CHAPTER 5

Inside the Cisco Centri Firewall

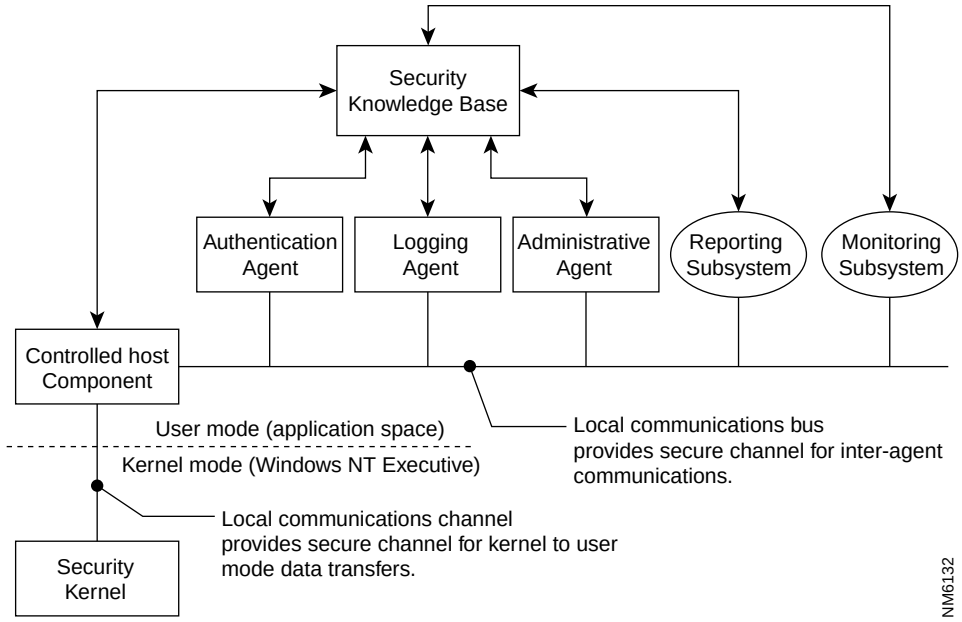
Introduction

With its visionary design, Cisco Centri Firewall defines the new standard in firewall architectures. It combines all of the advantages of the old-style firewall architectures with the speed of kernel-level evaluation and "session awareness" at every protocol layer to define the most secure and extensible firewall architecture available.

The Cisco Centri Firewall architecture propagates its central tenets of security, usability, extensibility, and high performance throughout. As you will see, we have incorporated many concepts that are new to firewall architectures to improve the performance and scalability of your security solution. We have also built carefully planned security mechanisms throughout the system.

Because security is an unbounded problem, we designed the Cisco Centri Firewall using an autonomous agent-based architecture. This agent-based architecture, shown in Figure 5-1, divides system tasks into well-organized, independent programs that provide additional features, such as per-site tuning, and that also improve upon the fault tolerance of the system. Agents interact with one another through well-defined interfaces, coming together to form a complex system from a set of specialized autonomous agents. Because each agent performs simple, specialized tasks, we can ensure its individual quality and reliability more easily.

Figure 5-1 Agent-Based Architecture



Note The Cisco Centri Firewall architecture uses multiple task-oriented agents to provide an extensible, secure system that organizes complex processes into small manageable tasks. This reduction in complexity provides a more reliable system that is highly extensible and simplifies the fundamental security objectives of minimization and validation.

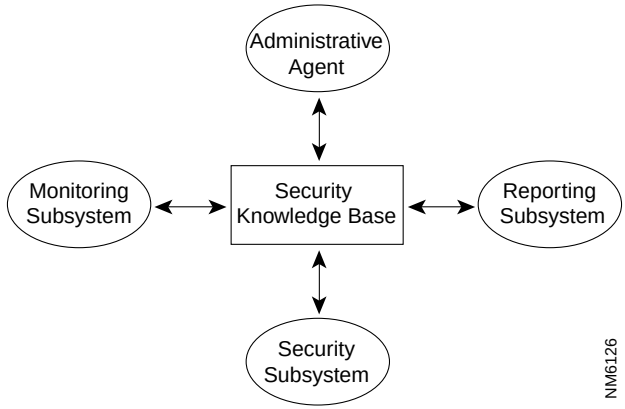
This agent-based architecture allows us to add new agents quickly to address security concerns as they arise in the future; it also enables us to improve upon existing features without adversely affecting the stability and reliability within other parts of the system. The primary elements within this architecture are as follows:

- Security Subsystem
- Security Knowledge Base

- Reporting Subsystem
- Monitoring Subsystem
- Administrative Agent\

Figure 5-2 depicts the relationships among these primary elements.

Figure 5-2 High-Level View of the Cisco Centri Firewall Architecture



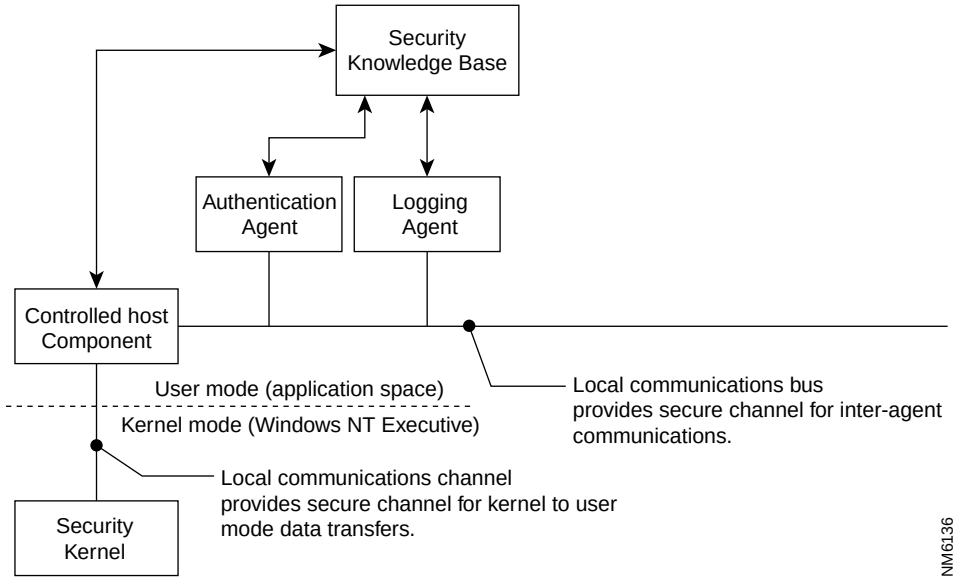
Note Cisco Centri Firewall comprises four major architectural elements. Each of these elements performs a specific collection of tasks for the security system.

The following sections explain the role that each element plays within the security system and identify the various agents composing these elements.

Security Subsystem

As shown in Figure 5-3, the Security Subsystem comprises four major components: the Security Kernel, the Controlled Host Component and Communications Channels, the Logging Agent, and the Authentication Agent. Each of these components performs a security-related task within the Cisco Centri Firewall.

Figure 5-3 The Security Subsystem



Note The Security Subsystem comprises five primary components: the Security Kernel, the Controlled Host Components and secure communication channels, the Logging agent, and the Authentication agent.

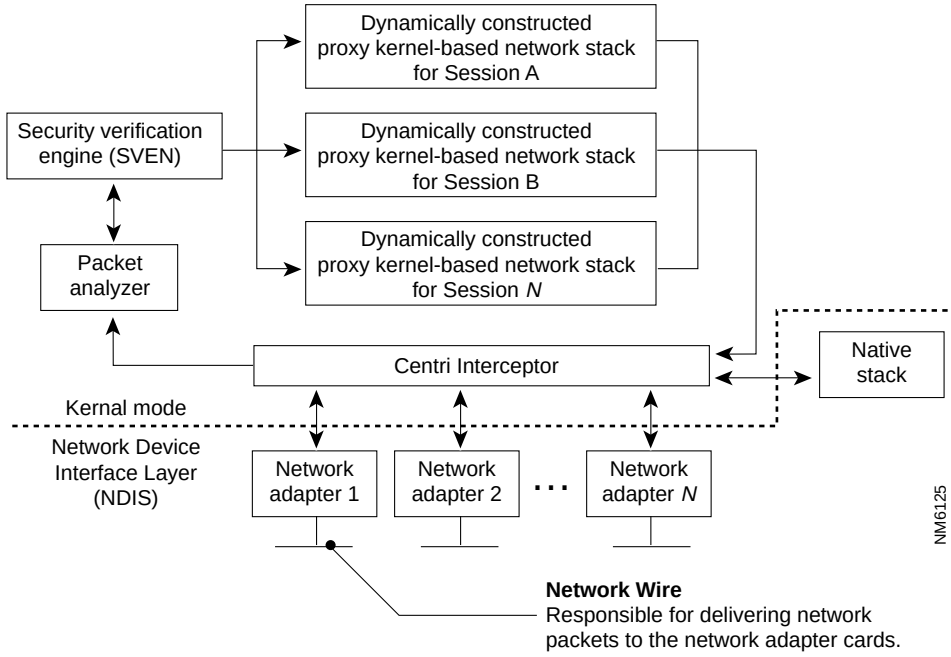
The following sections describe each of these components in detail.

Security Kernel

Cisco Centri Firewall's primary security component is the Security Kernel. The *Security Kernel* evaluates each incoming and outgoing network packet traversing the firewall server and enforces downloaded security policies against all such packets. The key to Cisco Centri Firewall's innovative architectural design is the Kernel Proxy™, the first modular, kernel-based, multi-layer session evaluation technology that runs in the Windows NT Executive, which is the kernel mode of Windows NT. Because Kernel Proxy operates within the Windows NT kernel, Cisco Centri Firewall does not suffer from the numerous interrupts and kernel-to-application-space context switches that slow the performance of traditional firewalls. In addition, because the Cisco Centri Firewall Kernel Proxy design is modular, it can be easily updated to address new security threats as they arise.

The Security Kernel component contains three major technologies: the Interceptor/Packet Analyzer, the Security Verification Engine, and Kernel Proxies. Figure 5-4 identifies the network packet flow within the Security Kernel.

Figure 5-4 Network Traffic Flow Through the Security Subsystem



Note The Security Subsystem inspects each network packet that traverses the firewall server. From the perspective of this subsystem, the native TCP/IP stack appears as another host on the network and it is subject to the same security inspections as a host that truly exists on the network.

The following sections detail each of these technologies.

The Interceptor and Packet Analyzer

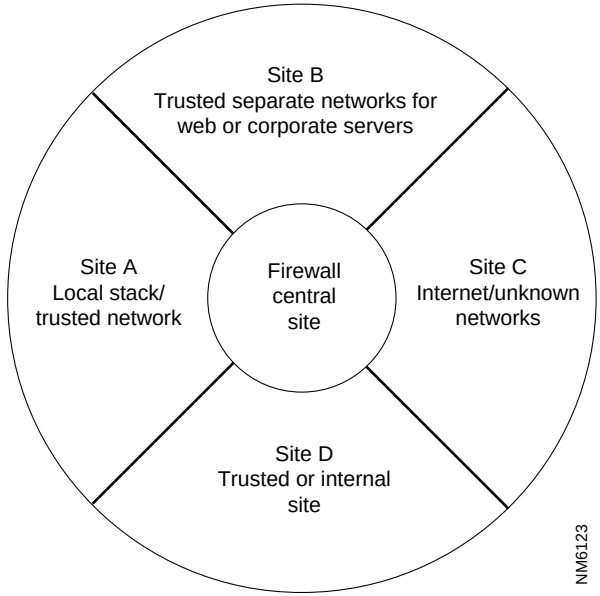
The first of Cisco Centri Firewall's technologies to handle network packets is the Interceptor. While a single driver comprises the Interceptor and the Packet Analyzer, they are logically two distinct components. The *Interceptor* captures all network packets arriving at the firewall server, whether they originate from the native network stack or from an adapter card connected to a network. In addition, it accepts network packets that have been processed by the custom network stacks (see the "Kernel Proxies" section) and puts them back out onto the network for delivery to the final destination. The Interceptor is positioned between the native network stack and the network adapter drivers, which allows it to intercept all network traffic before it reaches the native network stack.

Unlike most previous firewall implementations, Cisco Centri Firewall does not alter the source code or replace the native network stack provided with Windows NT or any third-party network stack that you have installed. From the perspective of Centri Firewall, an existing stack is just another network host, complete with a distinct IP address, residing on your internal network. As such, an existing stack can run network servers, such as DNS, and have its own security policy. Cisco Centri Firewall does not require that you dedicate another computer to your network servers.

Note Using the native stack to run network servers increases the complexity of the firewall server and the reliance on other software for stability. Because we believe that the firewall server should focus on protecting your network and that the services it provides should be minimal, we recommend that you dedicate your firewall server to enforcing your network security policies rather than having it assume a multi-purpose role within your network.

To enable this feature, Centri Firewall uses the concept of sites. A *site* represents one or more networks, which are trusted, untrusted, or unknown, and every site must be assigned to one or more network adapter cards. Likewise, each network adapter card that is installed on the firewall server must have one or more sites assigned to it. When a network packet arrives at the firewall server, it arrives from a particular site. The site that it arrives from determines which security policy applies to that packet. Figure 5-5 depicts the relationships among multiple sites within Cisco Centri Firewall.

Figure 5-5 Relationships Among Sites



Note The design of sites allows you to define security policies that let Site C communicate with Site B, but prevent it from communicating with Site A and Site D.

Sites determine how security policies are applied, how networks are organized, and how network address translation works within the firewall server. All data passes through the firewall's central site, which is where the Security Kernel resides. How a network packet travels across two sites determines which security polices are applied. It identifies the source and destination of the packet. If a network packet does not change sites, then no security policy applies to that network packet.

If a network packet is not destined for the firewall server, it does not even pass through the native stack. However, even those packets destined for the firewall server must undergo the security analysis defined by your security policies before they are passed to the native stack.

Once the Interceptor captures a network packet, it passes it to the Packet Analyzer. The *Packet Analyzer* recognizes the header information included with the network packets, prepares them to be looked at as a session by deriving signature data, and passes this derived signature data and the network packets to the Security Verification Engine.

The Security Verification Engine

The *Security Verification Engine* (SVEN) has two primary responsibilities:

- accepting valid security policies that are downloaded into the kernel from the Administrative Agent; and
- initializing and tracking sessions for all communications.

Using the Administrative Agent, administrators construct the security policies that control how Cisco Centri Firewall secures their network. It identifies how to control the network traffic that traverses between internal trusted sites and external sites, whether they are untrusted or unknown. During the system initialization and each time new security policies are applied to network objects, these security policies are converted into a security policy decision tree and downloaded to the SVEN, which uses this decision tree to evaluate incoming network packets.

This decision tree represents the cumulative security policy that encompasses the inheritance of security policies within the Networks tree of the user interface. By using a decision tree, Cisco Centri Firewall greatly reduces the processing time required to determine the appropriate policy to apply to a session, a vast improvement over the serial evaluations used by previous firewall architectures based on “accept” and “deny” rules stored in flat files.

In addition to accepting downloaded security policies, the SVEN accepts preprocessed information (signature data that associates a packet with a session) from the Packet Analyzer. At this point, the SVEN makes one of three determinations:

- do not accept the packet (drop it without further processing);
- the packet belongs to an existing session; or
- the packet indicates that a new, valid session should be set up by the SVEN.

If SVEN determines that an ongoing session exists to which that network packet belongs, it passes the packet to the custom-built protocol stack that for that session, which also performs any network address translations as defined by the security policy.

If the network packet is not part of an ongoing session, the SVEN looks up the security policy associated with the site assigned to the adapter card from which the packet originated and evaluates the provided signature data against that policy to determine whether to accept the session request. If the session is accepted, the SVEN then sets up a session and initializes a Kernel Proxy-based stack custom built according to the applicable security policy. The dynamic stacks are constructed based on the user-defined restrictions for a specific network service. After the SVEN constructs and initializes the session-specific dynamic stack, it passes the packet to that stack.

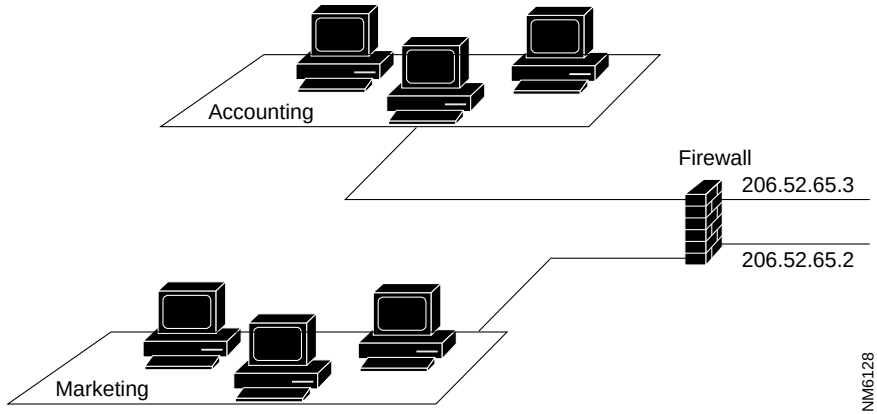
To set up a new session, the SVEN evaluates the information processed by the Packet Analyzer, whether it is TCP- or UDP-based, to identify the unique characteristics of that communication attempt. The unique characteristics include the protocols that compose the network packet and the information contained within certain layers of the network packet, such as the source and destination IP addresses and port numbers.

A session completes when either one (or both) of the communicating sides closes the connection or, in the case of UDP, the communication exceeds the timeout value assigned to the UDP proxy. Once a session completes, the SVEN tears down the session and frees the system resources associated with that session.

As mentioned earlier, the custom proxy stack performs “on-the-fly” basic network address translation (NAT) for those packets requiring it. During session initialization, the SVEN evaluates the security policy to determine whether to assign an address translation to that session. If the SVEN assigns an address translation to a session’s stack, the proxy stack simply performs a lookup to determine what the correct address mappings are for the remaining packets belonging to that session.

Network address translation is a method of sharing a common pool of IP addresses among a larger collection of computers. It was developed in response to the dwindling supply of unique IP addresses available to users on the Internet. As depicted in Figure 5-6, network address translation attempts to solve the pending address shortage in IP networks.

Figure 5-6 Network Address Translation Maps Multiple Internet Addresses to a Single External Address



Note In this example, the registered IP address of 206.52.65.3 always designates members of Accounting, while 206.52.65.2 always designates members of the Marketing department. Such designations can provide additional information in the audit records about how your network is being used.

In addition to conserving IP addresses, network address translation provides additional security features for your networks by hiding its internal structure and allowing logical mappings to users who compose the different groups and departments within your company. It also prevents an external user from deriving information about the network topology by using a traffic analysis program because only those registered external addresses that are publicly available can be determined. Thus, external attackers can learn nothing about the host-specific information they desire.

Kernel Proxies

To perform its function of inspecting the actual network packets and enforcing security policies, the Security Kernel uses dynamic, custom TCP/IP-based stacks. One purpose of these custom TCP/IP stacks is to reduce the dependency on untrusted source code, such as

the native TCP/IP stack provided with the Windows NT operating system. These custom stacks are session dependent, which means that they are constructed on-the-fly when a new session request arrives at the firewall. Unlike normal TCP/IP stacks, the Security Kernel constructs these stacks out of kernel-level proxies, the technology we refer to as Cisco Centri Firewall Kernel Proxy.

Currently, Cisco Centri Firewall includes eight kernel proxies. Each of these *kernel proxies* evaluates the network packet headers and data to provide security checks at every layer in the dynamic stacks. They verify that the network packets and their data conform to the specifications of the network services. Table 5-1 lists the currently available proxies and the user-definable security checks that they can perform, as well as the integrated security countermeasures and preventive techniques that they employ:

Table 5-1 Kernel Proxies and Security Countermeasures

Kernel Proxy	Security Inspection Description
IP	**Source and Destination Checks.** This check verifies that the source IP address has permission to communicate with the destination IP address.
	Ping of Death Attack Prevention. Cisco Centri Firewall ensures that each IP packet does not exceed the maximum length, which prevents Ping of Death attacks. If a packet exceeds the maximum length, the IP proxy drops the packet and generates an audit record for the event. To protect the native TCP/IP stack provided with Windows NT 4.0, we recommend that you install the Windows NT Service Pack 2, which corrects the problem and replaces the *ping* command with one that does not allow you to send such requests.
	IP Spoof Attack Prevention. When a new IP connection starts, the IP proxy looks up the source site in static routing tables, rather than the dynamic routing tables. Every IP packet belonging to an IP session comes in on a network adapter that is associated with a specific site. If the actual site does not match the routing site, the IP proxy drops the packet and generates an audit record for the event.
ICMP	**Message Type.** This proxy allows or denies a network session based on the type of its ICMP message. When you define a network service that requires ICMP, you define the setting for this option. If the ICMP proxy drops a packet based on the settings in the security policy, it generates an audit record for the event. Using this setting, you can deny echo requests, thereby preventing ICMP_ECHO flood attacks.

Table 5-1 Kernel Proxies and Security Countermeasures (Continued)

Kernel Proxy	Security Inspection Description
TCP	**Port Check.** This check verifies that the initiating host requests services through a valid TCP port. When you define a network service or network application that requires TCP as its transport layer service, you define the setting for this option. **TCP SYN Flood Prevention.** The TCP proxy contains a threshold value for the number of half-open connections. This value adjusts dynamically based on the amount of physical memory currently available (You can adjust this value by installing additional physical memory in the firewall server.) When the number of half-open connections exceeds this threshold value, the firewall considers itself to be in TCP_SYN attack mode. While in this mode, all half-open connections are compared against a cache of the most recently accessed IP addresses. For those IP addresses that have not been accessed recently, a percentage of the oldest half-open connections to those addresses are closed on a site-by-site basis. This closing of half-open connections frees system resources, and the firewall server continues cycling through the list of half-open connections until the number of half-open connections returns below the threshold value.
UDP	**Port Check.** This check verifies that the initiating host requests services through a valid UDP port. When you define a network service or network application that requires UDP as its transport layer service, you define the setting for this option.

Table 5-1 Kernel Proxies and Security Countermeasures (Continued)

Kernel Proxy	Security Inspection Description
HTTP	**Inline User Authentication.** This check authenticates the user by passing the session request up to the Authentication Agent. The Authentication Agent authenticates the user according to one of the possible authentication methods. **HTTP Filtering.** This check refers to a list downloaded to the kernel that controls file and site access over the HTTP network service. Using the HTTP filters, you can prevent access to specific sites on the Internet and to specific types of files, such as Java applets. Each entry in the deny list can be an IP address, a domain name, or a file type. A domain name can be a specific computer name, such as www.undesired_site.com. It can also be a domain name that represents multiple computers, such as undesired_site.com. In this case, the entry applies to all computers within that domain. A file type denotes the extension on the file, such as *.class*, *.ocx*, and *.alx*, which specifies that you want to filter any Java and ActiveX files. **Allowed Action Checks.** These checks evaluate all actions associated with the HTTP protocol to determine whether a user has permission to use particular services provided by the protocol: • post an object onto an HTTP server; and • get an object from an HTTP server. **HTML Filtering.** These checks require that the contents of the network packets be assembled and evaluated as a whole and therefore, they are performed in the application layer. Primarily, these services modify the data of the HTML documents that are transferred during an HTTP session. • *ActiveX Control Filtering.* Removes the <OBJECT> tags from any HTML document. • *Java Applet Filtering.* Removes the <APPLET> tags from any HTML document. • *JavaScript and VBScript Filtering.* Removes the <SCRIPT> tags from any HTML document.

Table 5-1 Kernel Proxies and Security Countermeasures (Continued)

Kernel Proxy	Security Inspection Description
FTP	**Inline User Authentication.** This check authenticates the user by passing the session request up to the Authentication Agent. The Authentication Agent authenticates the user according to one of the possible authentication methods. **Non-Transparent Proxy Mode Support.** This mode is automatically enabled by the proxy if the destination address of a session request is the firewall server. It allows users to connect to an IP address assigned to a firewall server. From the firewall server, users can then connect to other computers behind the firewall server using common FTP commands. This mode is useful when trying to connect from an external network to internal network servers. **Allowed Action Checks.** These checks evaluate all actions associated with the FTP protocol to determine whether a user has permission to use particular services provided by the protocol: • read an object from an FTP server; • write an object to an FTP server; • delete an object from an FTP server; and • traverse an FTP directory.
Telnet	**Inline User Authentication.** This check authenticates the user by passing the session request up to the Authentication Agent. The Authentication Agent authenticates the user according to one of the possible authentication methods. **Non-Transparent Proxy Mode Support.** This mode is automatically enabled by the proxy if the destination address of a session request is the firewall server. It allows users to connect to an IP address assigned to a firewall server. From the firewall server, users can then connect to other computers behind the firewall server using common Telnet commands. This mode is useful when trying to connect from an external network to internal network servers. **Port Check.** This check verifies that the initiating host requests services through a valid Telnet port.

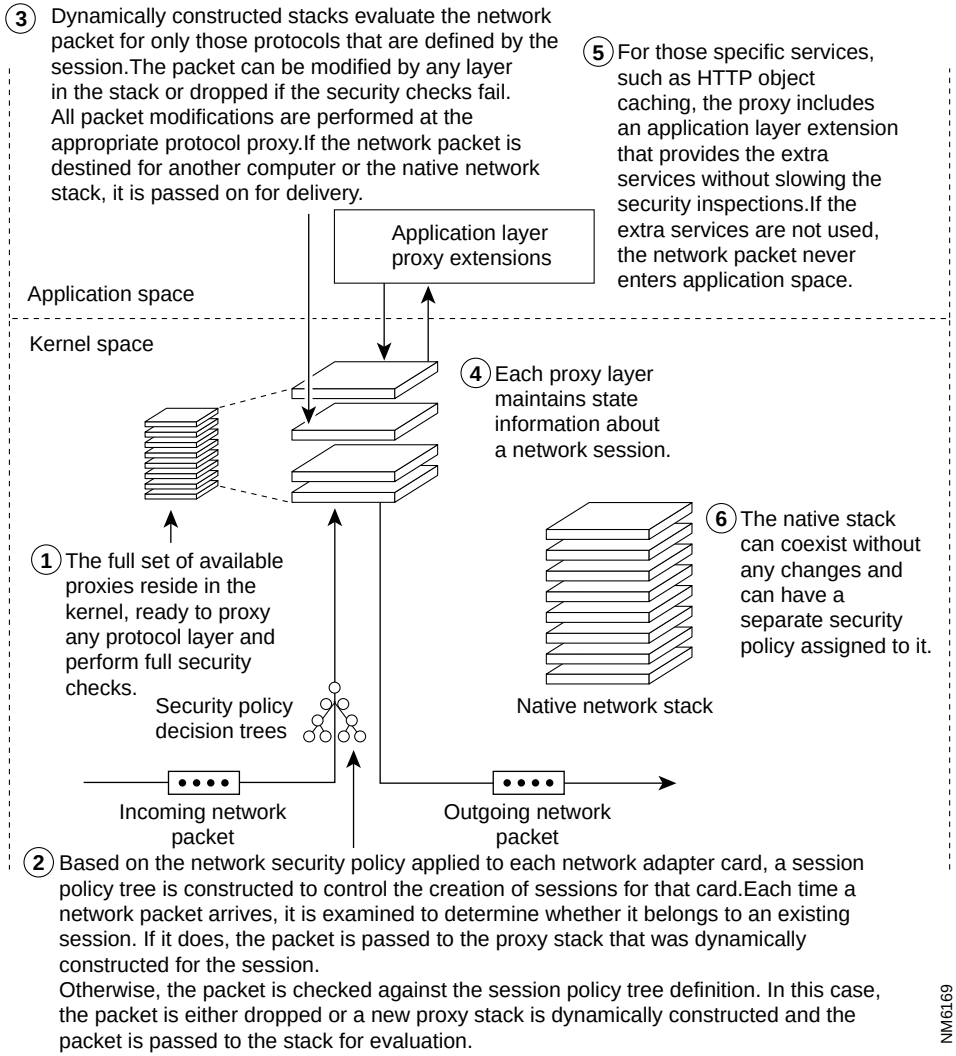
Table 5-1 Kernel Proxies and Security Countermeasures (Continued)

Kernel Proxy	Security Inspection Description
SMTP	**SMTP Flood Countermeasure.** Because of the difficulties associated with mail generators that randomly choose the source IP address of the mail that sent to a single destination, mail floods are difficult to counter effectively without harming the productivity of users on the network to which the mail is being sent. The SMTP proxy allows you to reject all mail sent from a particular IP address or network by defining security policies that specify who can and cannot send mail to your network.
	Allow Routing Characters. Indicates whether the recipient path of the message header can contain routing characters (!, [,], :, and %). By default, routing characters are allowed.
	Limit the Number of At Signs (@). Indicates whether the proxy allows recipient paths to contain more than one at-sign (@). The default allows any number.
	Allow the Verify Command. Indicates whether the VRFY command (verify) can be passed through the firewall to the SMTP server. The default value allows this command to pass through the firewall server.
	Allow the Expand Command. Indicates whether the EXPN command (expand) can be passed through the firewall to the SMTP server. The default value allows this command to pass through the firewall server.
	Limit the number of Recipients. Allows you to limit how many recipients can be addressed in a single message. The default value allows any number.
	Limit the Message Size. Allows you to limit the number of bytes that can be contained in a single message. The default value allows messages up to a terabyte.

Taken directly from proven military-grade security practices, Cisco Centri Firewall employs the concept of *network sessions*—a complete communication exchange between two network objects—to construct proxy-based stacks dynamically for each session. These custom stacks comprise only those protocol proxies that are relevant to the session for which they were built, allowing the firewall server to custom-tailor the level of stringency used to evaluate all packets belonging to a single network session.

Kernel proxies perform security checks where they should be performed, in the kernel as the data passes up or down the network stack. Doing in-line inspections increases performance and prevents the possibility of passing invalid network packets up the network stack into application space. Figure 5-7 depicts the network packet evaluation process used by Centri Firewall's kernel proxies.

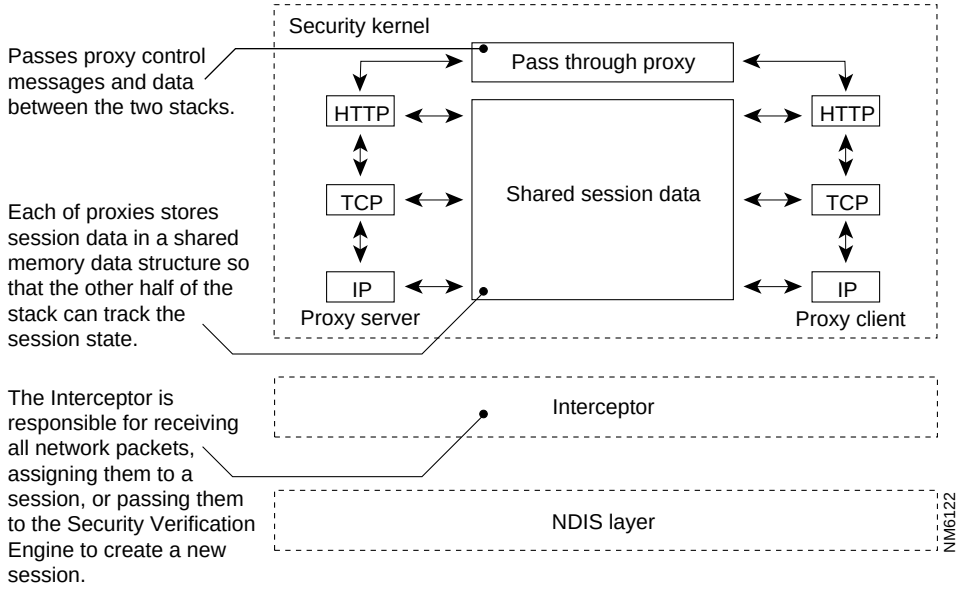
Figure 5-7 Cisco Centri Firewall Kernel Proxy Architecture



Note This architecture analyzes, in kernel space, the complete command set for each relevant protocol. The SVEN determines the set of analyzing protocols on a per session basis as defined by your security policy. This architecture also provides value-added services and leaves your native network stack intact.

The dynamic custom stacks that comprise the kernel proxies make it possible to examine and modify each network packet at every layer as the packet travels up the custom, high-speed network stack, obviating the need to pass the packet from kernel to application space and back again. This evaluation technique provides both maximum security and optimum performance. As a network packet passes through each proxy layer, that kernel proxy evaluates the packet to ensure that both it and its data are valid. If a packet fails to pass the evaluation of any proxy layer, that kernel proxy drops the packet without propagating through the remainder of the stack. Figure 5-8 depicts a session-specific network stack.

Figure 5-8 Intelligent Session-Specific Stacks



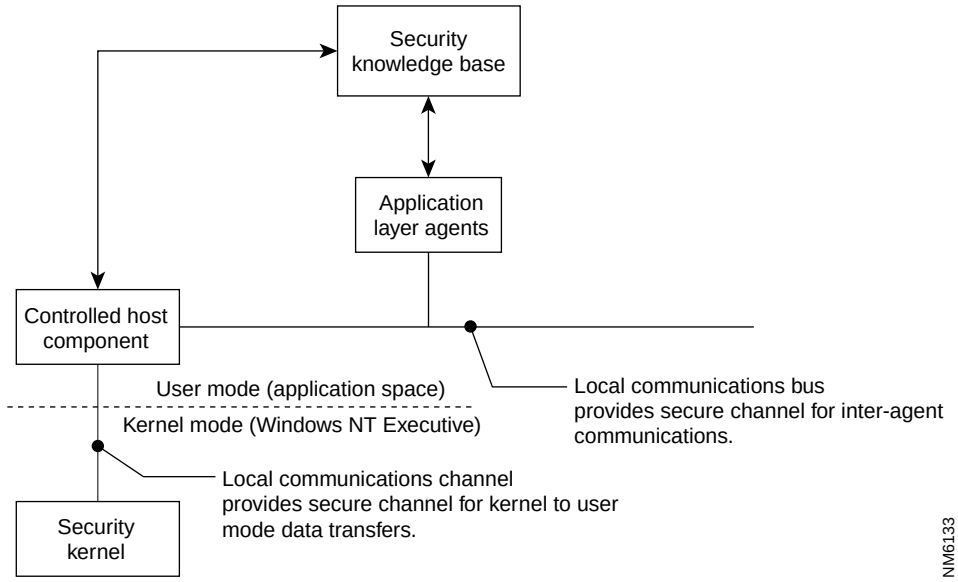
Note The Kernel Proxy architecture couples the fastest possible performance with the strictest security evaluations by dynamically constructing stacks on a per session basis. Each custom stack contains only those protocols that are required by the session.

For example, the connection depicted in Figure 5-8 shows an HTTP session. When the Interceptor receives an HTTP service request, it passes the request to the SVEN, which constructs two kernel proxy stacks that are best suited to evaluate that session based on what requirements are identified by the security policy. The stacks that are constructed contain the kernel-mode IP, TCP, and HTTP proxies only. The stacks share the data derived by the kernel proxies to ensure that session state synchronizes across the stacks.

Controlled Host Component and Communications Channels

When transferring information between each other and between the application and kernel layers of Windows NT, the agents composing the Cisco Centri Firewall use two secure communication channels: the local communications channel and the local communications bus. The *local communications channel* (LCC) transfers data securely between the kernel-mode agents and the user-mode agents. When using this channel, the Security Kernel only sends information to the Controlled Host Component, which acts as a pass-through to other agents running in the application layer. Figure 5-9 depicts the relationship between the two communication channels and the Controlled Host Component.

Figure 5-9 The Controlled Host Component and Communication Channels



Note The Controlled Host Component passes information between the application layer agents and the Security Kernel using two secure communication channels.

The *Controlled Host Component* (CHC) operates in the application layer and transfers data requests between the user-mode agents and the Security Kernel agents. The CHC also performs system integrity checks and starts and authenticates all of the agents that compose Cisco Centri Firewall as they are loaded on an as needed basis. During system initialization, the CHC performs integrity checks on the Cisco Centri Firewall's files by comparing a computed cryptographic checksum of each binary against a pre-calculated value stored in the Security Knowledge Base (discussed later in this chapter). If the values do not match, the CHC logs the fact that the binary file was modified without authorization.

Each time that a service or agent starts up, the CHC, in combination with the Security Knowledge Base, authenticates the agent or service using a public-private key handshake. This authentication method uses the Microsoft Crypto API to perform the handshakes.

The agents running in user mode, such as the Authentication and Logging Agents, use the *local communications bus* (LCB) to exchange data securely with each other. Both the LCC and the LCB are designed to exchange data quickly and securely.

Logging Agent

The *Logging Agent* processes audit events generated by all other system agents and the Security Kernel and stores them as audit records in the Security Knowledge Base. When an agent running in the application layer generates an audit event, it sends it to the Logging Agent via the secure LCB. When a kernel-level agent or proxy generates an event, the agent or proxy uses the CHC to pass the audit event to the Logging Agent. The Logging Agent allows the administrator to preprocess audit events because it allows you to specify which audit records should be stored in the Security Knowledge Base.

Authentication Agent

The *Authentication Agent* verifies that a user is who that user is claiming to be by verifying that user knows a piece of shared, secret information before gaining access to a particular firewall service. The Authentication Agent provides two types of user authentication:

1. Out-of-band authentication for all network services. This authentication relies upon the Windows NT authentication model provided by Windows NT Server. All network services are authenticated when the security policies are applied to Windows NT Domains, Group accounts, and User accounts.
2. In-line authentication for FTP, HTTP, and Telnet. As dictated by the security policies, these select network services can be defined so as to require user authentication. The user authentication method can be based on reusable passwords or S/Key.

Reusable Passwords are the simplest form of authentication. This method requires the user to enter a text string that only the user knows. Every time a user needs to authenticate, the user enters the same password. Reusable passwords, however, are vulnerable to packet sniffers and common password attacks, and therefore, they are not considered a reliable authentication mechanism. For this reason, we do not recommend using reusable passwords, and we strongly recommend against using reusable passwords to gain access from untrusted networks. This method does not require any third-party application support.

S/Key uses a one-time password system developed at Bellcore. Under this system, the user generates a set of passwords based on a "seed" word or phase. When the firewall server prompts the user for authentication information, it provides a challenge based on the result of an algorithm applied iteratively to the seed value. The user must enter the password appropriate for that challenge. While S/Key can validate the user's current response, it has no way of predicting the user's next response. Each time users attempt to log in, they are prompted for a different password. This method requires the additional support of a third-party S/Key authentication server.

Currently, only the HTTP, FTP, and Telnet proxies are supported by in-line authentication; however, because the Authentication Agent specializes in authenticating end users, its services may be used by other kernel and application proxies.

Within Cisco Centri Firewall, you define who should authenticate, as well as when, within your security policies. Even though the Authentication Agent runs in the application layer, its effect on performance is negligible because once the user authenticates to the Authentication Agent, the kernel proxies resume their inspection of the network packets.

Thus, Cisco Centri Firewall performs the overall communication relatively fast because it evaluates only a small part of the session in user mode (the application layer of Windows NT).

Security Knowledge Base

The Cisco Centri Firewall product family incorporates a proprietary knowledge store, which is an active object-oriented database derived from the frame technologies developed within the artificial intelligence community. The *Security Knowledge Base* acts as a central repository for the Cisco Centri Firewall. It stores configuration data as well as information that the security system generates when it is active, including audit records.

The Security Knowledge Base is an event-driven system that provides call-backs and notifications to agents that register an interest in particular data and state changes associated with the data in its store. These notifications allow the agents to react to changes in the Security Knowledge Base. For example, when a new audit record is generated by a Logging Agent, elements of the Monitoring Subsystem that are interested in this kind of audit records are notified. These elements then examine the new data and update any associated state related to that audit record.

The Security Knowledge Base provides a common communication interface for the agents within the Cisco Centri Firewall. This common interface reduces the complexity of the security system and allows new agents to be added to the system without affecting the existing agents. An agent only needs to be aware of what data it should register an interest in and what data it generates, and it does not understand the specifics of communicating with other agents. Because the interface with the Security Knowledge Base is constant, the agents only have to know how to interface with it.

The Security Knowledge Base also maintains authentication information, and in fact, when an administrator of the Cisco Centri Firewall authenticates himself to the system from the user interface, he really authenticates to the Security Knowledge Base. (The Centri Administrator authentication uses the Microsoft Cryptographic API (Crypto API) and passwords encrypted using a symmetric algorithm based on RC4.)

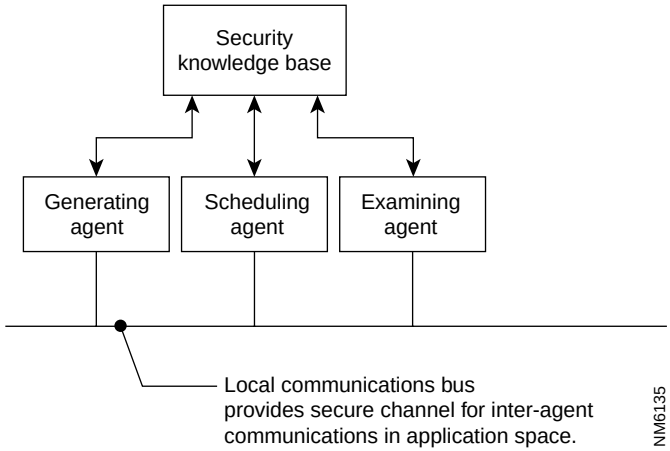
All traffic passed between the Security Knowledge Base and agents is encrypted using support provided by the Microsoft Crypto API. By encrypting this traffic, we can ensure that information cannot be analyzed by packet sniffers when Cisco Centri Firewall agents are distributed to other computers. An example of a distributed agent is the Administrative Agent when it is configured for remote administration.

Reporting Subsystem

The *Reporting Subsystem* generates and presents all reports about the network activity within the Cisco Centri Firewall's domain. (Several report file formats exists. Currently, the Reporting Subsystem provides HTML-based reports, as well as plain text reports.) This subsystem generates on-demand reports, which provide statistics about the running system whenever the administrator wants to view them. It also generates scheduled reports based on specified time periods and types of reports defined by the administrator. Typical scheduled reports are daily, weekly, and monthly usage statistics, as well as network service breakdowns.

As depicted in Figure 5-10, the Reporting Subsystem uses three agents: the Generating Agent, the Examining Agent, and the Scheduling Agent. The Generating Agent processes the data stored in the Security Knowledge Base, fills in the report templates to generate the report, and then passes the complete report off to the agent that requested its services.

Figure 5-10 The Reporting Subsystem



The Examining and Scheduling Agents are the two agents that request reports from the Generating Agent. The *Examining Agent* allows you to view on-demand reports, as well as reports generated by the Scheduling Agent. It listens for HTML requests from either a standard web browser or from the Administrative Agent (described later in this section).

Depending on the type of request received, the Examining Agent directs the requesting agent either to existing reports made by the Scheduling Agent or to an on-demand report generated by the Generating Agent as a result of the request. When the *Generating Agent* completes the on-demand request, it passes the generated report to the Examining Agent for presentation to the user.

The *Scheduling Agent* wakes up according to user-specified intervals and kicks off the Generating Agent, which generates the reports according to the preferences defined by the administrator and passes it back to the Scheduling Agent. The Scheduling Agent then timestamps the report and creates a file that can be viewed at any time using standard file tools or via the Examining Agent.

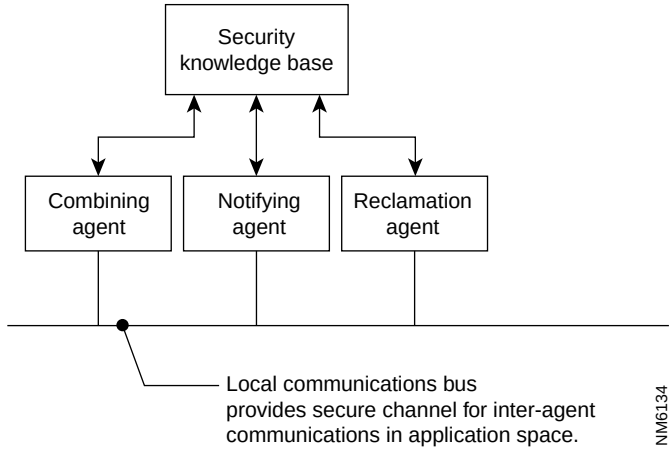
Currently, three categories of reports are generated:

- **Summary.** Provides summary information about network activity.
- **Detailed.** Provides detailed information about network activity.
- **Warning.** Provides information about the state of the security system.

Monitoring Subsystem

The *Monitoring Subsystem* generates all statistical data and processes all system operation events and audit records within the Cisco Centri Firewall. It monitors all audit events and records before and after they are stored in the Security Knowledge Base, looking for statistical information, for records of audit events that indicate possible ongoing attacks, and other records of interest to the security of the system. It then alerts the network administrator when it detects such records. As depicted in Figure 5-11, the Monitoring Subsystem makes use of three agents: the Combining Agent, the Notifying Agent, and the Reclamation Agent. Each of these agents performs a specific task in the evaluation, generation, and maintenance of audit events and audit records. The combined efforts of these agents ensure that administrators are notified of suspicious network activities and provide detailed analysis of what is happening within the security system.

Figure 5-11 The Monitoring Subsystem



Note *Audit events* are the system events that trigger the generation of an audit record. *Audit records* summarize what actions caused the audit event to be triggered.

The *Combining Agent* processes audit records generated by the security system. It parses the audit record's data and combining it into statistics that are interesting and meaningful to administrators. It contains a combining element for each type of proxy and an Alerter element. These proxy elements work by deriving statistical and summary data (event aggregation) about the audit events generated by the security system, as well as the audit records already stored in the Security Knowledge Base. Once an element derives knowledge about the audit events, it stores the new records, summarizing that knowledge, in the Security Knowledge Base. By pre-and post-processing the audit events, the combining elements can provide filtering to reduce the amount of information stored in the Security Knowledge Base. In addition, they can evaluate streams of lower level audit events for some higher level meaning.

The Alerter element is a little different. While it is a combining element, the Alert element does not produce the same type of data as other combining elements. Rather than creating new statistical and summary data streams, the *Alerter element* creates alert streams that are processed further by a different part of the security system–the Notifying Agent. However,

the set of input streams the Alerter element can process is the same as the set of input streams for all other combining elements; steams of audit records are generated by various parts of the security system, such as the Security Kernel and other combining elements.

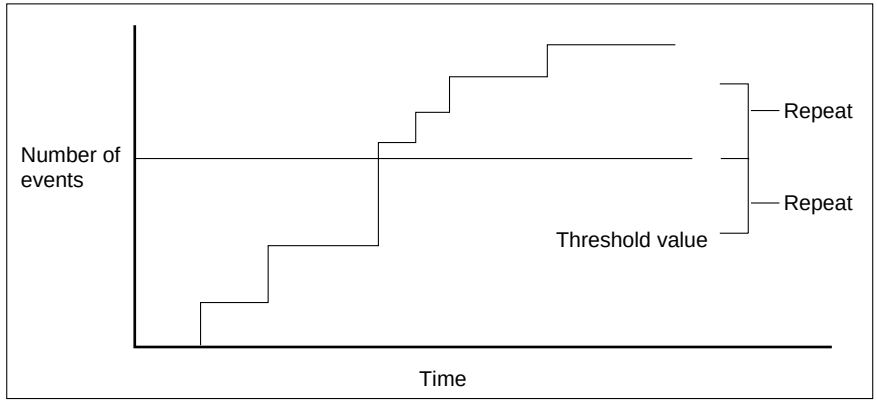
Each combining element monitors these streams of data and picks out only the information that it is interested in and generates new information using this data. Combining elements can be stacked upon each other to perform more sophisticated studies of the information in the input streams. In other words, the output of one combining element can be the input for another combining elements, including the Alerter element. The Alerter element considers the data that it picks out of its input streams in relation to the data that it has already processed to determine whether any statistical significance exists about the state of the system's security. These determinations are based on thresholds that are defined by the administrator of the system.

When the Alerter element determines that a possible security threat is in progress, it signals the Notifying Agent, passing it information about what type of warning to issue. The *Notifying Agent* contacts the administrator. The Notifying Agent currently contains four notifying elements: e-mail, pager, pop-up dialog boxes on the firewall server, and the Administrative Agent (remote and/or local to the firewall). (The Notifying elements are based on standard Microsoft APIs, such as MAPI and TAPI.) Each of these notifying elements performs a specific type of notification. In addition, the Notifying Agent generates audit records concerning failures in notification attempts, such as a failure to submit an e-mail notification. These audit records provide information about the status of your computer and the operation of network services, such as your e-mail server.

When you modify the audit event settings, you are modifying the threshold values and the criteria for when to alert the administrator about the activities that are noticed. You are also determining which events are important to the Alerter element. As part of these settings, you can define threshold values and repeat criteria. If the event counter passes the threshold value, then a notification occurs. You can specify a repeat value that determines how often you want to be notified after a particular threshold value is crossed. For example, you can specify that the first administrator notification should occur after a particular event occurs 10 times, and from then on, you only want to be notified every 400 times the event occurs.

As Figure 5-12 shows, the relationship among audit events, threshold values, and repeat notification values can be defined using a simple step chart.

Figure 5-12 Threshold Values and Repeat Notification Values



The last agent included in the Monitoring Subsystem is the Reclamation Agent. The *Reclamation Agent* archives historical data to any ODBC-compliant database, as well as cleaning up the audit records within the Security Knowledge Base. Based on user-defined settings, the Reclamation Agent periodically removes old audit records from the Security Knowledge Base. The removal of old audit records is based on the age of the record and/or the size of the Security Knowledge Base.

Administrative Agent

The *Administrative Agent* is the Cisco Centri Firewall user interface, natively designed for Windows NT, that translates between the administrator and the security system. It presents technical system information to the Centri Administrator using metaphors that people understand more easily, and it translates the modifications to the system for the appropriate system components in a language that the system components understand.

As with the major subsystems within Cisco Centri Firewall, the Administrative Agent comprises multiple, specialized libraries that perform specific tasks. In much the same way that the system agents interface with the Security Knowledge Base, each Administrative Agent library interfaces with a single executable to transfer data to and from its graphical presentation. By dividing the user interface into modular, task-specific libraries, a single file

can be replaced to add new features and functionality or enhance old features without re-testing every feature within the interface. In addition, new libraries can be added without concern as to how they will affect the existing libraries.

The Administrative Agent allows you to define a single security policy and apply it to as many network objects as you want via clear and easy-to-understand graphical representations of your network. The Administrative Agent also allows you to view and administer your network in as much, or as little, detail as your organization's security policy requires. Such time and effort cost savings are provided throughout the Administrative Agent. The Administrative Agent introduces five technological breakthroughs in firewall user interface design:

- Natural Network Views
- Bundled Applications
- Policy Builder
- "Drag-and-Drop" deployment of security policies
- Built-in web browser control

These five technologies are described in the following sections.

Using the Natural Network Administration Model

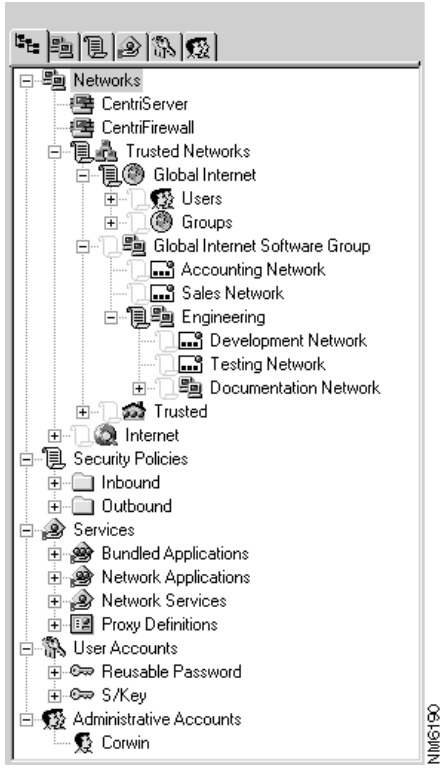
With Cisco Centri Firewall, you can represent and view your network from a physical layout perspective or a logical network perspective, whichever you prefer. These *Natural Network Views* are presented using a tree control, similar to the one found in the Windows Explorer interface. When defining new network objects, you can represent abstract concepts, such as entire networks, subnets, and individual hosts.

In addition, you can identify any Windows NT Domain controllers on your network, and the Group accounts and User accounts for that domain will automatically be added to the Networks tree. Windows NT Domain controllers support is particularly useful in environments that use DHCP to dynamically allocate IP addresses for network hosts. It enforces security policies that are applied to a Windows NT Domain, Group, or User account regardless of which computer a user logs into, assuming that the computer belongs to the domain.

By organizing your network objects using logical networks, you do not have to have Windows NT Domains to make sense out of your network. You can define and organize the network objects that you want to protect into logical groups that function much like folders within the Windows Explorer interface. You can apply security policies to an element within the logical groups or to the logical group itself, which lets you define the level of detail required to administer it effectively.

Note Current generation user interfaces focus on writing rules, but Cisco Centri Firewall focuses on implementing security policies (see Figure 5-13). The user interface allows you to group your network objects and network services to support any organizational model that provides meaning to you. In addition, you can reference these groups of objects and services when developing security policies to keep them simple so that they can inherit the knowledge created from your organizational model.

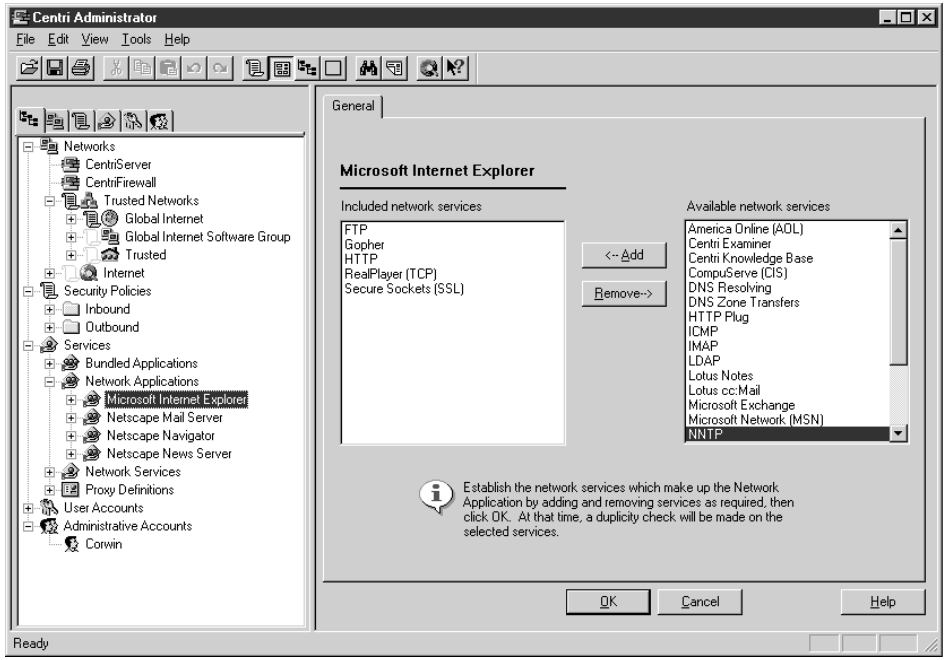
Figure 5-13 Graphically Define Your Network Objects



Using Bundled Applications to Simplify Security Policies

Just as you can group your network objects into logical groups within Cisco Centri Firewall, you can group network services into more meaningful bundles. Within the Services tree, we provide two layers of abstraction from the details of defining the settings of network services, such as TCP, UDP, HTTP, and SMTP. We believe that most administrators do not think of user programs that use the network in terms of which protocols they require. Instead, they think of them as the applications that they commonly use and want to exert control over, such as Microsoft Exchange or Netscape Navigator. This abstraction layer provides logical groupings for the network services that compose a user program. We call this first layer of abstraction *Network Applications* (see Figure 5-14).

Figure 5-14 Network Applications Branch



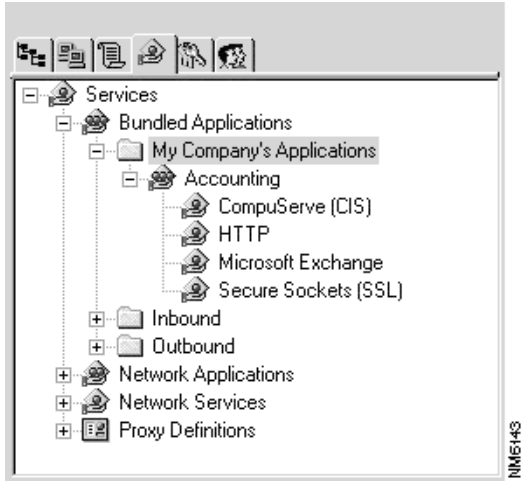
Note Network Applications provide a level of abstraction from the protocol to port mappings, allowing you to logically group network services used in common user applications.

Cisco Centri Firewall includes pre-defined, “out-of-the-box” network applications. With definitions for the most common user applications, such as Internet Explorer and Netscape Navigator, you can provide secure Internet services to your organization without having to determine which protocols and network services are required by an application. These pre-defined applications reduce the time that administrators spend on guessing which services are required and testing to verify that the application works as it should. You only

have to determine what you do and do not want users to access via those applications. Simply decide which security options you do not want your network users to utilize, such as Java and ActiveX applications, and enable or disable them within your security policy.

In addition to thinking about user programs in terms of the product name, it is more natural to organize them according to the role of the person in an organization. We can assume that accountants do not require the same programs as graphics artists or software developers. Some of the applications may be standard, but each job may require a few special programs. The second abstraction layer within Cisco Centri Firewall is designed for just this purpose—to organize software programs according to common application within an organization. We call this abstraction layer *Bundled Applications* (see Figure 5-15).

Figure 5-15 Bundled Applications Branch



Note Bundled Applications provide an additional level of abstraction, allowing you to logically group applications and services into bundles that reflect your users' common software needs.

To simplify security policy development, you can define network applications and bundled applications and reference them in security policies. For example, let's say that you have defined a Bundled Application called *Accounting* that contains four network applications: Microsoft Exchange, Lotus Notes, Netscape CoolTalk, and CompuServe Information Systems (CIS). The network services that make up these applications might include TCP listening on several different ports, SMTP, FTP, and HTTP, and they could contain the port settings and value-added service settings, such as requiring user authentication for FTP and HTTP and allowing JavaScript programs within your web pages.

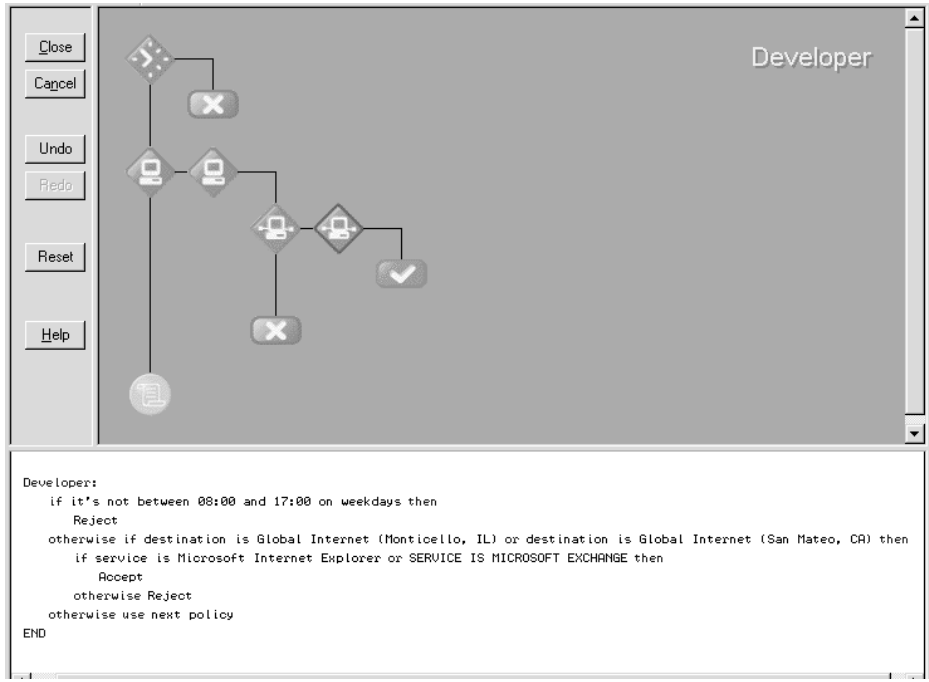
Instead of setting these services up each time that you want to define a security policy with them, you can set them up once and reference them as the *Accounting* bundled application in the security policy whenever you want to provide those services for a particular network object.

Defining Security Policies with Policy Builder

Current rule-based firewalls are hard to manage, and it is often difficult to verify that a security policy is being enforced as intended. With Centri's Policy Builder, you can create security policies simply by selecting options presented in graphical decision trees. These security policy decision trees are displayed along with a clear explanation written in Centri's Secure Script, a policy language based on plain English rather than the often-confusing table-based languages used in traditional firewalls. Because you do not have to peruse complex table rules to discover that obscure single IP address for which you did not define an important network service, you can ensure more easily that no holes exist in your network security policy.

Within Centri Firewall, the Policy Builder contol (shown in Figure 5-16) validates your security policy as you write it to ensure that your security policy is correct the first time you deploy it. In addition, you can review each security policy by reading the English translation displayed in the Secure Script pane.

Figure 5-16 Design Graphical Security Policies



The Policy Builder control allows you to define security policies visually around abstract network applications, such as Netscape Navigator and Microsoft Exchange, as well as around network services like FTP and HTTP.

Applying Security Policies Using Drag and Drop

As mentioned earlier, you can apply security policies to an entire network, a subnet, or a single host, as well as to all trusted, untrusted, and unknown networks. Once you write a security policy, you simply drag and drop it onto each network object that you want to protect using that policy. While many applications take advantage of drag and drop, Cisco Centri Firewall uses this technology to save you time and ease the administration of complex networks.

If you have defined logical network groupings, you can simply drag and drop them onto the network objects that comprise those groupings or onto the groupings themselves. You apply the security policies only where they are needed.

For homogenous Windows networks, you can apply security policies to common network objects, such as Windows NT Domains, Group accounts, or individual User accounts, which are automatically discovered and remain current with the users in your domain. To find out which security policy is running where, simply point and click your way through the graphic presentation of your network and click the **View Policy** option on the shortcut menu.

Built-in Browser Control

The Cisco Centri Firewall user interface also incorporates the Microsoft Internet Explorer browser control. This control allows the administrator to view the reports generated by the security system, as well as HTML-based documentation from within the user interface.

By selecting the Internet Help icon on the toolbar, the administrator can load the browser control in the right-hand pane, or view pane, of the Administrative Agent. The default HTML page loads, which provides options for navigating to generated reports and online reference documentation.

The next chapter presents an overview of the process that you should use to configure your new security system. It provides worksheets to help you properly identify your network assets and to determine what network services and applications you need to allow through your firewall, as well as who needs access to those services.