mitigation (eg: DoS and replay attacks).
- Instead of ISAKMP (the use of **ipsec-isakmp** keyword along with the **crypto map** global configuration command), **manual keying** (the use of **ipsec-manual** keyword along with the **crypto map** global configuration command) which require manual entry of the shared secret session keys (used for hashing and encryption) on both crypto endpoints is also possible.
- IPsec operation requires both ends to be configured with the same **transform set**, which specifies the methods for encrypt and decrypt the data. IPsec uses 2 primary security protocols – **Authentication Header (AH)** and **Encapsulating Security Payload (ESP)**. These protocols are used for secured data transmission through an IPsec-based VPN tunnel. IPsec-based VPNs can be established using AH only, ESP only, or both AH and ESP.
- The **Authentication Header (AH)** protocol provides authentication for both the IP header and data of a packet using a one-way hash function. The sender first generates a one-way hash, and then the receiver generates the same one-way hash. If the packet has changed in any way, it won't be authenticated and will be dropped. IPsec relies upon AH to guarantee authenticity. AH provides integrity check on the entire packet, but it doesn't provide any encryption services.
- ESP only provides integrity check (and encrypts) on the data of a packet (and the ESP header); while AH checks the entire packet – both header and data. AH is used for authentication only, while ESP can be used for either encryption or authentication only; or both.
- Although AH and ESP are typically used independently, they are often being used together to provide data encryption service. **Note:** It is important to use authentication even if encryption is used, as encrypt-only implementations are subject to some forms of effective attacks.
- ESP provides the following 4 functionalities or capabilities:

<b>Confidentiality (Encryption)</b>	Provided through the use of symmetric encryption algorithms, eg: DES, 3DES. Confidentiality can be selected separately from all other services, but the confidentiality selected must be the same on all VPN endpoints.
<b>Data Origin Authentication and Connectionless Integrity</b>	Joint services offered as an option in conjunction with the confidentiality option. Authentication ensures that the connection is established with the desired system.
<b>Anti-Replay Protection</b>	This service works only if data origin authentication is selected. It is based upon the receiver – it is effective only if the receiver checks the sequence number of the received packets with a sliding window of the destination gateway or host to prevent replay attacks.
<b>Traffic Flow Confidentiality</b>	This service works only when tunnel mode is selected. It is most effective if implemented at a security gateway, where the source-destination patterns of attacks is visible.

- The degree of security of IPsec VPN is based on the encryption algorithm used and the length of the pre-shared key. The longer the key, the harder it is to be broken.
- IPsec is not bound to any specific encryption or authentication algorithm, keying or technology, or security algorithms, which allows IPsec to support newer and better algorithms.

- IPsec supports the following 3 types of encryption algorithms:

<b>Data Encryption Standard (DES)</b>	Uses a 56-bit key that ensures high performance encryption. Uses a symmetric key cryptosystem.
<b>Triple DES (3DES)</b>	A variant of DES that breaks data into 64-bit blocks. 3DES then processes each block 3 times, each time with an independent 56-bit key, hence providing significant improvement in encryption strength over DES. Uses a symmetric key cryptosystem.
<b>Advanced Encryption Standard (AES)</b>	Provides stronger encryption than DES and is more efficient than 3DES. Key lengths can be 128-, 192-, and 256-bit keys.

- Encryption algorithms (eg: DES, 3DES, and AES) require a symmetric shared secret key to perform encryption and decryption. The **Diffie-Hellman Key Exchange (D-H)** is a public key exchange process that allows 2 parties that have no prior knowledge of each other to negotiate symmetric shared secret keys used for encryption and decryption over an insecure channel. The shared secret keys are negotiated between crypto endpoints dynamically, which only the prime modulus size for use in the D-H exchange is needed to be specified. IKE uses the D-H keys to encrypt the ISAKMP SA when establishing the IPsec SAs. Additionally, D-H is also used to generate shared secret keys to be used in the ciphers specified in the IPsec transforms, which are then used in conjunction with the D-H keys by the IPsec to encrypt and decrypt the data passes through the IPsec VPN tunnel.

- IPsec uses a data integrity algorithm called **Hash-based Message Authentication Code (HMAC)** that adds a hash to the message to ensure data integrity. The hash guarantees the integrity of the original message. If the transmitted hash matches the received hash, the message is considered has not been tampered.

- IPsec uses the following 2 HMAC algorithms:

<b>Message Digest Algorithm 5 (MD5)</b>	Uses a 128-bit shared secret key. The message and 128-bit shared secret key are combined and run through the MD5 hash algorithm, producing a 128-bit hash. This hash is then added to the original message and sent to the destination host.
<b>Secure Hash Algorithm 1 (SHA-1)</b>	Uses a 160-bit shared secret key. The message and 160-bit shared secret key are combined and run through the SHA-1 hash algorithm, producing a 160-bit hash. This hash is then added to the original message and sent to the destination host.

- Below lists the 2 modes of IPsec operation:

<b>Transport</b>	Only the payload of the IP packet is encrypted and/or authenticated. The routing is intact, as the IP header is neither modified nor encrypted. It is used for host-to-host or end-to-end communications (end systems perform the security processing).
<b>Tunnel</b>	The entire IP packet is encrypted and/or authenticated. The original packet is encapsulated entirely into a new packet with a new set of source and destination IP addresses for routing to work. It is used for network-to-network or portal-to-portal communications (gateways or routers perform the security processing). It is the traditional and <b>default</b> mode of IPsec VPNs.

- The Tunnel mode is most commonly used to secure existing IP traffic for communication between end systems on networks connected to IPsec-enabled routers. With routers or VPN endpoints performing the IPsec encryption, no changes are required to the software and drivers on the end systems – the IPsec implementation is **transparent** to end systems.

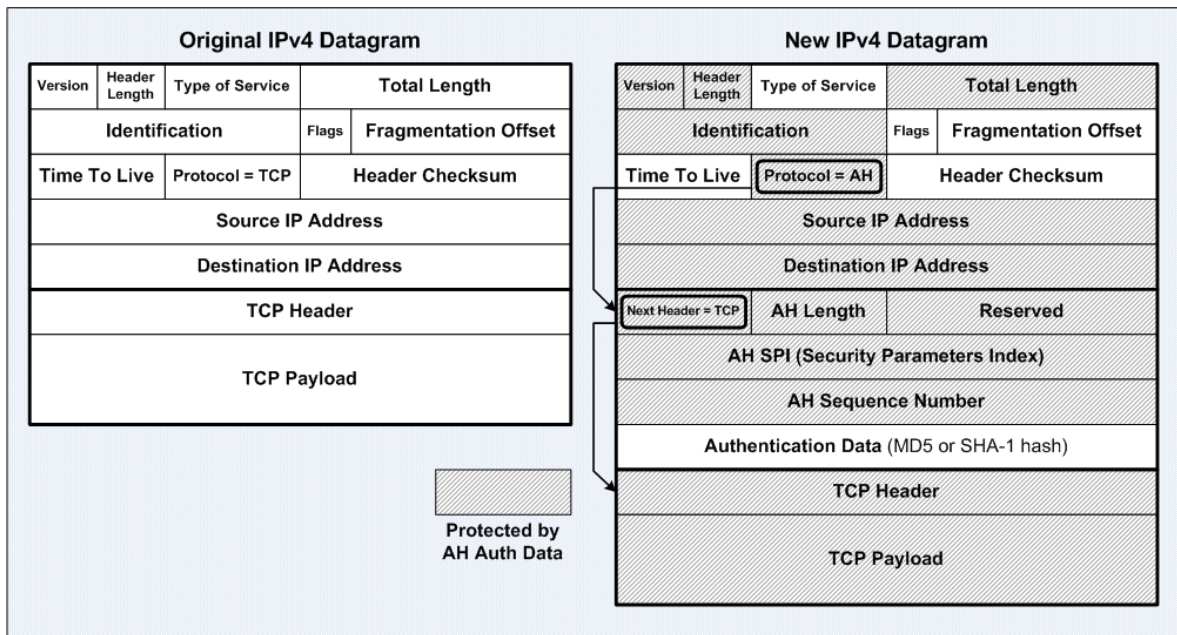


Figure A6-23: IPsec in AH Transport Mode

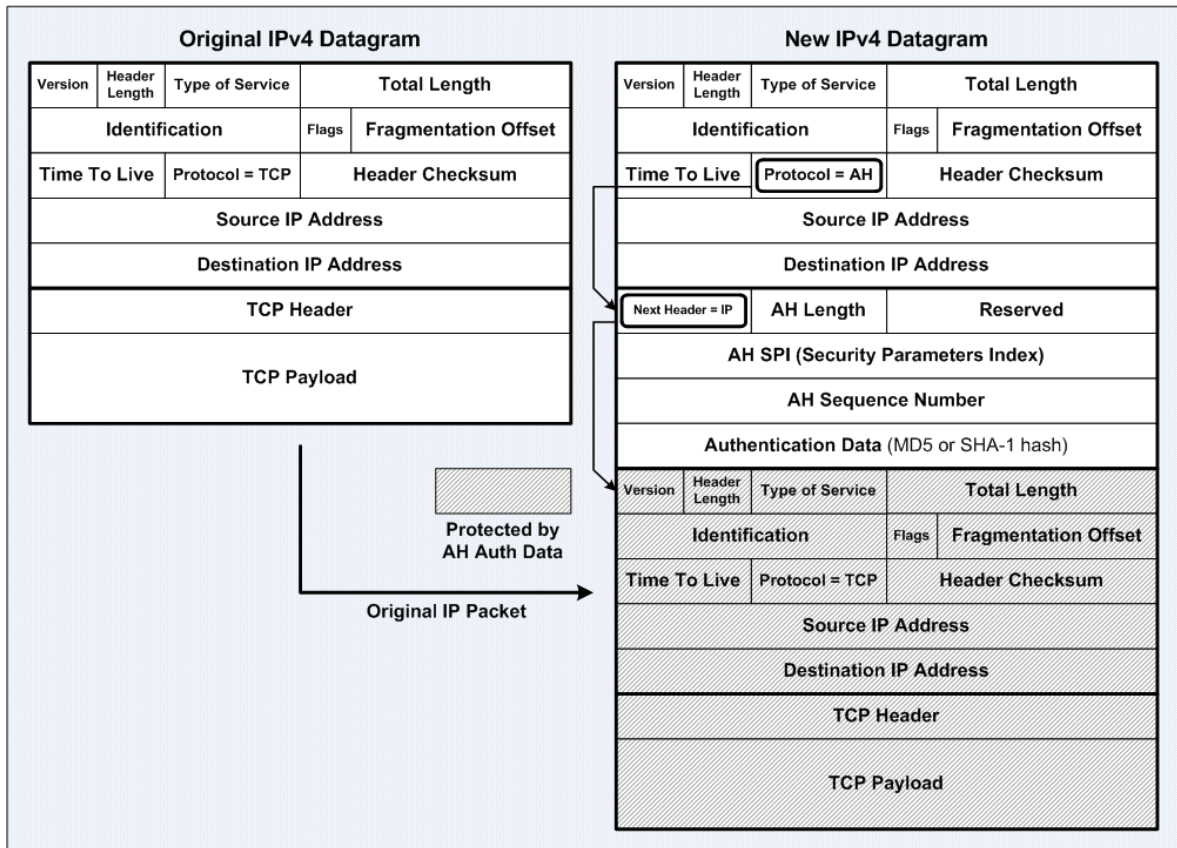


Figure A6-24: IPsec in AH Tunnel Mode

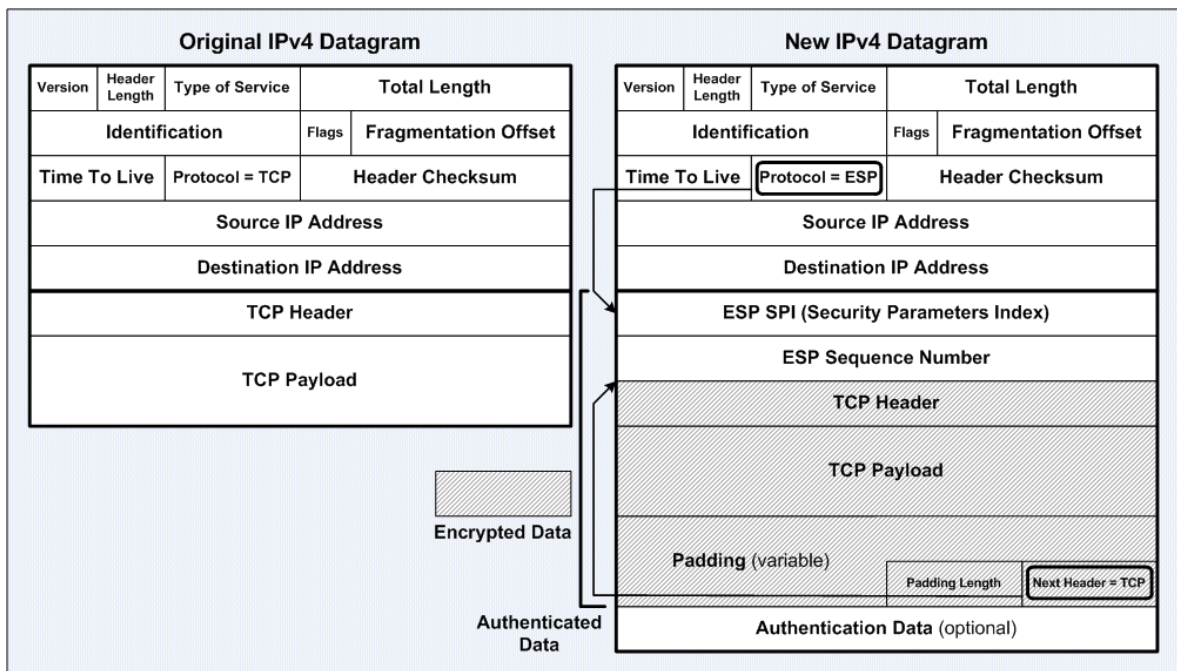


Figure A6-25: IPsec in ESP Transport Mode

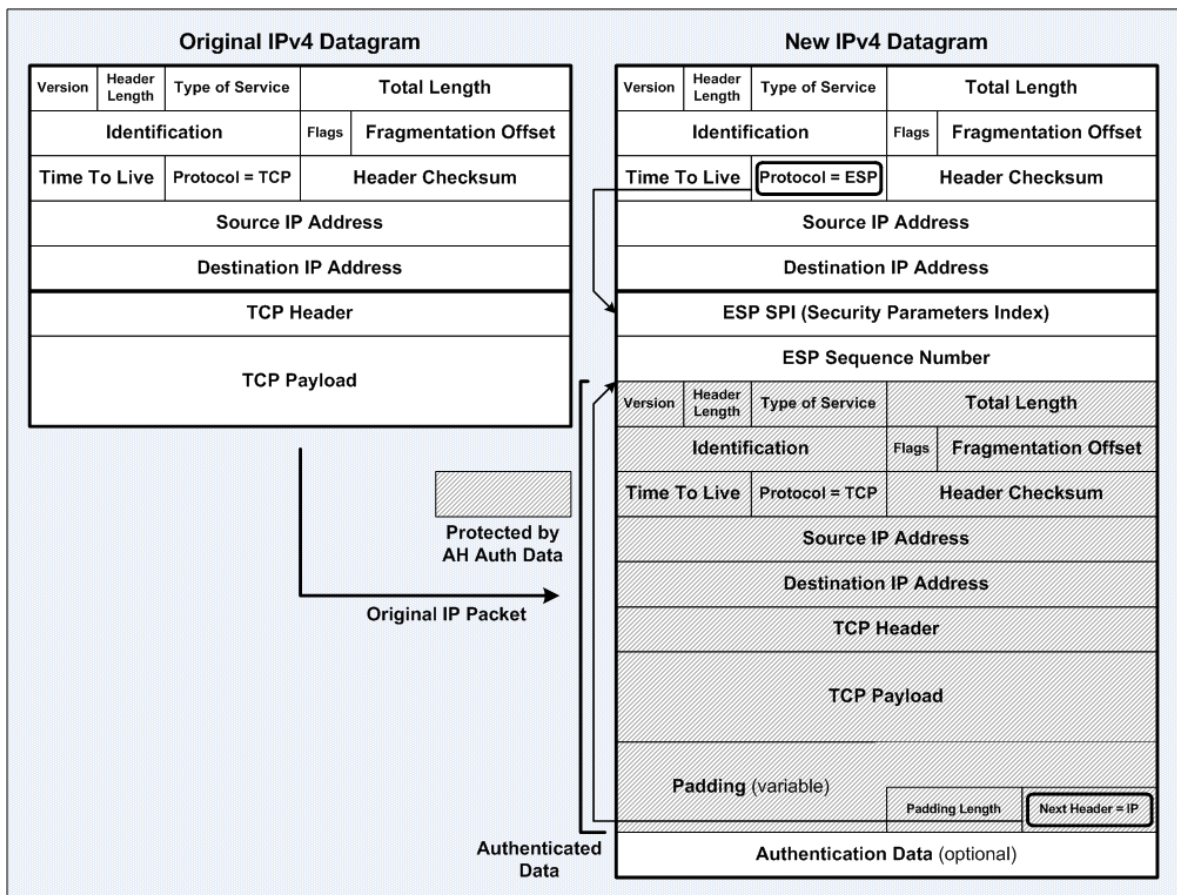


Figure A6-26: IPsec in ESP Tunnel Mode

- The main problem of IKE and IPsec protocols is NAT, as both protocols were not designed to work through NAT. NAT Traversal (NAT-T) has evolved as a method of enabling IPsec-protected IP packets to work well in NAT environments.

## IPsec Configuration

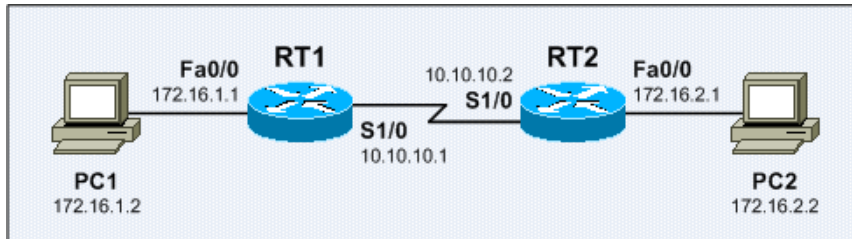


Figure A6-27: Sample IPsec-Based VPN Network

- IPsec configuration on RT1:

```
hostname RT1
!
crypto isakmp policy 1
  encr 3des
  authentication pre-share
  group 2
crypto isakmp key CISCO-1234 address 10.10.10.2
!
!
crypto ipsec transform-set ESP-3DES-SHA esp-3des esp-sha-hmac
!
crypto map CMAP-Site2 1 ipsec-isakmp
  description *** IPsec Tunnel to RT2 ***
  set peer 10.10.10.2
  set transform-set ESP-3DES-SHA
  match address 102
!
interface Serial1/0
  crypto map CMAP-Site2
!
access-list 102 permit ip 172.16.1.0 0.0.0.255 172.16.2.0 0.0.0.255
!
```

- IPsec configuration starts with configuring the **ISAKMP protection suite**. The **crypto isakmp policy** global configuration command first defines an ISAKMP policy. It is possible to define multiple policies; the **priorities** of the policies determine the sequence of the policies during the IKE negotiation phase (IKE Phase 1).
- The **authentication pre-share** ISAKMP subcommand tells IKE to use the manual key configured with the **crypto isakmp key** global configuration command for authentication.  
**Note:** The other 2 options beside the **pre-share** keyword are **rsa-encr** and **rsa-sig**, which configures RSA Encryption and RSA Signature respectively. These keywords are used when configuring ISAKMP using a CA (**Certification Authority**) instead of pre-shared keys.  
**Note:** CA is a 3rd-party entity which is responsible for issuing and revoking digital certificates. Each device that has its own certificate and public key of the CA can authenticate other devices within a particular CA domain.
- The **group {Diffie-Hellman group}** ISAKMP subcommand defines the size of the modulus to use for Diffie-Hellman calculation. Group 1 is 768-bit long, group 2 is 1024-bit long, and group 5 is 1536-bit long. The higher-number groups are significantly more CPU intensive but are more secure than other lower-number groups. The default is group 1.

- It is possible to specify up to 6 transform sets for a particular crypto map and allow the peers to negotiate a mutually agreed transform.

```
!  
crypto ipsec transform-set ESP-DES-MD5 esp-des esp-md5-hmac  
crypto ipsec transform-set ESP-3DES-SHA esp-3des esp-sha-hmac  
!  
crypto map CMAP-Site2 1 ipsec-isakmp  
description *** IPsec Tunnel to RT2 ***  
set peer 10.10.10.2  
set transform-set ESP-DES-MD5 ESP-3DES-SHA  
match address 102  
!
```

## Troubleshooting ISDN

### Troubleshooting ISDN BRI Layer 1

- This section is only applicable to troubleshooting ISDN BRI Layer 1 deactivated status. If the ISDN BRI Layer 1 is up and active as shown in the **show isdn status** EXEC command, kindly proceed to **Troubleshooting ISDN BRI Layer 2**.
- Detailed information on ISDN Layer 1 states and signals are defined in the ITU-T I.430 standard.
- The **show controller bri {num}** EXEC command is used for advanced ISDN Layer 1 troubleshooting by displaying the information about the ISDN Layer 1 activation status.

```
Router>sh controllers bri0/0
BRI unit 0:
Layer 1 is ACTIVATED. (ISDN L1 State F7)
--- output omitted ---
```

- Use the following table to interpret the ISDN Layer 1 states:

L1 State	L1 State Name	L1 State Description
F1	Inactive	In this inactive (powered off) state, the terminal equipment (TE) is not transmitting and cannot detect the presence of any input signal.
F2	Sensing	This state is entered after the TE has been powered on but has not determined the type of signal (if any) that the TE is receiving. When in this state, a TE may go into a lower power consumption state.
F3	Deactivated	This is the deactivated state of the physical protocol. Neither the network termination (NT) nor the TE is transmitting. When in this state, a TE may go to a low power consumption mode.
F4	Awaiting Signal	When the TE wishes to initiate activation, it sends an Activation signal to the NT and awaits a response.
F5	Identifying Input	At the first receipt of any signal from the NT, the TE stops sending Activation signals and awaits the activation signal or synchronized frame from the NT.
F6	Synchronized	When the TE has received an Activation signal from the NT, it responds with a synchronized frame and is awaiting a synchronized frame from the NT.
F7	Activated	This is the normal active state, with the protocol activated in both directions. Both the NT and the TE are transmitting normal frames. State F7 is the only state where the B and D channels contain operational data.
F8	Lost Framing	This is the condition when the TE has lost frame synchronization and is awaiting re-synchronization.

- **Note:** Most of the ISDN Layer 1 states are temporary and can be cleared with the **clear interface bri {num}** privileged command or a router reboot. Contact the Telco for further troubleshooting if those states persist for a long period.

## Troubleshooting ISDN E1/T1 Physical Layer Alarms and Error Events

- This section explains the common alarm types and error events that may appear during the operations of ISDN E1/T1 as well as the troubleshooting techniques.
- The **show controller e1 | t1 {num}** EXEC command displays the controller status specific to the controller hardware, error statistics about the E1/T1 link, local or remote alarm information, and information for identifying and troubleshooting ISDN physical and data link layer problems.
- Issue the **show controller e1 | t1 {num}** EXEC command repeatedly to see if the counters for the frame loss, line code, and slip seconds errors and various alarms are increasing for a particular duration of time, eg: 1 minute, 5 minutes.
- Below shows the sample output of the **show controller e1 | t1 {num}** EXEC command. The output comprises of the alarm and error event sections.

```
Router>sh controller e1
E1 1/0 is up.
  Applique type is Channelized E1 - balanced
  No alarms detected.
  alarm-trigger is not set
  Framing is CRC4, Line Code is HDB3, Clock Source is Line.
  Module type is Channelized E1/T1 PRI
  Version info Firmware: 0000001D, FPGA: C
  Hardware revision is 0.0          , Software revision is 29
  Protocol revision is 1
  number of CLI resets is 16
  Last clearing of alarm counters 1d22h
  receive remote alarm : 0,
  transmit remote alarm : 0,
  receive AIS alarm : 0,
  transmit AIS alarm : 0,
  loss of frame : 0,
  loss of signal : 0,
  Loopback test : 0,
  transmit AIS in TS 16 : 0,
  receive LOMF alarm : 0,
  transmit LOMF alarm : 0,
  MIB data updated every 10 seconds.
  Data in current interval (443 seconds elapsed):
  0 Line Code Violations, 0 Path Code Violations
  0 Slip Secs, 0 Fr Loss Secs, 0 Line Err Secs, 0 Degraded Mins
  0 Errored Secs, 0 Bursty Err Secs, 0 Severely Err Secs, 0 Unavail Secs
  Total Data (last 24 hours)
  0 Line Code Violations, 0 Path Code Violations,
  0 Slip Secs, 0 Fr Loss Secs, 0 Line Err Secs, 0 Degraded Mins,
  0 Errored Secs, 0 Bursty Err Secs, 0 Severely Err Secs, 0 Unavail Secs
Router>
```

- Contact the Telco for encoding and framing settings. HDB3 is the only defined encoding scheme for E1 links, while CRC4 framing format is most widely used. Look for `Clock Source is Line` in the output of the **show controller e1 | t1 {num}** EXEC command to verify that the clocking is derived from the Telco.

- Alarms are serious conditions that require attentions. Excessive errors, hardware problems, and signal disruption can trigger alarms. The alarms are coded as **colors**. Different vendors define alarm differently, and finding the details descriptions of the alarms can be challenging. RFC 1232 describes most alarms; however, they are defined for use in SNMP and are not intended to become a standard for hardware implementation.
- A **Red Alarm** is triggered when a Loss of Signal (LoS) or Loss of Frame (LoF) error is detected. After a local device has a red alarm, it would send a yellow alarm to the far-end. When the far-end receives the yellow alarm, it would trigger a yellow alarm.  
**Note:** A red alarm is usually indicated on the opposite end of the yellow alarm, and vice versa.
- **Loss of Signal (LoS)** is the state where no electrical pulses have been detected – the line is dead, there is no alarm and signal. LoS is equivalent to not connecting any cable to the interface.  
**Loss of Frame (LoF)** indicates that a certain number of frames have been received with framing bits in error. The data is considered garbage as the synchronization between both ends is invalid.
- A sample scenario of red alarm is when something has failed on an ISDN switch, the switch would trigger a local red alarm and sends out yellow alarm signal to alert the neighboring router regarding the problem. Another sample scenario is when an ISDN switch is sending garbled frame to the neighboring router, which sees consecutive LoF errors and declares a red alarm. When the router declares a red alarm, a yellow alarm is sent back out across the link.
- Interpreting the behaviors of red alarms can be challenging and confusing. Generally, a red alarm indicates that there is a problem happening on the local device. However, the router in the 2nd scenario above was receiving too much LoF errors until it is unable to recognize the signals and frames from the ISDN switch.
- A **Yellow Alarm** is also known as a **Remote Alarm Indication (RAI)**. A yellow alarm is declared when a yellow alarm signal is received from the far-end. A yellow alarm does not necessarily indicate a problem on the local device; rather there is a problem at the far-end device, which has declared a red alarm and notifies the local device. Generally, a yellow alarm indicates that the far-end receiver is in a LoS or LoF state. When the receiver experiences LoS or LoF, the transmitter sends a yellow alarm. The circuit is considered as a **one-way link** – the transmitting line towards the far-end device is experiencing problem while the transmitting line towards the local device is functioning as it is able to receive the Yellow Alarm signal.  
**Note:** A yellow alarm is usually indicated on the opposite end of the red alarm, and vice versa.
- A **Blue Alarm** is also known as an **Alarm Indication Signal (AIS)**. RFC 1232 does not define about this condition. A blue alarm is triggered when the receiver receives an AIS. Generally, a blue alarm indicates that there is a problem upstream. An AIS in a framed or unframed all-1s signal which is transmitted to maintain transmission continuity. It typically occurs when the far-end CSU has lost its terminal side equipment or a cable is disconnected.
- The blue alarm is often sent from the service provider to both ends of the circuit to notify them that there is a problem within the service provider's network.

- Below addresses ISDN E1/T1 controller alarms along with the procedures to correct them:

<b>Receive Alarm Indication Signal (rxAIS)</b>	Indicates that a stream of framed or unframed all-1s signal is received. This problem should be cleared when the LoF error is rectified. Change the framing format with the <b>framing {esf   sf   crc4   no-crc4}</b> interface subcommand.
<b>Receive Remote Alarm Indication (rxRAI)</b>	Indicates that the far-end device has a problem with the signal which is receiving from the local device. Perform hard plug loopback tests to verify the ISDN controller hardware is OK. Check or replace cables to isolate cabling problems.
<b>Transmit Remote Alarm Indication (txRAI)</b>	Indicates that the ISDN interface has a problem with the signal it receives from the far-end equipment. Check the settings at the remote end to ensure that they match the local port settings.
<b>Transmit Alarm Indication Signal (txAIS)</b>	Indicates that the ISDN controller is shut down. Issue the <b>show controller e1   t1 {num}</b> EXEC command to verify that the ISDN controller is up. If the ISDN controller is down, issue the <b>no shutdown</b> interface subcommand to bring it up.

- Below describes the various error events that occur on ISDN E1/T1 lines along with the troubleshooting guidelines for them:

<b>Linecode Violations</b>	Indicates that either a Bipolar Violation (BPV) or excessive zero error event is present. BPVs are inserted upon the synchronization of circuits with B8ZS linecoding. Linecode errors occur when BPVs that are used for synchronization are received. Excessive zero errors occur when 8 or more 0s in sequence are received on circuits with AMI linecoding. These errors normally occur due to an AMI/B8ZS linecoding misconfiguration on the intermediate devices along the transmission path.
<b>Pathcode Violations</b>	Examples are frame synchronization errors for SF and cyclic redundancy check (CRC) errors for ESF. Both linecode and pathcode violations are often present at the same time; hence always verify the linecoding configuration. Note that some errors can occur due to impulse noise, in which the errors might appear only a few times a day and the impact or interrupt is minimal.
<b>Slip Seconds</b>	Indicates a clocking problem. The ISDN network would provide the clocking in which the CPE must synchronize. Look for <code>clock Source is Line</code> in the output of the <b>show controller e1   t1 {num}</b> EXEC command to verify that the clocking is derived from the Telco.
<b>Loss of Frame</b>	Triggered when the receiver detects multiple framing-bit errors. The data is considered garbage as the synchronization between both ends is invalid. When this state is triggered, the framer starts searching for a correct framing pattern. The LoS state ends when reframe occurs. Ensure the framing format is configured correctly. Change the framing format with the <b>framing {esf   sf   crc4   no-crc4}</b> interface subcommand.

<b>Loss of Signal</b>	Triggered upon observing 175 +/- 75 contiguous pulse positions with no pulses of either positive or negative polarity for T1 lines; or when more than 10 consecutive 0s are detected for E1 lines.
<b>Code Violation Error</b>	The CRC value in the received frame does not match with the corresponding locally calculated CRC value.
<b>Bipolar Violation</b>	Occurs when 2 mark signals (1s) occur in sequence with the same polarity (when not part of a B8ZS substitution). T1 signaling specifies that each mark must be the opposite polarity of the one preceding it. It also includes other error patterns such as 8 or more consecutive 0s and incorrect parity.
<b>Errored Second</b>	A second with one or more Code Violation Error or Loss of Frame events have occurred. The presence of Bipolar Violations also triggers an Errored Second. Some errored seconds may occur on a normal circuit.
<b>Severely Errored Second</b>	A second with 320 or more Code Violation Error or Loss of Frame events have occurred. Also known as Extreme Errored Second (EES).
<b>Unavailable Second</b>	A second that the CSU is in the Unavailable Signal state, including the initial 10 seconds to enter the state, but excluding the 10 seconds to exit the state.

## Troubleshooting ISDN Layer 2

- Understanding the output messages of the **debug isdn q921** command is vital in troubleshooting ISDN Layer 2. Firstly, issue the **debug isdn q921** to enable the ISDN Layer 2 debugging which provides the details of ISDN Layer 2 transactions between the router and the Telco ISDN switch. Subsequently, issue the **clear interface bri {num}** or **clear interface serial {num:15}** privileged command to reset the ISDN interface which forces the router to renegotiate ISDN Layer 2 information with the Telco ISDN switch.
- Below shows the sample output messages of the **debug isdn q921** command:

```
Router#debug isdn q921
debug isdn q921 is          ON.
Router#
Router#sh isdn status
Global ISDN Switchtype = basic-net3
ISDN BRI0/0 interface
    dsl 0, interface ISDN Switchtype = basic-net3
    Layer 1 Status:
        ACTIVE
    Layer 2 Status:
        TEI = 76, Ces = 1, SAPI = 0, State = TEI_ASSIGNED
    Layer 3 Status:
        0 Active Layer 3 Call(s)
    Active dsl 0 CCBs = 0
    The Free Channel Mask: 0x80000003
    Total Allocated ISDN CCBs = 0
Router#
Router#clear int bri0/0
Router#
23:20:28: ISDN BR0/0 Q921: User TX -> IDREQ ri=24872 ai=127
23:20:28: ISDN BR0/0 Q921: User RX <- IDASSN ri=24872 ai=75
23:20:28: ISDN BR0/0 Q921: L2_EstablishDataLink: sending SABME
23:20:28: ISDN BR0/0 Q921: User TX -> SABMEp sapi=0 tei=75
23:20:28: ISDN BR0/0 Q921: User RX <- UAf sapi=0 tei=75
23:20:28: %ISDN-6-LAYER2UP: Layer 2 for Interface BR0/0, TEI 75 changed to up
23:20:38: ISDN BR0/0 Q921: User RX <- RRp sapi=0 tei=75 nr=0
23:20:38: ISDN BR0/0 Q921: User TX -> RRf sapi=0 t    Total Allocated ISDN CCBs = 0
23:20:48: ISDN BR0/0 Q921: User RX <- RRp sapi=0 tei=75 nr=0
23:20:48: ISDN BR0/0 Q921: User TX -> RRp sapi=0 tei=75 nr=0
23:20:48: ISDN BR0/0 Q921: User TX -> RRf sapi=0 tei=75 nr=0
23:20:48: ISDN BR0/0 Q921: User RX <- RRf sapi=0 tei=75 nr=0
Router#
Router#sh isdn status
Global ISDN Switchtype = basic-net3
ISDN BRI0/0 interface
    dsl 0, interface ISDN Switchtype = basic-net3
    Layer 1 Status:
        ACTIVE
    Layer 2 Status:
        TEI = 75, Ces = 1, SAPI = 0, State = MULTIPLE_FRAME_ESTABLISHED
    Layer 3 Status:
        0 Active Layer 3 Call(s)
    Active dsl 0 CCBs = 0
    The Free Channel Mask: 0x80000003
    Total Allocated ISDN CCBs = 0
Router#
```

```

--- output omitted ---
23:23:05: ISDN BR0/0 Q921: User RX <- DISCp sapi=0 tei=75
23:23:05: %ISDN-6-LAYER2DOWN: Layer 2 for Interface BR0/0, TEI 75 changed to
down
23:23:05: ISDN BR0/0 Q921: User TX -> UAf sapi=0 tei=75
Router#sh isdn status
Global ISDN Switchtype = basic-net3
ISDN BRI0/0 interface
    dsl 0, interface ISDN Switchtype = basic-net3
    Layer 1 Status:
        ACTIVE
    Layer 2 Status:
        TEI = 75, Ces = 1, SAPI = 0, State = TEI_ASSIGNED
    Layer 3 Status:
        0 Active Layer 3 Call(s)
    Active dsl 0 CCBs = 0
    The Free Channel Mask: 0x80000003
    Total Allocated ISDN CCBs = 0

```

IDREQ	Identity Request transmitted from the router to the ISDN switch requesting a Terminal Endpoint Identifier (TEI). All IDREQ messages have an AI value of 127 which indicates that the ISDN switch can assign any available TEI value.
IDASSN	Identity Assigned message with the TEI value of 75 assigned by ISDN switch is received from the ISDN switch.
SABMEp	Request the connection to be in Multiple Frame Established state. <b>SABME</b> → Asynchronous Balanced Mode Extended.
UAf	Unnumbered Acknowledgment (UA) of the SABME message. The ISDN Layer 2 is now in Multiple Frame Established state.

- An operational and functioning ISDN circuit (Multiple Frame Established) should have periodic exchanges of RRp sapi = 0 and RRf sapi = 0 messages between the router and ISDN switch. The interval between **Receiver Ready poll** (RRp) and **Receiver Ready final** (RRf) messages is usually 10 or 30 seconds.  
**Note:** TX -> indicates that a message is generated by the router while RX <- indicates that a message is received by the router.
- Below shows the **debug isdn q921** messages originated from the ISDN switch which indicate various ISDN Layer 2 problems:

Message	Explanation
ID-Denied	The ISDN switch cannot assign the requested TEI. If this message has an AI value of 127, it indicates that the ISDN switch has no TEI available.
IDREM	The ISDN switch has removed the TEI from the connection. The router must terminate all existing communication using the particular TEI.
DISC	The sending side of the DISConnect message has terminated the operation of ISDN Layer 3 for the connection. It may be unnumbered acknowledged by the other side. The router should then send a SABME message to reestablish the link.
DM	Indicates the Acknowledged Disconnect mode. The ISDN switch does not wish to enter the Multiple Frame Established state, and hence the router will remain in Layer 2 TEI_ASSIGNED state. The router will continuously send SABME messages until the ISDN switch responds with a UA instead of DM.
FRMR	The Frame Reject Response indicates an error that cannot be recovered by retransmission. The router will initiate a Layer 2 reset and transmit a SABME message to request to enter into Multiple Frame Established state.

### Troubleshooting ISDN Layer 3

- Only proceed to this section after ISDN Layers 1 and 2 on both ends of a circuit are verified OK.
- ISDN call failures could be due to any of the following:
  - i) Dial-on-Demand Routing (DDR) related issues.
  - ii) ISDN Layer 1, 2, or 3 problems.
  - iii) Point-to-Point Protocol (PPP), including authentication, Link Control Protocol (LCP), or IP Control Protocol (IPCP) related issues.
- Figure A6-28 illustrates common Q.931 transactions during a successful ISDN call setup.

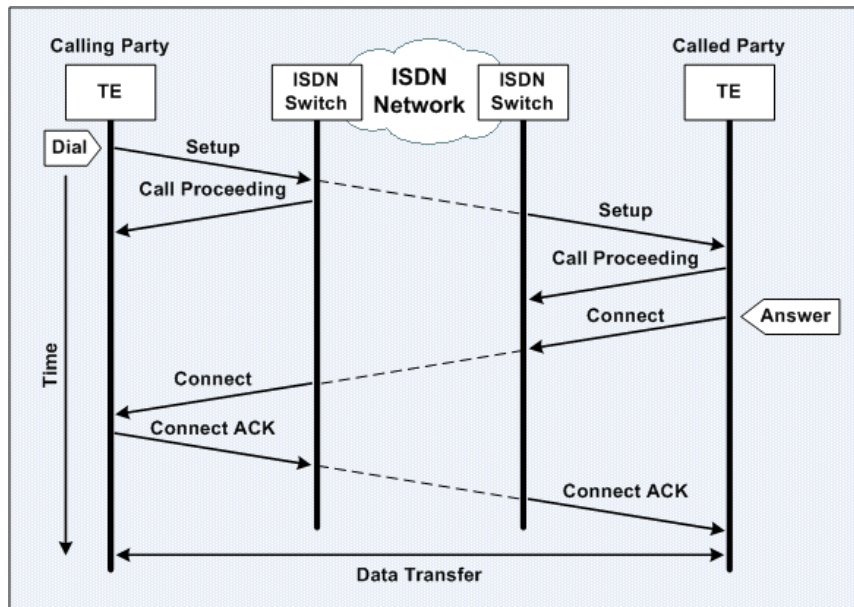


Figure A6-28: ISDN Call Setup Sequence

- Pay attention to the direction of the output messages from the **debug isdn q931** privileged command. The `TX ->` indicates that a message is generated by the router while the `RX <-` indicates that a message is received by the router.
- The source of a problem can be identified by following the direction of a particular message and the response. Ex: If the local router unexpectedly receives a `RELEASE` message from the ISDN switch, then it will reset its end of the call as well. This indicates that there is an issue in the ISDN switch or the remote router.
- Verify that the message received or sent is the expected one. Ex: If the called party receives a `SETUP` message but responds with a `DISCONNECT` message instead of a `CONNECT` message, then troubleshoot the called router and not the ISDN network.

- Below lists all the possible Q.931 messages during call establishment and termination:

Message	Description
<b>SETUP</b>	Indicates that a device would like to establish an ISDN Layer 3 call.
<b>CALL_PROC</b>	The SETUP message is received and is being processed by the network and/or the remote device.
<b>ALERTING</b>	Informs the network that the router is alerting the user, which would normally be the case for a telephone and the alert would ring the handset. This message is normally associated with equipment using a handset, eg: an ISDN telephone or TA and is not usually seen for data calls.
<b>CONNECT</b>	Call is accepted.
<b>CONNECT_ACK</b>	<b>Connect Acknowledge.</b> The device has received the CONNECT message. Higher layer protocols (eg: PPP) should now begin the negotiation process.
<b>DISCONNECT</b>	The router initiated a DISCONNECT message, which usually indicates that an operational ISDN call is disconnected due to some higher layer issues (eg: DDR, PPP, etc). The 3-way Disconnect Handshake will be accompanied by a RELEASE message, a Disconnect Cause Code, and a RELEASE_COMP message.
<b>RELEASE</b>	Acknowledges the DISCONNECT message and continues the circuit termination process. The RELEASE message is sent between the DISCONNECT and RELEASE_COMP messages.
<b>RELEASE_COMP</b>	<b>Release Complete.</b> The call termination is completed. This message is often seen during a normal call termination initiated by one of the routers; in response to a SETUP message from the calling party when there is a mismatch of bearer capability between the ISDN switch and the router; or due to protocol error if the coding of the SETUP message does not comply with the Q.931 standard or the configuration of the ISDN switch.

- If the calling router does not send a SETUP message to the ISDN switch, the problem is likely related to ISDN Layer 1, 2, or DDR issues, and is not ISDN Layer 3 related. Perform the following troubleshooting tasks on the calling and called routers:
  - Verify that the ISDN switch types on both routers are configured correctly.
  - Verify that the ISDN Layers 1 and 2 on both routers are functioning.
  - Perform ISDN loopback test calls on both routers to verify they are able to initiate and accept calls.
  - Verify the ISDN network of the called side is functioning by making a test call to the called router with a regular analog phone, in which the router should receive a SETUP message, although the call would eventually fail as it is not an ISDN call.
  - Verify that the calling router has a route to the destination with the **show ip route EXEC** command.
  - Verify that the interesting traffic on the calling router is identified correctly.
  - Verify that the appropriate **dialer string** or **dialer map** interface subcommand on the calling router refers to the correct number to the called router.
  - Check the DDR configuration and use the **debug dialer** privileged command to verify that the calling router is able to initiate calls.
  - If the called router does not send a CONNECT message, check if the call is rejected due to the misconfiguration of the **isdn caller {incoming-number}** interface subcommand.
  - Contact the Telco and determine whether the long distance call service is activated.

## ISDN Loopback Test Call

- In a loopback call, a router dials the ISDN number of its own BRI. The call is proceed to the ISDN switch, which then being switched to the 2nd BRI channel. The call is seen by the router as an incoming call on the 2nd channel. Therefore, the router both sends and receives an ISDN call on different B channels and act as both the **called** and **calling** routers.
- Loopback call tests the ability of a router to initiate and accept ISDN calls. It can also provide an indication that the physical links to the ISDN switch along with the ISDN switch are functioning.

## Q.931 ISDN Layer 3 Disconnect Cause Codes

- This section explains how to interpret the ISDN disconnect cause codes which appear in the output of the **debug isdn q931** privileged command to indicate the reason for ISDN call disconnection or failure.

- The ISDN Layer 3 disconnect cause code the has the following format:

Cause i = 0xAABBCC - reason.

<b>AA</b>	Cause Code Origination Point
<b>BB</b>	Disconnect Cause Code
<b>CC</b>	Optional Diagnostics Field

- Below lists all the Cause Code Origination Points:

<b>80</b>	Router
<b>81</b>	Private network near the local router (possibly a local private branch exchange – PBX)
<b>82</b>	Public network near the local router (local Telco switch)
<b>83</b>	Transit network in the ISDN cloud
<b>84</b>	Public network near the remote router (remote Telco switch)
<b>85</b>	Private network near the remote router (possibly a remote private branch exchange – PBX)
<b>87</b>	International network
<b>8A</b>	Network beyond the internetworking point

- Below lists all the standard ITU-T Q.931 Disconnect Cause Codes:

<b>80</b>	<b>Normal Disconnect.</b> The ISDN call disconnects normally.
<b>81</b>	<b>Unallocated or unassigned number.</b> The ISDN switch receives the ISDN number in the correct format. However, the number is not in the ISDN switch's routing table or it has no path across the ISDN network. Check the routing table to see whether the number is available; and check the correct digits were dialed and it is a valid number.
<b>82</b>	<b>No route to specified transit network.</b> The ISDN switch receives a request to route the call through an unrecognized intermediate (or transit) network – the ISDN switch has no route to the transit network. This problem can be due to either the transit network does not exist, or the transit network exists but does not serve the ISDN switch. This cause is supported on a network-dependent basis.
<b>83</b>	<b>No route to destination.</b> The dialed number is in the ISDN switch's routing plan, but there is no physical route to the destination and hence the called party is not reachable. This problem can be due to either the D channel is down at either end, or the span or WAN is not connected correctly. This cause is supported on a network-dependent basis.

84	<b>Send special information tone.</b> The called party is not reachable and a special information tone should be returned from the calling party. Check the dialing number and verify whether any prefix is required to access the network, eg: a 9 might need to be dialed for outbound calls through a PBX. Contact the Telco or PBX administrator for details.
85	<b>Misdialed trunk prefix.</b> There is erroneous inclusion of a trunk prefix in the dialed number. Check the dialing number and verify whether any prefix is required to access the network, eg: a 9 might need to be dialed for outbound calls through a PBX. Contact the Telco or PBX administrator for details.
86	<b>Channel unacceptable.</b> The service quality of the specified channel is insufficient to accept the connection. The call fails due to the channel is not acceptable (unusable) for the call. If a PBX is in used, check the configuration of the PBX. If a PRI is in used, find out how many channels are provided by the Telco.
87	<b>Call awarded and being delivered in an established channel.</b> The called party receives an incoming call, but the call is being connected to a channel already established for similar calls (eg: packet-mode virtual calls).
88	<b>Preemption.</b> The call is being preempted or blocked. Sometimes calls are blocked if another call has a higher priority than the current call, which is common with voice calls. Wait and call again later. If a local or remote PBX is in used, check the configuration of the PBX. Contact the Telco if the problem persists.
89	<b>Preemption, circuit reserved for re-use.</b> The call is being cleared as one of the routers involved in the call has requested to terminate and clear the call.
90	<b>Normal call clearing.</b> The call disconnects and normal call clearing occurs due to one of the routers involved in the call requested to terminate and clear the call. This is one of the most common cause codes and is received for many reasons. Most of the time, the ISDN network is not the source of this cause. If a call fails with this cause, it is most likely fails with a higher layer protocol, eg: PPP, authentication, or idle timeout related issues. Verify the router configuration to resolve such problems. Additionally, if callback is configured on the local router, the remote router will disconnect the calls, generates this code, and calls the local router back.
91	<b>User busy.</b> The called party acknowledges the connection request. However, it is unable to accept the call due to all B channels are in use (the dialed number is busy). The routers are compatible with the call in this situation. <b>Note:</b> If there are multiple ISDN circuits, the Telco can configure them in a hunt-group, which calls are switched to next available circuit.
92	<b>No user response.</b> The called party does not respond to the call establishment message within a prescribed duration, or it does not wish to answer the call. The called party must respond with either an alert or connect indication according to ITU-T Q.931, when either timer T303 or T310 expires.
93	<b>No answer from user.</b> The called party is alerted to the connection request but does not respond with a connect indication to establish the connection within a prescribed duration. This cause is not necessary generated by Q.931 procedures; it may be generated by internal network timers. The problem is at the remote router.
94	<b>Subscriber absent.</b> The remote router is unavailable or disconnected from the ISDN network. Contact the person responsible for the remote router.
95	<b>Call rejected.</b> The remote router rejects the call due to an unknown reason. <b>Note:</b> The remote router is able to accept the call as it is able to respond with this cause, it is neither busy nor incompatible. This cause may also be generated by the ISDN network indicating that the call was cleared due to a supplementary service constraint.

96	<b>Number changed.</b> The called number is no longer assigned to any device. The new called number may optionally be included in the diagnostic field. If the network does not support this cause, the caller receives disconnect cause code 81 – unallocated or unassigned number.
97	<b>Redirection to new destination.</b> The call is routed to a different number. Check the called number and verify the configuration of the PBX if PBX is in used.
99	<b>Exchange routing error.</b> The call cannot be successfully routed to the remote router. Check the called number and verify the configuration of the PBX if a PBX is in used.
9A	<b>Non-selected user clearing.</b> The remote router is able to accept the call. However, it rejects the call due to the call is not assigned to any user.
9B	<b>Destination out of order.</b> The remote router is not reachable as there is a problem sending the signaling message to the remote router. This condition is often temporary, but can also last for an extended period in some cases. Possible causes are ISDN layer 1 or 2 fails at the remote end, or the remote router is powered off.
9C	<b>Invalid number format.</b> The call fails due to the called number is not in a valid format or is incomplete. This can also happen when the local router calling out using network-specific ISDN Type of Number (TON) when it should be calling out using National or Unknown for the ISDN Type of Number (TON). Verify whether the format of the number is correct, which includes any necessary digits for a PBX or long distance.
9D	<b>Facility rejected.</b> The ISDN network is unable to provide a requested supplementary service.
9E	<b>Response to STATUS ENQUIRY.</b> This cause code is included in a STATUS message when the STATUS message is generated upon the receipt of a STATUS ENQUIRY message.
9F	<b>Normal, unspecified.</b> This is a very common cause code and happens when the network is not able to determine the next course of action for the call. No action is required.
A1	<b>Circuit out of order.</b> The call fails due to some problems in the ISDN network.
A2	<b>No channel available.</b> The call fails due to no B channel is available to answer the call.
A3	<b>Destination unattainable.</b> The destination is not reachable through the ISDN network. Contact the Telco.
A4	<b>Out of order.</b> Some parts of the ISDN network necessary to route the call is out of order. The destination is not reachable due to a network malfunction. This condition can last for a relatively long period. An immediate attempt to reconnect will probably fail. Try to use a Presubscribed Inter-exchange Carrier (PIC) if a long distance carrier is being used. A PIC allows us to verify whether the problem lies with the long distance carrier.
A6	<b>Network out of order.</b> The destination is not reachable due to a network malfunction. This condition can last for a relatively long period. An immediate attempt to reconnect will probably fail. Try to use a Presubscribed Inter-exchange Carrier (PIC) if a long distance carrier is being used. A PIC allows us to verify whether the problem lies with the long distance carrier.
A7	<b>Permanent frame mode connection out of service.</b> This cause code is included in a STATUS message to indicate that a permanently established frame mode connection is terminated. Contact the Telco if the problem persists.
A8	<b>Permanent frame mode connection operational.</b> This cause code is included in a STATUS message to indicate that a permanently established frame mode connection is fully operational again and capable of carrying user information after a termination. The connection is probably terminated by a faulty equipment previously.
A9	<b>Temporary failure.</b> The call is disconnected due to a network malfunction. This condition is not likely to last a long period of time; another call attempt can be tried immediately. Contact the Telco if the problem persists.

<b>AA</b>	<b>Switching equipment congestion.</b> The destination is unreachable due to a temporary high traffic load on the network switching equipment. Try again later.
<b>AB</b>	<b>Access information discarded.</b> The ISDN network is unable to provide the requested access information. <b>Note:</b> The diagnostics field may indicate the particular type of discarded access information, eg: user-to-user information, low layer compatibility, high layer compatibility, and sub-address.
<b>AC</b>	<b>Requested channel not available.</b> The remote router is unable to provide the requested channel due to an unknown reason. This may happen when there is a glare condition, in which both sides are selected top-down or bottom-up channel hunting. This problem is usually temporary.
<b>AF</b>	<b>Resources unavailable, unspecified.</b> This cause code is used to report a resource unavailable event only when no other cause in the resource unavailable class applies. This problem is usually temporary.
<b>B1</b>	<b>Quality of server (QoS) unavailable.</b> The ISDN network unable to provide the requested Quality of Service as defined in Recommendation X.213. This problem can occur due to a subscription problem, or the ISDN network does not support throughput or transit delay.
<b>B2</b>	<b>Requested facility not subscribed.</b> The local router is not authorized to use a requested supplementary service which is implemented by the ISDN switch. The administrator has probably not completed the necessary administrative arrangements with the service provider. The ISDN network can also return this cause code if a call is made without supplying the SPIDs, or the SPIDs are entered wrongly. Ensure that the SPIDs are correct, or contact the Telco for verification.
<b>B4</b>	<b>Outgoing calls barred.</b> There are some restrictions on outgoing calls. The ISDN network does not allow the local router to make outgoing calls.
<b>B5</b>	<b>Outgoing calls barred within CUG.</b> There are some restrictions on outgoing calls. The ISDN network does not allow the local router to make outgoing calls although the calling party is a member of the CUG for the outgoing CUG call. Outgoing calls are not allowed for this member of the CUG. Contact the Telco.
<b>B6</b>	<b>Incoming calls barred.</b> The ISDN network does not allow the local router to receive calls. Contact the Telco.
<b>B7</b>	<b>Incoming calls barred within CUG.</b> The ISDN network does not allow the local router to receive calls although the called party is a member of the CUG for the incoming CUG call. Incoming calls are not allowed for this member of the CUG. Contact the Telco.
<b>B9</b>	<b>Bearer capability not authorized.</b> The local router requests a bearer capability that the ISDN switch implements, however the local router is not authorized to use the capability. A subscription problem usually causes this problem.
<b>BA</b>	<b>Bearer capability not presently available.</b> The ISDN network normally provides the requested bearer capability. However, the capability is unavailable at the particular moment. Normally caused by a temporary ISDN network problem or a subscription problem.
<b>BE</b>	<b>Inconsistency in designated outgoing access information and subscriber class.</b> There is an inconsistency in the designated outgoing access information and subscriber class.
<b>BF</b>	<b>Service or option not available, unspecified.</b> This cause code is used to report a service or option is unavailable event only when no other cause in the service or option not available class applies. Normally caused by a subscription problem.
<b>C1</b>	<b>Bearer capability not implemented.</b> The ISDN network is unable to provide the requested bearer capability, eg: requesting 64kb data when only speech is supported. Contact the Telco for further troubleshooting.

C2	<b>Channel type not implemented.</b> The equipment sending this cause does not support the requested channel type.
C5	<b>Requested facility not implemented.</b> The equipment sending this cause does not support the requested supplementary service.
C6	<b>Only restricted digital info bearer capability available.</b> The equipment sending this cause does not support and hence unable to provide unrestricted digital information bearer service (64kb). It only supports the restricted version of the requested bearer capability.
CF	<b>Service or option not implemented, unspecified.</b> This cause code is used to report a service or option not implemented event only when no other cause in the service or option not implemented class applies. Normally caused by a subscription problem.
D1	<b>Invalid call reference value.</b> The equipment sending this cause receives a call with a call reference that is not currently in use or assigned on the user-network interface. Ex: The call that is being referenced by the call reference value does not exist on the system.
D2	<b>Identified channel does not exist.</b> The equipment sending this cause receives a request to use an inactive channel for a call. Ex: When a PRI is subscribed to use channels 1 to 12 and the router or the ISDN network attempts to assign a call to channels 13 to 25.
D3	<b>Suspended call exists, but call id does not.</b> The ISDN switch receives a call resume request which contains a Call Identify (ID) information element that differs from the ID in use for any currently suspended call.
D4	<b>Call id in use.</b> The ISDN switch receives a call suspend request which contains a call ID that is already in use for a suspended call which the call can be resumed instead of suspended again.
D5	<b>No call suspended.</b> The ISDN switch receives a call resume request when there is no currently suspended call. This transient error can be resolved through successive call retries.
D6	<b>Call with requested call id has been cleared.</b> The ISDN switch receives a call resume request which contains a call ID that indicates a suspended call. However, either a network timeout or the remote router has cleared the call while the call was suspended.
D7	<b>User not member of CUG.</b> The called party for the incoming CUG call is not a member of the specified CUG; or the calling party is an ordinary subscriber calling a CUG subscriber.
D8	<b>Incompatible destination.</b> Indicates an attempt to connect to non-ISDN device (eg: an analog line), in which the equipment sending this cause is not capable of answering a particular type of call – it is unable to accommodate a low layer compatibility, high layer compatibility, or other compatibility attributes. This cause code often appears when the calling device dials a wrong number and reaches a non-ISDN device; a data call is made to a voice number; a voice call is made to a number that only supports data; calling a restricted line in unrestricted mode; or calling a POTS phone using unrestricted mode. Check that the correct number is dialed. If the dialed number is correct, contact the Telco to verify that the ISDN switch configuration.
DA	<b>Non-existent CUG.</b> The specified CUG does not exist. Contact the Telco if the problem persists.
DB	<b>Invalid transit network selection.</b> The ISDN switch is requested to route the call through an unrecognized intermediate network – it receives a transit network identification which is in an incorrect format as defined in Annex C of ITU-T Q.931.
DF	<b>Invalid message, unspecified.</b> This cause code is used to report an invalid message event only when no other cause in the invalid message class applies. This problem usually occurs due to a D channel error. Contact the Telco if the problem occurs systematically.

<b>E0</b>	<b>Mandatory IE missing.</b> The equipment sending this cause receives a message is missing an information element that is necessary for it to process the message. This problem usually occurs due to a D channel error. Upgrade the Cisco IOS software on the router to resolve this problem. Contact the Telco if the problem occurs systematically.
<b>E1</b>	<b>Message type not implemented.</b> The equipment sending this cause receives an unrecognized message due to either the message type is invalid, or it does not support or implement the message type. The problem usually occurs due to the D channel of the local router or the configuration of the remote router.
<b>E2</b>	<b>Message not compatible with call state or not implemented.</b> The equipment sending this cause receives a message that is not permissible in the call state according to the procedures, or it receives a STATUS message which indicates an incompatible call state. This problem usually occurs due to a D channel error. Contact the Telco if the problem occurs.
<b>E3</b>	<b>IE not implemented.</b> The equipment sending this cause receives a message that contains an unrecognized information element which it does not support or implement. However, the message does not need to contain the information element in order for the equipment sending this cause to process the message. This problem usually occurs due to a D channel error. Contact the Telco if the problem occurs.
<b>E4</b>	<b>Invalid IE contents.</b> The ISDN switch receives a message that contains invalid contents in the information element – the information element is implemented, but one or more of the fields in the information element are coded differently. This cause is usually followed by the information element that is causing the problem.
<b>E5</b>	<b>Message not compatible with call state.</b> The ISDN switch receives a message that does not correspond to the current call state for the call.
<b>E6</b>	<b>Recovery on timer expired.</b> Occurs when ISDN messages do not arrive in specified time according to the Q.931 specification. This cause if sometimes followed by the expired timer. Wait and try again later. Contact the Telco if the problem persists.
<b>E7</b>	<b>Parameter not implemented.</b> The equipment sending this cause receives a message which contains an unrecognized parameter which it does not support or implement. Contact the Telco if the problem occurs.
<b>EE</b>	<b>Message with unrecognized parameter discarded.</b> The equipment sending this cause has discarded a received message which contains an unrecognized parameter.
<b>EF</b>	<b>Protocol error, unspecified.</b> This cause code is used to report a protocol error event only when no other cause in the protocol error class applies. This problem usually occurs due to a D channel error.
<b>FF</b>	<b>Interworking, unspecified.</b> The ISDN network is interworking with another network which does not provide the next course of action. The precise problem is unknown.

**Note: Closed User Group (CUG)** is a facility in X.25 and ISDN networks that allows a called number to be available only to a limited number of users in a virtual private network.

**IE** – Information Element.