

This chapter covers the following key topics:

- **General Switch and Layer 2 Security**—This section discusses some of the basic steps you can take to make Layer 2 environments and switches more secure.
- **Port Security**—This section discusses how to restrict access on a port basis.
- **IP Permit Lists**—This section talks about using IP permit lists to restrict access to the switch for administrative purposes.
- **Protocol Filtering and Controlling LAN Floods**—This section talks about controlling floods on LANs.
- **Private VLANs on Catalyst 6000**—This section deals with setting up private VLANs on Catalyst 6000 switches to provide Layer 2 isolation to connected devices.
- **Port Authentication and Access Control Using the IEEE 802.1x Standard**—This section talks about how the 802.1x protocol can be used to improve security in a switched environment by providing access control on devices attaching to various ports.

## Secure LAN Switching

---

In order to provide comprehensive security on a network, it is important take the concept of security to the last step and ensure that the Layer 2 devices such as the switches that manage the LANs are also operating in a secure manner.

This chapter focuses on the Cisco Catalyst 5000/5500 series switches. We will discuss private VLANs in the context of the 6000 series switches. Generally, similar concepts can be implemented in other types of switches (such as the 1900, 2900, 3000, and 4000 series switches) as well.

Security on the LAN is important because some security threats can be initiated on Layer 2 rather than at Layer 3 and above. An example of one such attack is one in which a compromised server on a DMZ LAN is used to connect to another server on the same segment despite access control lists on the firewall connected on the DMZ. Because the connection occurs at Layer 2, without suitable measures to restrict traffic on this layer, this type of access attempt cannot be blocked.

### General Switch and Layer 2 Security

Some of the basic rules to keep in mind when setting up a secure Layer 2 switching environment are as follows:

- VLANs should be set up in ways that clearly separate the network's various logical components from each other. VLANs lend themselves to providing segregation between logical workgroups. This is a first step toward segregating portions of the network needing more security from portions needing lesser security. It is important to have a good understanding of what VLANs are. VLANs are a logical grouping of devices that might or might not be physically located close to each other.
- If some ports are not being used, it is prudent to turn them off as well as place them in a special VLAN used to collect unused ports. This VLAN should have no Layer 3 access.

- Although devices on a particular VLAN cannot access devices on another VLAN unless specific mechanisms for doing so (such as trunking or a device routing between the VLANs) are set up, VLANs should not be used as the sole mechanism for providing security to a particular group of devices on a VLAN. VLAN protocols are not constructed with security as the primary motivator behind them. The protocols that are used to establish VLANs can be compromised rather easily from a security perspective and allow loopholes into the network. As such, other mechanisms such as those discussed next should be used to secure them.
- Because VLANs are not a security feature, devices at different security levels should be isolated on separate Layer 2 devices. For example, having the same switch chassis on both the inside and outside of a firewall is not recommended. Two separate switches should be used for the secure and insecure sides of the firewall.
- Unless it is critical, Layer 3 connectivity such as Telnets and HTTP connections to a Layer 2 switch should be restricted and very limited.
- It is important to make sure that trunking does not become a security risk in the switching environment. Trunks should not use port numbers that belong to a VLAN that is in use anywhere on the switched network. This can erroneously allow packets from the trunk port to reach other ports located in the same VLAN. Ports that do not require trunking should have trunking disabled. An attacker can use trunking to hop from one VLAN to another. The attacker can do this by pretending to be another switch with ISL or 802.1q signaling along with Dynamic Trunking Protocol (DTP). This allows the attacker's machine to become a part of all the VLANs on the switch being attacked. It is generally a good idea to set DTP on all ports not being used for trunking. It's also a good idea to use dedicated VLAN IDs for all trunks rather than using VLAN IDs that are also being used for nontrunking ports. This can allow an attacker to make itself part of a trunking VLAN rather easily and then use trunking to hop onto other VLANs as well.

Generally, it is difficult to protect against attacks launched from hosts sitting on a LAN. These hosts are often considered trusted entities. As such, if one of these hosts is used to launch an attack, it becomes difficult to stop it. Therefore, it is important to make sure that access to the LAN is secured and is provided only to trusted people.

Some of the features we will discuss in the upcoming sections show you ways to further secure the switching environment.

The discussion in this chapter revolves around the use of Catalyst 5xxx and 6xxx switches. The same principles can be applied to setting up security on other types of switches.

## Port Security

*Port security* is a mechanism available on the Catalyst switches to restrict the MAC addresses that can connect via a particular port of the switch. This feature allows a specific MAC address or a range of MAC addresses to be defined and specified for a particular port. A port set up for port security only allows machines with a MAC address belonging to the range configured on it to connect to the LAN. The port compares the MAC address of any frame arriving on it with the MAC addresses configured in its allowed list. If the address matches, it allows the packet to go through, assuming that all other requirements are met. However, if the MAC address does not belong to the configured list, the port can either simply drop the packet (restrictive mode) or shut itself down for a configurable amount of time. This feature also lets you specify the number of MAC addresses that can connect to a certain port.

## MAC Address Floods and Port Security

Port security is especially useful in the face of MAC address flooding attacks. In these attacks, an attacker tries to fill up a switch's CAM tables by sending a large number of frames to it with source MAC addresses that the switch is unaware of at that time. The switch learns about these MAC addresses and puts them in its CAM table, thinking that these MAC addresses actually exist on the port on which it is receiving them. In reality, this port is under the attacker's control and a machine connected to this port is being used to send frames with spoofed MAC addresses to the switch. If the attacker keeps sending these frames in a large-enough quantity, and the switch continues to learn of them, eventually the switch's CAM table becomes filled with entries for these bogus MAC addresses mapped to the compromised port.

Under normal operations, when a machine receiving a frame responds to it, the switch learns that the MAC address associated with that machine sits on the port on which it has received the response frame. It puts this mapping in its CAM table, allowing it to send any future frames destined for this MAC address directly to this port rather than flood all the ports on the VLAN. However, in a situation where the CAM table is filled up, the switch is unable to create this CAM entry. At this point, when the switch receives a legitimate frame for which it does not know which port to forward the frame to, the switch floods all the connected ports belonging to the VLAN on which it has received the frame. The switch continues to flood the frames with destination addresses that do not have an entry in the CAM tables to all the ports on the VLAN associated with the port it is receiving the frame on. This causes two main problems:

- Network traffic increases significantly due to the flooding done by the switch. This can result in a denial of service (DoS) for legitimate users of the switched network.
- The attacker can receive frames that are being flooded by the switch and use the information contained in them for various types of attacks.

Figure 5-1 shows how MAC address flooding can cause CAM overflow and subsequent DoS and traffic analysis attacks.

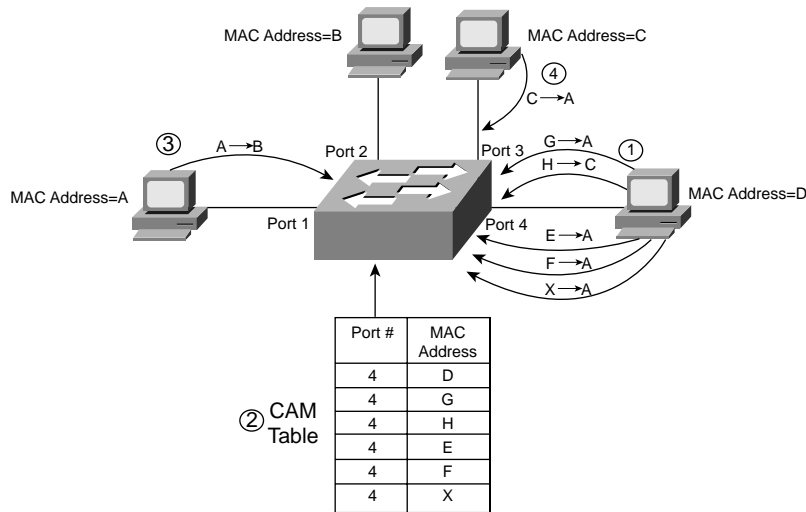
**Figure 5-1** MAC Address Flooding Causing CAM Overflow and Subsequent DoS and Traffic Analysis Attacks

Figure 5-1 shows a series of steps that take place to orchestrate a MAC address flooding attack. Given below is the list of steps that takes place as shown in the Figure 5-1:

- Step 1** A compromised machine is attached to port 4. Frames sourced from fictitious MAC address denoted by G, H, E and F etc. are sent on the port 4. The actual MAC address of the compromised machine is denoted by D.
- Step 2** Due to the flooding of frames on port 4, the CAM table of the switch fills up and it is unable to 'learn' any more MAC address and port mappings.
- Step 3** A host situated on port 1 with a MAC address denoted by A, sends a frame sourced from the MAC address A to MAC address B. The switch is unable to learn and associate port 1 with the MAC address A since its CAM table is full.
- Step 4** Host on port 3 with a MAC address denoted by C sends a frame to MAC address A. Since the switch does not have an entry in its CAM table for A, it floods the frame to all its ports in that VLAN. This results in flooding causing DOS as well as an opportunity for traffic analysis by the attacker who receives the flooded frames on port 4 as well.

The danger of attacking a switch by flooding the CAM table can be avoided either by hard-coding the MAC addresses that are allowed to connect on a port or by limiting the number of hosts that are allowed to connect on a port. Both these features are part of the port security feature set on Cisco switches.

The switch configuration shown in Example 5-1 enables port security on port 1 of module 2. The MAC address 00-90-2b-03-34-08 is configured as the only MAC that is allowed to access the LAN using this port. In case of a violation, meaning another MAC address trying to use this port, the port shuts down for 600 minutes, or 10 hours.

**Example 5-1** *Port Security Using Various Parameters Enabled on a Switch*

```
Console> (enable) set port security 2/1 enable
Console> (enable) set port security 2/1 enable 00-90-2b-03-34-08
Console> (enable) set port security 2/1 shutdown 600
```

Example 5-2 shows how you can restrict the number of MAC addresses a switch learns on a port.

**Example 5-2** *Restricting the Number of MAC Addresses a Switch Learns on a Port*

```
Console> (enable) set port security 3/2 maximum 20
```

In Example 5-2, the number of MAC addresses that the switch can learn on the port is restricted to a maximum of 20. By setting this threshold to 1, the first MAC address that the switch learns on a port can be made the only address it allows on that particular port.

As you can see, setting up port security, especially to allow only certain MAC addresses to connect to the various ports, can be very administratively resource-consuming. However, it is still a useful safeguard against the type of attack discussed in this section.

## IP Permit Lists

IP permit lists are used to restrict Telnet, SSH, HTTP, and SNMP traffic from entering the switch. This feature allows IP addresses to be specified that are allowed to send these kinds of traffic to the switch.

The configuration shown in Example 5-3 on a switch enables the **ip permit list** feature and then restricts Telnet access to the switch from the 172.16.0.0/16 subnet and SNMP access from 172.20.52.2 only. The host, 172.20.52.3, is allowed to have both types of access to the switch.

**Example 5-3** *Setting Up IP Permit Lists on a Switch to Control Various Types of Access*

```
Console> (enable) set ip permit enable
Console> (enable) set ip permit 172.16.0.0 255.255.0.0 telnet
Console> (enable) set ip permit 172.20.52.2 255.255.255.255 snmp
Console> (enable) set ip permit 172.20.52.3 all
```

IP permit lists are an essential feature to configure on a switch in situations where Layer 3 access to the switch is needed. As stated earlier, Layer 3 access to a switch should remain fairly limited and controlled.

## Protocol Filtering and Controlling LAN Floods

Attackers can cause broadcast floods to disrupt communications over the LAN. You saw an example of this in the section “MAC Address Floods and Port Security.” Therefore, it is important to control flooding on the switches. There are two main ways to do this:

- Set up threshold limits for broadcast/multicast traffic on ports
- Use protocol filtering to limit broadcasts/multicasts for certain protocols

Catalyst switches allow thresholds for broadcast traffic to be set up on a per-port basis. These thresholds can be set up either in terms of bandwidth consumed by broadcasts on a port or in terms of the number of broadcast packets being sent across a port. It is best to use the first method in most cases, because it is done in hardware and also because variable-length packets can render the second method meaningless.

The following command sets the threshold for broadcast and multicast packets on ports 1 to 6 of module 2 at 75%. This implies that as soon as 75% bandwidth of the port on a per-second basis is consumed by broadcast/multicast traffic, all additional broadcast/multicast traffic for that 1-second period is dropped.

```
Console> (enable) set port broadcast 2/1-6 75%
```

Protocol filtering provides another very useful mechanism for isolating and controlling environments that are susceptible to flooding attacks. Using the protocol-filtering feature on Catalyst switches, you can define protocol groups. Each group has certain protocols associated with it. It also has a set of ports that belong to it. Only the broadcast or multicast traffic for the protocols associated with a group is allowed to be sent to the ports that belong to that group. You should realize that although VLANs also create broadcast domains for the ports associated with them, protocol-filtering groups allow these domains to be created based on various protocols as well. Using protocol filtering, ports that have hosts on them that do not need to participate in the broadcast traffic for a certain protocol can be made part of a group that does not allow broadcast traffic for that protocol.

With the Catalyst 5000 family of switches, packets are classified into the following protocol groups:

- IP (ip)
- IPX (ipx)
- AppleTalk, DECnet, and Banyan VINES (group)
- Packets not belonging to any of these protocols

A port can be configured to belong to one or more of these four groups and be in any one of the following states for that group:

- On
- Off
- Auto

If the configuration is set to on, the port receives all the flood (broadcast/multicast traffic) traffic for that protocol. If the configuration is set to off, the port does not receive any flood traffic for that protocol. If the configuration is set to auto, a port becomes a member of the protocol group only after the device connected to the port transmits packets of the specific protocol group. The switch detects the traffic, adds the port to the protocol group, and begins forwarding flood traffic for that protocol group to that port. Autoconfigured ports are removed from the protocol group if the attached device does not transmit packets for that protocol within 60 minutes. Ports are also removed from the protocol group when the supervisor engine detects that the link on the port is down. Example 5-4 shows how port filtering can be configured on a switch.

**Example 5-4** *Port Filtering Configured on a Switch*

```
Console> (enable) set port protocol 2/1-6 ip on
Console> (enable) set port protocol 2/1-6 ipx off
Console> (enable) set port protocol 2/1-6 group auto
```

The configuration shown in Example 5-4 sets up ports 1 to 6 on module 2 for protocol filtering. Ports 1 to 6 have only the IP group set to on and IPX is turned off. Therefore, these ports do not receive any broadcast traffic for IPX. However, ports 1 to 6 are also set up to be in group “group” in the auto state. This means that if the switch detects a host on any of these ports sending out AppleTalk, DECnet, and Banyan VINES traffic, it enables these ports for broadcast traffic for these protocols as well.

Port filtering can be used in conjunction with the bandwidth or packet threshold-based flood control discussed earlier in this section.

## Private VLANs on the Catalyst 6000

The Catalyst 6000 product line has introduced some enhancements to the switching arena for security purposes. We will discuss some of these in this section and see how they can be a useful security element in Layer 2 design.

A normal VLAN does not allow devices connected to it to be segregated from each other on Layer 2. This means that if a device on a VLAN becomes compromised, other devices on the same VLAN can also be attacked from that compromised device.

Private VLANs allow restrictions to be placed on the Layer 2 traffic on a VLAN.

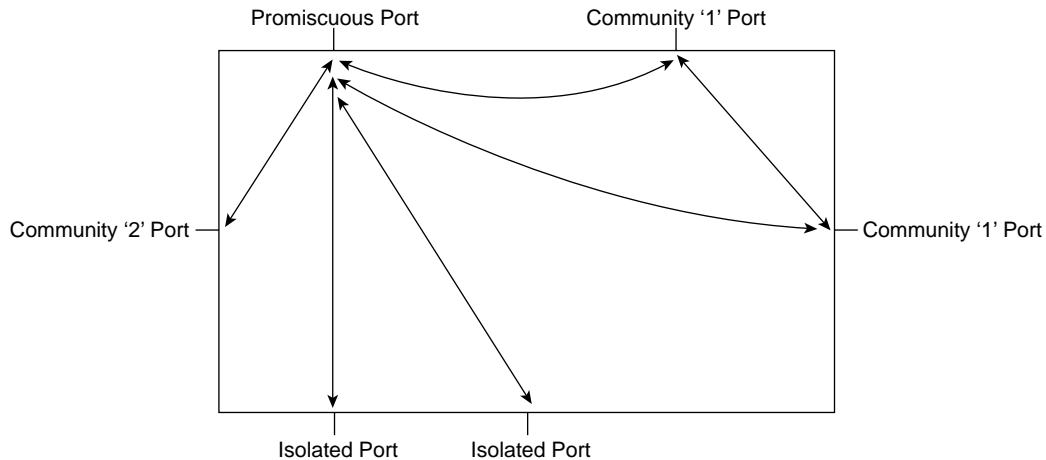
There are three types of private VLAN ports:

- **Promiscuous ports**—Communicates with all other private VLAN ports. This is generally the port used to communicate with the router/gateway on a segment.
- **Isolated ports**—Has complete Layer 2 isolation from other ports within the same private VLAN, with the exception of the promiscuous port.
- **Community ports**—Communicate among themselves and with their promiscuous ports. These ports are isolated at Layer 2 from all other ports in other communities or isolated ports within their private VLAN.

In essence, isolating a port stops any other machine on the same logical or physical segment as the machines on the isolated port from sending any traffic to this port.

When a port is isolated, all machines connected to the network using this port are provided complete isolation from traffic in all other ports, except for the promiscuous port. This means that no machines located on any of the other ports on the switch can send any traffic to the machines located on the isolated VLAN port. It is similar to placing two ports in two separate VLANs. The isolated ports communicate with the rest of the world through the promiscuous VLAN port, which can send traffic to and receive traffic from the isolated VLAN ports. Figure 5-2 gives a graphical view of which ports can communicate with which other ports in a private VLAN setup.

**Figure 5-2** Private VLANs



Unless Explicitly Indicated by an Arrow, Communication Between Ports Is Not Allowed

In order to set up private VLANs, a primary VLAN is created that contains the promiscuous ports, and then the secondary VLANs are created that contain the isolated or community ports. These two components are then bound together. Example 5-5 shows how a private VLAN is set up on a Catalyst 6000 switch.

**Example 5-5** *A Private VLAN Set Up on a Catalyst 6000 Switch*

```
!The configuration line below creates the primary VLAN and gives it the name 7.
6500 (enable) set vlan 7 pvlan-type primary
!The line below defines the secondary VLAN, 42, and configures it to be an isolated
!VLAN.
6500 (enable) set vlan 42 pvlan-type isolated
!The line below binds the primary and secondary VLANs (7 and 42, respectively) and
!defines the ports that belong to the isolated VLAN 42.
6500 (enable) set pvlan 7 42 3/9-10
!The line below defines the first port (3/31) that is used as the promiscuous port
!in this setup.
6500 (enable) set pvlan mapping 7 42 3/31
!The line below defines the second port (3/32) that is used as a promiscuous port.
6500 (enable) set pvlan mapping 7 42 3/32
```

Example 5-5 shows how to create a private VLAN and set it up so that ports 3/31 and 3/32 are the promiscuous ports and ports 3/9 and 3/10 are the isolated ports. Note that the isolated ports can communicate with the promiscuous ports and vice versa, but they cannot communicate with each other.

## ARP Spoofing, Sticky ARP, and Private VLANs

A security problem that private VLANs resolve is that of ARP spoofing. Network devices often send out what is known as a *gratuitous ARP* or *courtesy ARP* to let other machines on their broadcast domain know their IP address and the corresponding MAC address. This generally happens at bootup, but it can also occur at regular intervals after that. An attacker who has gained access to a compromised machine on the LAN can force the compromised machine to send out gratuitous ARPs for IP addresses that do not belong to it. This results in the rest of the machines sending their frames intended for those IP addresses to the compromised machine. This type of attack can have two consequences:

- It can result in a DoS attack if the attacker spoofs the IP address/MAC address of the network's default gateway in its gratuitous ARPs. This causes all the machines on the broadcast domain to send the traffic destined for the default gateway to the compromised host, which in turn can simply drop this traffic, resulting in a DoS.
- The attacker can analyze the traffic being sent to it and use the information found therein for various malicious activities.

Private VLANs offer protection from this type of attack by providing isolation between various ports on a VLAN. This stops an attacker from receiving traffic from the machines, sitting on all the other ports on a switch, on a port that has a compromised machine sitting on it.

Another feature, known as *sticky ARP*, which is available in conjunction with private VLANs, can also help mitigate these types of attacks. The sticky ARP feature makes sure that the ARP entries that are learned by the switch on the private VLANs do not age out and cannot be changed. Suppose an attacker somehow compromises and takes control of a machine on a private VLAN. He tries to do ARP spoofing by sending out gratuitous ARPs, announcing the machine as the owner of a certain MAC address/IP address mapping that it does not own. The switch ignores these ARPs and doesn't update its CAM tables to reflect these mappings. If there is a genuine need to change a port's MAC address, the administrator must do so manually.

## Port Authentication and Access Control Using the IEEE 802.1x Standard

802.1x is the standard developed by IEEE to provide a mechanism for authentication to occur for devices that connect to various Layer 2 devices such as switches using IEEE 802 LAN infrastructures (such as Token Ring and Ethernet).

The primary idea behind the standard is devices that need to access the LAN need to be authenticated and authorized before they can connect to the physical or logical port of the switch that is responsible for creating the LAN environment. In the case of Ethernet and Token Ring, the ports are physical entities that a device plugs into. However, in the case of setups such as the IEEE 802.11b wireless setup, the ports are logical entities known as *associations*. In either case, the standard's primary goal is to allow for controlled access to the LAN environment.

### 802.1x Entities

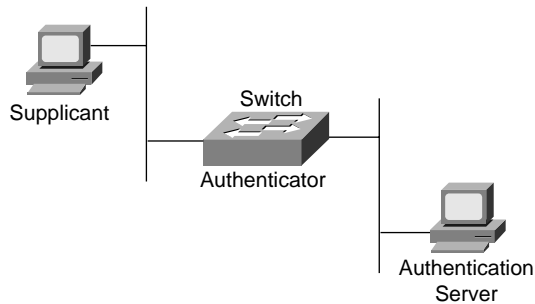
The 802.1x standard defines the following three main entities that take part in the access control method set up in this standard:

- **Supplicant**—This device needs to access the LAN. An example is a laptop that needs to connect to a LAN.
- **Authenticator**—This device is responsible for initiating the authentication process and then acting as a relay between the actual authentication server and the supplicant. This device is generally also the device that is responsible for the overall workings of the LAN. An example of this type of device is a Catalyst 6000 switch to which various supplicants can connect and be authenticated and authorized via the 802.1x standard before being allowed to use the ports on the switch for data traffic.

- **Authentication server**—This device is responsible for doing the actual authentication and authorization on behalf of the authenticator. This device contains profile information for all the users of the network in a database format. It can use that information to authenticate and authorize users to connect to the ports on the authenticator. An example of an authentication server is the Cisco Secure ACS.

Figure 5-3 shows these three entities in a graphical format.

**Figure 5-3** *The Three Entities Defined in the 802.1x Standard*



In addition to these three main entities, the 802.1 defines some other entities as well. One of these is the Port Access Entity (PAE). The PAE is essentially responsible for maintaining the functionality of the 802.1x standard on the authenticator or the supplicant or both. It can be viewed as the daemon that is responsible for the functioning of the 802.1x standard. For our purposes, we will assume that this entity is transparent to the network administrator as we talk about various aspects of this standard.

---

**NOTE** The authenticator and the authentication server can be colocated on the same system.

---

## 802.1x Communications

In order for the 802.1x standard to function, communication needs to occur between the three entities just defined. 802.1x protocol uses an RFC standard known as Extensible Authentication Protocol (EAP) to facilitate this communication. The authentication data between the three entities is exchanged using EAP packets that are carried either in EAPOL frames (between the supplicant and the authenticator, as discussed later) or in TACACS+, RADIUS, or some other such protocol's packets (between the authenticator and the authenticating server). The following sections look at each of these pieces and discuss how they come together to form the 802.1x communication infrastructure.

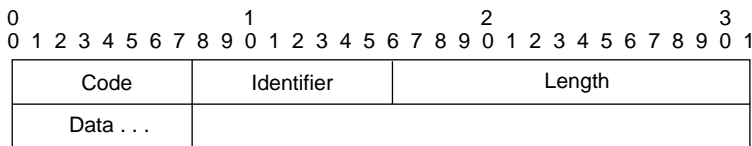
## EAP

EAP is a fairly flexible protocol. It was originally designed to carry only PPP authentication parameters, but it also can be used by other protocols such as 802.1x for their authentication needs.

EAP can carry authentication data between two entities that want to set up authenticated communications between themselves. It supports a variety of authentication mechanisms, including one-time password, MD5 hashed username and password, and transport layer security (discussed later). EAP, using the packets described in the next section, allows the authenticator, the supplicant, and the authentication server to exchange the information they need to exchange to authenticate the supplicant.

RFC 2284 defines the EAP packet format as shown in Figure 5-4.

**Figure 5-4** *EAP Packet Format*



RFC 2284 defines the various fields in this packet as follows:

- **Code**—The Code field is one octet. It identifies the type of EAP packet. EAP codes are assigned as follows:
  - **1**—Request
  - **2**—Response
  - **3**—Success
  - **4**—Failure
- **Identifier**—The Identifier field is one octet. It aids in matching responses with requests.
- **Length**—The Length field is two octets. It indicates the length of the EAP packet, including the Code, Identifier, Length, and Data fields. Octets outside the range of the Length field should be treated as data link layer padding and should be ignored on receipt.
- **Data**—The Data field is zero or more octets. The format of the Data field is determined by the Code field.

EAP makes use of four types of messages:

- Request
- Response
- Success
- Failure

The request and response messages are used for the bulk of the messaging used in EAP. These messages can be further classified into the following six most important types of messages (note that more message types are being defined all the time for various purposes):

- Identity
- Notification
- NAK
- MD5-Challenge
- One-time password
- Transport-Level Security (TLS)

The *identity message* is generally sent by the authenticator in the 802.1x scheme of things to the supplicant. The purpose of this message is to ask the supplicant to send its identity information (such as the username) to the authenticator. The supplicant responds with an EAP-Response message of the same type containing the requested information.

The *notification message* is used to display a message on the supplicant machines that the user can see. This message is sent by the authenticator. This message could be a notification that a password is about to expire or some other notification of this type.

The *NAK message* is generally sent by the supplicant to the authenticator when the authentication mechanism offered by the authenticator is unacceptable to the supplicant.

The *MD-5 challenge messages* are sent in the form of EAP Request and Response messages to allow the supplicant to authenticate itself using a method of authentication that is analogous to PPP CHAP protocol using MD5 hashing. The Request contains a “challenge” message to the peer. A Response *must* be sent in reply to the Request. The Response may be either Type 4 (MD5-Challenge) or Type 3 (NAK). The NAK reply indicates the peer’s desired authentication mechanism type.

*One-time password messages* implement a one-time password authentication system as defined in RFC 1938. The Request message contains an OTP challenge. It is responded to in the form of a Response message of the same one-time password type or the NAK type message. If NAK is used to respond to the challenge, the NAK reply indicates the peer’s desired authentication mechanism type.

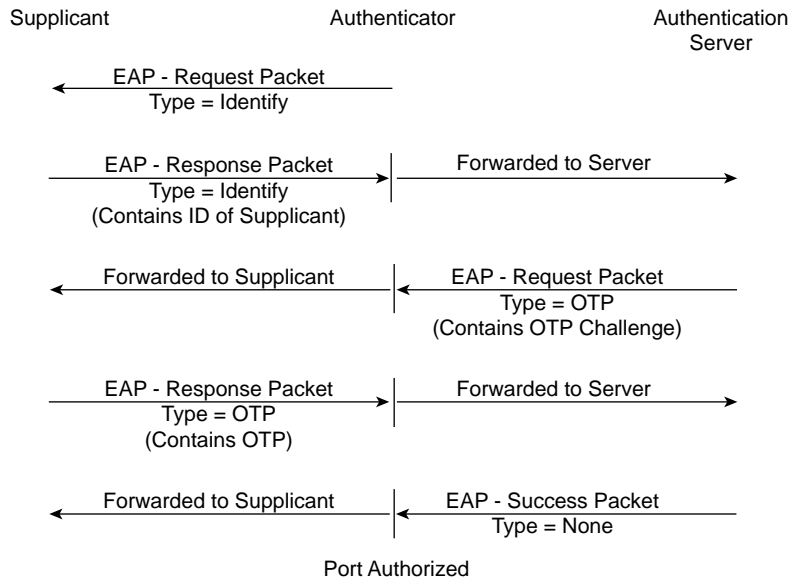
*Transport-Layer Security (TLS) messages* allow a supplicant to authenticate itself with the authentication server after doing a key exchange and integrity-protected ciphersuite negotiation. The workings of EAP-TLS are described in RFC 2716. EAP-TLS allows the supplicant and the authentication server to use digital certificates to authenticate each other. This is a more scalable mechanism for doing authentication than using a database of usernames and passwords. Also, EAP-TLS is a mutual authentication method, which means that both the client and the server prove their identities.

The next section covers how these messages are used to perform the 802.1x functionality.

### Using EAP as the Underlying Communications Mechanism in 802.1x

As stated earlier, the 802.1x standard uses EAP as the underlying mechanism for carrying authentication information back and forth between the three entities—the supplicant, the authenticator, and the authenticating server. In general, the authenticator sends the first EAP-Request message of the identity type to the supplicant to ask for the supplicant's identity. (The supplicant can also start this process if it so desires.) The supplicant returns an EAP-Response message containing its identity, such as its username. The authenticator receives this message and forwards it to the authentication server. From this point onward, the authenticator becomes a pass-through device. It forwards the EAP messages received from the authentication server to the supplicant and the EAP messages received from the supplicant to the authentication server. However, the authenticator remains knowledgeable of the exchange and waits for the success or failure of the exchange to enable the port to which the supplicant wants to connect (in case authentication success occurs). The authentication server, in response to the EAP-Response message containing the supplicant's identity, sends the supplicant an EAP message corresponding to the type of authentication it wants to do with this particular supplicant. The supplicant responds to the authentication server (via the authenticator). It either sends an EAP Response frame of the same type that the authentication server has indicated with a response to what the authentication server has asked for (for example, a one-time password generated on the supplicant) or sends back an EAP-NAK if it does not want to use this particular method of authentication. If NAK is used, the NAK reply indicates the peer's desired authentication mechanism type. EAP messages are sent back and forth until the authentication server sends either an EAP-Success or EAP-Failure message to the supplicant. At this point, the authenticator, upon seeing this message (EAP-Success), opens the port, on which the supplicant has been sending EAP traffic, for normal data activity as well. Figure 5-5 shows an example of EAP exchange involving a successful OTP authentication.

**Figure 5-5** *EAP Exchange Involving Successful OTP Authentication*



## EAPOL

We have looked at EAP, which is the underlying 802.1x protocol. But we have not looked at how EAP messages are actually framed and transported from the supplicant to the authenticator. The 802.1x defines an encapsulating/framing standard to allow communication between the supplicant and the authenticator to take place. This encapsulation mechanism is known as *EAP Over LANs (EAPOL)*. EAPOL encapsulation is defined separately for both the Token Ring and Ethernet environments. EAPOL allows the EAP messages to be encapsulated using the EAPOL frames for transport between the supplicant and the authenticator. As soon as these frames reach the authenticator, it strips off the EAPOL headers, puts the EAP packet in a RADIUS or TACACS+ (or some other similar protocol) packet, and sends it to the authenticating server. Figure 5-6 shows the relationship between the supplicant and the authenticator using EAPOL.

**Figure 5-6** *Relationship Between the Supplicant and the Authenticator Using EAPOL*



EAPOL uses the same basic frame format as the Ethernet of the Token Ring frames. Figure 5-7 shows the frame format for EAPOL using Ethernet 802.3 (Ethernet).

**Figure 5-7** *Frame Format for EAPOL Using Ethernet 802.3*

PAE Ethernet Type = 88-8E
Protocol Version = 0000 0001
Packet Type EAP Packet or EAP-OL Start or EAPOL - Logoff or EAPOL-Key or EAPOL-Encapsulated-ASF-Alert
Packet Body Length = Length of Body Field in Octets
Packet Body (Only present if EAP-Packet, EAPOL-key or EAPOL-Encapsulated- ASF-Alert)

In summary, communications between the supplicant and the authenticator occur using EAP packets encapsulated using EAPOL. The authenticator then encapsulates the EAP frames in TACACS+ or RADIUS packets and sends them to the authentication server.

The various fields in the EAPOL are defined in the 802.1x standard as follows:

- **PAE Ethernet Type**—This field is two octets in length. It contains the Ethernet Type value assigned for use by the PAE.
- **Protocol Version**—This field is one octet in length, taken to represent an unsigned binary number. Its value identifies the version of EAPOL protocol supported by the sender of the EAPOL frame. An implementation conforming to this specification uses the value 0000 0001 in this field.
- **Packet Type**—This field is one octet in length, taken to represent an unsigned binary number. Its value determines the type of packet being transmitted. The following types are defined:
  - **EAP-Packet**—A value of 0000 0000 indicates that the frame carries an EAP packet.
  - **EAPOL-Start**—A value of 0000 0001 indicates that the frame is an EAPOL-Start frame.

- **EAPOL-Logoff**—A value of 0000 0010 indicates that the frame is an explicit EAPOL-Logoff request frame.
- **EAPOL-Key**—A value of 0000 0011 indicates that the frame is an EAPOL-Key frame.
- **EAPOL-Encapsulated-ASF-Alert**—A value of 0000 0100 indicates that the frame carries an EAPOL-Encapsulated-ASF-Alert.

All other possible values of this field are unused, because they are reserved for use in potential future extensions to this protocol.

The EAPOL-Encapsulated-ASF-Alert packet type is provided for use by the Alerting Standards Forum (ASF) as a means of allowing alerts (that is, specific SNMP traps) to be forwarded through a port that is in the Unauthorized state. All EAPOL frames with this packet type that are received on the uncontrolled port are passed to the protocol entity responsible for handling ASF alerts for validation and further processing in accordance with the relevant ASF protocol specifications.

- **Packet Body Length**—This field is two octets in length, taken to represent an unsigned binary number. The value of this field defines the length in octets of the Packet Body field. A value of 0 indicates that there is no Packet Body field.
- **Packet Body**—The Packet Body field is present if the Packet Type contains the value EAP-Packet, EAPOL-Key, or EAPOL-Encapsulated-ASF-Alert. For all other values of Packet Type, this field is not present.

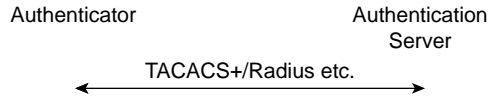
In a frame carrying a Packet Type of EAP-Packet, this field contains an EAP packet. Exactly one EAP packet is encapsulated.

In a frame carrying a Packet Type of EAPOL-Key, this field contains a key descriptor. Exactly one key descriptor is encapsulated.

In a frame carrying a Packet Type of EAPOL-Encapsulated-ASF-Alert, this field contains an ASF alert. Exactly one ASF alert frame is encapsulated.

## The Role of RADIUS, TACACS+, and Other Authentication Protocols in 802.1x

The communication between the authenticator and the authentication server takes place via a protocol such as RADIUS or TACACS+. The normal functionality of these protocols is evoked so that communication between these two entities can take place. The EAP packets that are sent to the authenticator using EAPOL encapsulation are decapsulated from EAPOL and are then put into TACACS+ or RADIUS packets and are sent to the authentication server. RADIUS is generally the preferred back-end protocol, because it has EAP encapsulation extensions built into it. Figure 5-8 shows the use of TACACS+/RADIUS protocols in the 802.1x setup.

**Figure 5-8** TACACS+/RADIUS Protocols in the 802.1x Setup

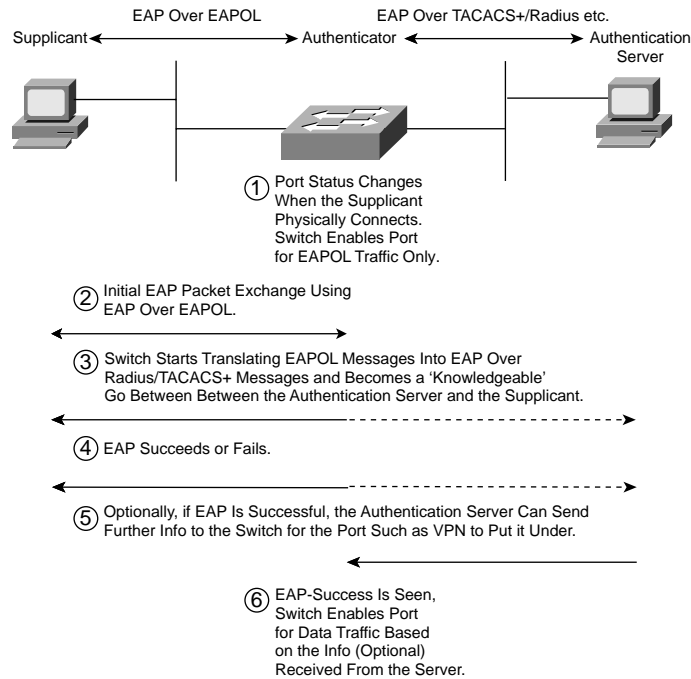
## 802.1x Functionality

This section puts together all the pieces of the 802.1x protocol discussed in the preceding sections and summarizes how 802.1x provides port authentication to supplicants.

The 802.1x functionality is based on a series of exchanges between the supplicant, authenticator, and authentication server. The authenticator plays an important role in these exchanges because it not only acts as a go-between for the supplicant and the authenticating server but also is responsible for enabling the port to which the supplicant is trying to connect for normal data traffic if the authentication is indeed successful.

The authentication process starts with the supplicant trying to connect to one of the ports on the authenticator. At this point, the port is open only for EAPOL traffic. The authenticator sees the port's operational state change to enable due to the supplicant's connecting to it and requests authentication from the supplicant. This is done by sending an EAP-Request/Identity frame to the supplicant. This message is sent encapsulated in an EAPOL frame. The supplicant responds by sending back an EAP-Response/Identity frame containing information about its identity, such as username/password. This message is also sent encapsulated in an EAPOL frame. The authenticator decapsulates the EAP message from the EAPOL frame and repackages this EAP frame in a RADIUS or TACACS+ packet and forwards it to the authentication server. The authentication server, upon receiving this packet, responds with an EAP-Response message that is based on the authentication method that the authentication server wants to use for this particular supplicant. This message is encapsulated in a TACACS+ or RADIUS packet. Upon receiving this message, the authenticator strips off the TACACS+/RADIUS header, encapsulates the EAP message in an EAPOL frame, and forwards it to the supplicant. This back-and-forth EAP exchange between the supplicant and the authentication server via the authenticator continues until the authentication either succeeds or fails, as indicated by an EAP-Success or EAP-Failure message sent by the authentication server to the supplicant. Upon seeing an EAP-Success message, the authenticator enables the port on which the supplicant is connected for normal data traffic. In addition to enabling the port for this type of traffic, the authenticator can place the port in a specific VLAN based on the information it receives from the authentication server. Figure 5-9 shows the overall 802.1x architecture and flow using EAP over EAPOL and EAP over TACACS+/RADIUS.

**Figure 5-9** Overall 802.1x Architecture and Flow Using EAP Over EAPOL and EAP Over TACACS+/RADIUS



## Setting Up Catalyst 6000 to Do Port Authentication Using 802.1x

Most of the newer versions of the Cisco switches support 802.1x port authentication. The Catalyst 6000 supports 802.1x starting in version 6.2. Refer to the documentation for the other types of switches for support information.

Example 5-6 shows how a Catalyst 6000 can be set up to do 802.1x authentication for devices connecting to its ports. RADIUS is used in this example as the authentication protocol. TACACS+ or Kerberos can also be used on the Catalyst to perform this role. However, RADIUS is generally the recommended protocol because it has built-in extensions that support EAP frames.

**Example 5-6** *Setting Up a Catalyst 6000 to do 802.1x Authentication for Devices Connecting to Its Ports*

```
# RADIUS configuration
set radius server 10.1.1.1 auth-port 1812 primary
set radius key test

# Global 802.1x configuration

!The following command enables 802.1x globally on the Catalyst 6000

set dot1x system-auth-control enable

!The following command sets the idle time between authentication attempts.

set dot1x quiet-period 60

!The following command sets how long the authenticator waits before retransmitting
!if it does not receive an EAP-Response/Identity type to the EAP-Request/Identity
!type message it sent to the supplicant.

set dot1x tx-period 5

!This is how long the switch waits for a response from the supplicant to any
!EAP-Request message before retransmitting

set dot1x supp-timeout 5

!Below is a timeout similar to the one before this. The only difference is that in
!this case this timeout is for the switch waiting on the authentication server
!rather than the supplicant to respond to its EAP messages.

set dot1x server-timeout 10

!The command below defines the maximum number of times the authenticator sends an
!EAP-Request to the supplicant before stopping

set dot1x max-req 3

!The time shown below is the time after which the switch reauthenticates the
!supplicants connected to its various authorized ports.

set dot1x re-authperiod 3600

# Port Level 802.1x configuration

!The command below enables 802.1x on port 5/1. Note that having done this line of
!configuration, you must initialize the port by issuing the command set port dot1x
!mod/port initialize

set port dot1x 5/1 port-control auto
```

**Example 5-6** *Setting Up a Catalyst 6000 to do 802.1x Authentication for Devices Connecting to Its Ports*

```

!The command below forces the state of port 5/3 to force-authorized permanently
!until this command is changed for this port. This is equivalent to disabling
!802.1x on this port. This command forces the port to become authorized for any
!and all hosts connecting to it. The network administrator wants to have no
!security enabled in port 5/3 in this example.

set port dot1x 5/3 port-control force-authorized

!The command below forces port 5/48 into the force-unauthorized state permanently
!until this command is changed for this port. This ensures that no one can
!authenticate and use this port. The network administrator in this example does
!not want anyone to use port 5/48.

set port dot1x 5/48 port-control force-unauthorized

!The command below sets up the port to allow multiple hosts to connect to it after
!the first host connecting to it has passed through the normal 802.1x
!authentication process. Note the security risk involved in using this feature.

set port dot1x 5/1 multiple-host enable

!The command below ensures that for the ports specified only a single host is
!allowed to connect to a port after authentication. Hosts are identified using
!their MAC addresses.

set port dot1x 5/2,5/4-47 multiple-host disable

!The command below ensures that the supplicants connected to the specified ports
!are prompted for reauthentication after the globally defined re-authperiod
!expires.

set port dot1x 5/2,5/4-47 re-authentication enable

!Automatic reauthentication has been disabled using the command below. Manual
!reauthentication can still be initiated by the network administrator.

set port dot1x 5/1 re-authentication disable

```

Please note that in Example 5-6 the RADIUS server also returns the following RFC 2868 attributes to the Catalyst 6000:

- [64]—Tunnel-Type = VLAN
- [65]—Tunnel-Medium-Type = 802
- [81]—Tunnel-Private-Group-Id = VLAN NAME

Attribute [64] must contain the value VLAN (type 13). Attribute [65] must contain the value 802 (type 6). Attribute [81] specifies the VLAN name in which the successfully authenticated 802.1x supplicant should be put.

## Summary

This chapter talked about issues related to security on Layer 2 networks. General recommendations to secure a LAN environment were discussed, as were various features available on the Catalyst switches to make the switching environment more secure. We looked at how some of the more well-known attacks can be orchestrated at Layer 2 of the OSI model. We also looked at ways to control these attacks using various features available in the Catalyst line of switches. We looked at methods of securing the ports on a switch, including the emerging 802.1x standard. Although the discussion focused on the high-end Catalyst switches—namely, the 5500 and 6500 switches—the concepts are generic and can be carried over to other switches as well with the appropriate commands and configurations. The discussion in this chapter is an important building block for your understanding of the overall security infrastructure in place in modern networks.

## Review Questions

- 1 Why is it imprudent to rely on VLANs to provide isolation and security?
- 2 What does port security do?
- 3 Which protocols are covered under IP permit lists?
- 4 What is an isolated VLAN port?
- 5 What is a promiscuous VLAN port?
- 6 What is the purpose of the 802.1x standard?
- 7 What is EAP?
- 8 What is the purpose of the EAPOL standard?
- 9 What is the purpose of EAP-Request Identity type messages?
- 10 What is the EAP-Response NAK message used for?

