McAfee NGFW Reference Guide for IPS and Layer 2 Firewall Roles 5.7

NGFW Engine in the IPS and Layer 2 Firewall Roles

McAfee®
An Intel Company

Legal Information

Revision: SGIRG_20140328

Table of Contents

Introduction

Chapter 1

Chapter 2

Chapter 3

Chapter 4

Setting up Security Engines

Chapter 5

CHAPTER 6

CHAPTER 7

CHAPTER 8

CHAPTER 9

TRAFFIC INSPECTION

APPENDICES

INTRODUCTION

In this section:

CHAPTER 1

USING SMC DOCUMENTATION

This chapter describes how to use this guide and related documentation. It also provides directions for obtaining technical support and giving feedback about the documentation.

The following sections are included:

- How to Use This Guide (page 12)
- Documentation Available (page 13)
- Contact Information (page 14)

How to Use This Guide

The *McAfee NGFW Reference Guide for IPS and Layer 2 Firewall Roles* provides information that helps administrators of McAfee® NGFW in the IPS and Layer 2 Firewall roles to understand the system and its features. It provides high-level descriptions and examples of the configuration workflows.

This guide is divided into sections, which each include one to several chapters. The first section provides a general introduction to intrusion detection and McAfee NGFW in the IPS and Layer 2 Firewall roles. The sections that follow each include the chapters related to one feature area. The last section provides detailed reference information in tabular form, and some guideline information.

For other available documentation, see Documentation Available (page 13).

Typographical Conventions

The following conventions are used throughout the documentation:

Table 1.1 Typographical Conventions

Formatting	Informative Uses
User Interface text	Text you see in the User Interface (buttons, menus, etc.) and any other interaction with the user interface are in **bold-face**.
References, terms	Cross-references and first use of acronyms and terms are in *italics*.
`Command line`	File names, directories, and text displayed on the screen are `monospaced`.
`User input`	User input on screen is in **`monospaced bold-face`**.
`Command parameters`	Command parameter names are in *`monospaced italics`*.

We use the following ways to indicate important or additional information:

Note – Notes prevent commonly-made mistakes by pointing out important points.

Caution – Cautions prevent breaches of security, information loss, or system downtime. Cautions always contain critical information that you must observe.

Tip – Tips provide additional helpful information, such as alternative ways to complete steps.

Example Examples present a concrete scenario that clarifies the points made in the adjacent text.

Documentation Available

SMC documentation is divided into two main categories: Product Documentation and Support Documentation (page 14). Each SMC product has a separate set of manuals.

Product Documentation

The table below lists the available product documentation.

Table 1.2 Product Documentation

Guide	Description
Reference Guide	Explains the operation and features of the SMC comprehensively. Demonstrates the general workflow and provides example scenarios for each feature area. Available as separate guides for McAfee® Security Management Center and McAfee Firewall/VPN, and as a combined guide for McAfee IPS and McAfee Layer 2 Firewall.
Installation Guide	Instructions for planning, installing, and upgrading the SMC. Available as separate guides for McAfee Security Management Center and McAfee Firewall/VPN, and as a combined guide for McAfee IPS and McAfee Layer 2 Firewall.
Online Help	Describes how to configure and manage the system step-by-step. Accessible through the Help menu and by using the Help button or the F1 key in any window or dialog. Available in the Management Client and the Web Portal. An HTML-based system is available in the SSL VPN Administrator through help links and icons.
Administrator's Guide	Describes how to configure and manage the system step-by-step. Available as a combined guide for McAfee Firewall/VPN, McAfee IPS, and McAfee Layer 2 Firewall, and as separate guides for the SSL VPN and the IPsec VPN Client.
User's Guide	Instructions for end-users. Available for the IPsec VPN Client and the Web Portal.
Appliance Installation Guide	Instructions for physically installing and maintaining McAfee NGFW appliances (rack mounting, cabling, etc.). Available for all McAfee NGFW appliances.

PDF guides are available at https://www.stonesoft.com/en/customer_care/documentation/current/. The *McAfee SMC Administrator's Guide*, and the *Reference Guides* and *Installation Guides* for McAfee Security Management Center, McAfee Firewall/VPN, McAfee IPS, and McAfee Layer 2 Firewall are also available as PDFs on the Security Management Center DVD.

Support Documentation

The McAfee support documentation provides additional and late-breaking technical information. These technical documents support the SMC guide books, for example, by giving further examples on specific configuration scenarios.

The latest technical documentation is available at http://www.stonesoft.com/support/.

System Requirements

The certified platforms for running the McAfee Next Generation Firewall from Intel Security (NGFW) can be found at the product pages at http://www.stonesoft.com/en/products/appliances/.

The hardware and software requirements for the SMC and version-specific details for all software products can be found in the *Release Notes* available at http://www.stonesoft.com/en/customer_care/kb/.

Contact Information

For general information about SMC products, visit our web site at http://www.mcafee.com/.

CHAPTER 2

INTRODUCTION TO INTRUSION DETECTION AND PREVENTION

This chapter introduces and discusses the underlying security principles of Intrusion Detection Systems (IDS) and Intrusion Prevention Systems (IPS) in general. In this chapter we will discuss what IDS and IPS are, how they are used, what they are capable of, as well as what their possible weaknesses are.

The following sections are included:

The Role of IDS and IPS

Intrusion detection systems (IDS) and intrusion prevention systems (IPS) are an important part of a comprehensive security solution. An IDS or IPS provides additional layers of defense to the existing security infrastructure by detecting and reacting to attacks and security breaches. Both the terms IDS and IPS are used here:

- An IDS system triggers an alarm whenever something defined as anomalous is detected on the monitored network.
- An IPS system can additionally block unwanted traffic.

McAfee IPS can be configured either as an IDS or IPS or in *hybrid mode*, in which the system operates simultaneously in both IDS and IPS configuration.

To better understand the role of IDS/IPS, we can compare it to a firewall:

- A well-designed firewall stops everything that is not explicitly allowed, but IDS and IPS systems only react to unwanted traffic and allow everything else.
- As a rough generalization, firewalls mainly concentrate on the type, source, and destination of the traffic; the IDS/IPS system concentrates more on the information that is transferred.

The IDS/IPS system can look more deeply into traffic that has been allowed through the firewall or that is transferred between two hosts in the same network segment. For example, a connection from the Internet to a company's Web server is allowed through the firewall if it uses HTTP and proceeds according to the TCP protocol standards. However, this particular Web server contains an unpatched security vulnerability: an attacker in the know can include special code in a standard-looking URL. When this special code reaches the Web server, it causes an error condition that allows the attacker to execute some administrative commands. Even while the firewall allows this connection, an IPS system between the firewall and the Web server can stop it.

Generally speaking, there are two different classes of IDS/IPS: *host-based* and *network-based*. McAfee IPS is a network-based IDS and IPS system.

- Host IDS devices, such as Windows Defender, keep track of the system state changes on the hosts, alerting whenever something suspicious is detected. Host IDS is installed separately on each monitored host.
- Network-based IDS passively monitors network segments and notifies administrators when something out of the ordinary is detected. IDS is especially well suited for monitoring traffic that is sent and received within the same network segment so that it does not naturally flow through any single point of enforcement.
- Network-based IPS is located in the traffic path (known as *inline mode*) allowing an IPS device to actively filter out offending traffic. IPS allows direct blocking of attacks when you can identify a clear threat path. Most often, IPS devices are deployed in the traffic path between the Internet and internal network segments.

To an extent, the division between firewall and IPS systems is being blurred at the level of physical devices. A McAfee firewall can be used to prevent some attacks, since the McAfee IPS product's inspection checks for a limited number of protocols have been integrated into the McAfee Firewall/VPN product. On the other hand, McAfee IPS can be used for access control, a traditional firewall task.

IDS and IPS Detection Methods

The following general techniques are employed in IDS and IPS systems:

- *Misuse detection* is based on signatures of well-known attacks. The monitored traffic is compared to these signatures and whenever a matching pattern is found, a response is triggered. The main limitation of this approach is that an ever increasing number of attacks must each be created a precise and unique signature that does not match other traffic. Any attack that does not have a specific fingerprint goes unnoticed. Also, the traffic stream may have been altered so that it cannot be compared directly with the signatures. *Traffic normalization* is needed to remove the ambiguities from the traffic before it can be compared with the signatures. Traffic normalization also makes it possible to detect *evasions*, attempts to bypass the IDS or IPS system, and to introduce attacks, exploits, or other malware into the network.
- *Vulnerability detection* is based on publicly known vulnerabilities that affect systems, network devices, and applications. Attackers may exploit vulnerabilities in many ways, for example, to gain unauthorized access to networks and to confidential information. There are several databases of currently known vulnerabilities (for example, Common Vulnerabilities and Exposures maintained by the MITRE Corporation). Vulnerabilities are closely connected to signatures of known attacks used in misuse detection. However, one vulnerability may be related to several known attacks and variations of attacks, which means that detecting vulnerabilities is more effective than inspecting traffic only for known attack patterns.
- *Anomaly detection* uses statistics to find abnormal network behavior. For example, a denial-of-service (DoS) attack is simply an abnormally high number of otherwise normal-looking connection attempts and anomaly detection is an effective way to detect it. However, this approach alone does not allow very comprehensive inspection, since it can be very hard to define just what is normal and what is exceptional based on statistical information alone, especially considering that the ever-changing profile of network traffic makes ‘normal’ a moving target.
- *Protocol validation* checks whether traffic conforms to the protocol standards defined in the relevant *Request for Comments* (RFC). As many current attacks are based on some sort of protocol violation, any deviation from the defined protocol could be considered malicious. This approach is capable of detecting new types of attacks, also known as *zero-day attacks*. Unfortunately, sometimes there are also legitimate products that have been designed to violate the relevant standards either due to misinterpretation or purposely to circumvent a limitation.
- As an enhancement to protocol validation, *protocol anomaly detection* takes into account that often attacks do not actually violate the protocol specifications, but take advantage of inaccuracies in standards. For example, a standard may not specify any limitations for the size of a given protocol field. In practice, there are practical limits to the size of the field in legitimate use, so an unusually large value can be considered a protocol anomaly. Naturally, this type of protocol validation also carries a risk of catching unusual but legitimate traffic.
- Protocol validation can also take the different states of connections into account, adding the principle of *stateful inspection* to the IDS/IPS. For example, what is normal—by the TCP protocol standards—for an opening SYN packet of a connection is not necessarily acceptable in an ACK packet within a TCP exchange. There is a low risk of disturbing any legitimate traffic, since communications must generally follow the rules of the basic protocols or they will either be refused or handled incorrectly by the recipient.

Given the advantages and disadvantages of the different methods, a comprehensive IDS or IPS system should be capable of more than one type of analysis. McAfee IPS uses all of the methods described above to provide the best possible inspection coverage for the protected networks.

Challenges of Intrusion Detection

This page explains some of the negative effects IDS and IPS systems may produce, regardless of how well they work in principle. The key to avoiding most drawbacks is tuning the system to be relevant in the context of the installation environment. This is necessary in all environments. This becomes evident if you consider that even a real attempt to exploit a vulnerability may not be relevant when you consider the context of use: for example, an attempt to exploit a vulnerability in a Windows application is not a critical threat when launched at random against a Linux server.

False Positives

IPS can be an efficient tool that allows you to quickly receive notifications of events and even automatically block malicious traffic. If the patterns that the IPS is looking for are not accurate enough, also legitimate traffic may trigger alarms or be stopped.

False Negatives

False negatives occur when the IDS/IPS fails to react to malicious traffic. This can happen for many different reasons, for example:

- Skillful attackers may use techniques specifically designed to confuse the IDS/IPS.
- The inspection policies may be configured too permissive when trying to eliminate false positives.
- The IDS/IPS may not receive exactly the same traffic as the hosts on the monitored network due to network configuration issues.
- The lack of frequent updates may leave the IDS/IPS lacking the attack signatures that detect the latest publicly known exploits.

Denial of Service

IDS and IPS may be vulnerable to denial-of-service (DoS) attacks. If the engines are busy processing a flood of packets, the attacker may succeed in slipping in some malicious traffic without the IDS/IPS being able to detect it.

In addition, an IDS/IPS configured with responses such as connection resetting or IP address blacklisting could be turned into an effective DoS tool by an attacker. If the administrators have not considered this possibility when designing the responses, an attacker could generate malicious traffic that makes the IDS/IPS stop legitimate traffic, for example, through IP address spoofing.

Overloaded Administrators

Some organizations, particularly MSSPs and other large organizations, will attract a high number of security incidents. Administrators may be engaged in investigating many events at the same time. If the IDS/IPS system is difficult to maintain and keep track of, the administrators will be overwhelmed and unable to respond to all incidents in a timely manner.

CHAPTER 3

INTRODUCTION TO MCAFEE NGFW IN THE IPS AND LAYER 2 FIREWALL ROLES

This chapter gives you an overview of the system architecture of McAfee Next Generation Firewall from Intel Security (NGFW) in the IPS and Layer 2 Firewall roles, and how the engines inspect traffic.

The following sections are included:

The McAfee NGFW Solution

The McAfee NGFW engines in the IPS and Layer 2 Firewall roles are part of the McAfee NGFW solution, which is especially well-suited to complex and distributed network environments. The IPS component provides intrusion detection and prevention. The Layer 2 Firewalls provide access control and deep inspection of traffic. In addition to the IPS engines and the Layer 2 Firewalls, the McAfee NGFW solution also supports clustered high availability Firewalls and virtual private networking (VPNs).

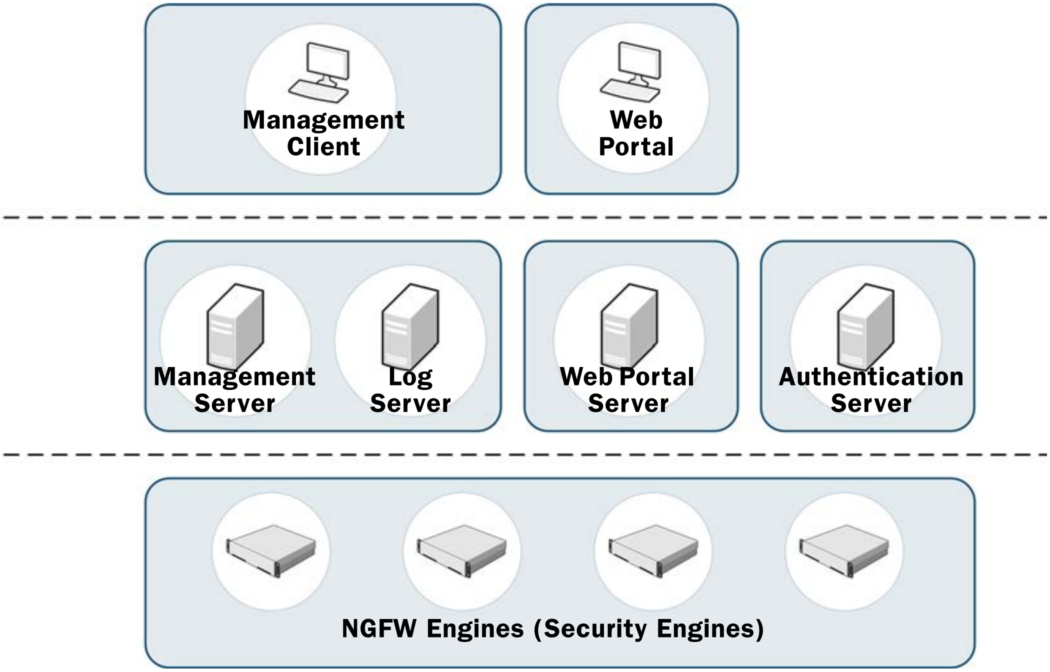
The configuration, monitoring, and control of the system is done through a centralized Security Management Center (SMC) that provides a single point of contact for a large number of geographically distributed administrators. The unified management platform provides major benefits for organizations of all sizes:

- Interaction between the McAfee Firewall/VPN, McAfee IPS, McAfee Layer 2 Firewall, Master Engine, and Virtual Security Engine components in the same SMC creates security benefits by allowing automatic coordinated responses when a security threat is detected, providing instant blocking of unwanted traffic, and reducing the need for immediate human intervention.
- Multiple administrators can log in at the same time to efficiently configure and monitor all McAfee NGFW engines. The SMC provides a single user interface that allows unified configuration, monitoring, and reporting of the whole McAfee NGFW solution with the same tools and within the same user session.
- The reuse of configuration information across components allows you to avoid the laborious duplicate work of configuring the same details for all components individually or exporting and importing the configurations between multiple SMCs.
- The SMC is designed to manage large installations and to be geographically distributed, so it is flexible and allows scaling up the existing components and adding new types of components without sacrificing its ease-of-use.

SMC Components

The SMC components and their roles are illustrated below.

Illustration 3.1 SMC Components



One Security Management Center can manage a large number of Security Engines, Master Engines, and Virtual Security Engines. The distributed architecture allows deploying the components effectively in different network environments. You can flexibly add, remove, and reposition SMC components according to your needs.

Table 3.1 SMC Components

Component	Description
Management Clients	Provide a user interface for configuring, controlling, and monitoring the system. Connect to the Management Server.
Management Servers	Store all configuration data, relay commands to the engines, and notify administrators of new alerts in the system.
Log Servers	Store logs and correlate events detected by multiple Security Engines.
Web Portal Servers	Provide restricted viewing of configuration information, reports, and logs.
Authentication Servers	Provide user linking and user authentication services for end-user and administrator authentication.

Table 3.1 SMC Components (Continued)

Component	Description
NGFW Engines (Security Engines)	Inspect and filter traffic. Correlate events in traffic inspected by the engine itself. NGFW engines are represented by Security Engine elements in the SMC. NGFW engines that have a license that allows the creation of Virtual Resources can be uses as a Master Engine to provide resources for Virtual Security Engines. See Master Engine and Virtual IPS Configuration (page 77) and Master Engine and Virtual Layer 2 Firewall Configuration (page 83).

All communications between SMC components are authenticated and encrypted. The engines work independently according to their installed configuration, so even if the connections to the Security Management Center are cut, traffic inspection continues without interruption.

IPS Engines and Layer 2 Firewalls

An IPS engine or a Layer 2 Firewall is responsible for picking up and examining network traffic in real time. The engines perform event correlation and analysis for traffic they inspect. The engjnes can also initiate immediate responses to any threats that they detect. Depending on how they are installed, engines may also block traffic based on commands that other components send.

IPS engines and Layer 2 Firewalls can be flexibly scaled up to form clusters of up to 16 devices that work as a single virtual entity. Clustering IPS engines provides additional performance and high availability for the traffic inspection service. Only one Layer 2 Firewall node in the Layer 2 Firewall Cluster is active at a time. The other Layer 2 Firewall nodes remain in standby mode. If the active Layer 2 Firewall node fails, one of the standby nodes automatically starts processing traffic.

The IPS engines and Layer 2 Firewalls include an integrated operating system (a specially hardened version of Linux). There is no need for separate operating system patches or upgrades; all software on the engines is upgraded during the NGFW software upgrade.

Accuracy

Effective response to network security incidents requires the capability to recognize an enormous number of possible threats. On the other hand, the IPS system must not produce a high number of false alarms that engage the system administrators in needless investigations or automatically stop legitimate business communications.

To provide the best possible accuracy, the IPS and Layer 2 Firewall engines provide multiple detection methods that complement each other. Attack signatures are supplemented with protocol-specific matching to produce accurate fingerprints of attacks. The observations on network traffic are not passed on to administrators directly, but instead collected together for further analysis and combined presentation.

What is considered to be a serious threat to a crucial system in one environment may not be considered an event at all in another network. There is no one set of traffic inspection policies that would work ideally in every environment, so IPS and Layer 2 Firewall provides detailed customization possibilities for the entire inspection process. The efficient configuration tools provide default policies that can be edited using drag-and-drop, while still allowing highly detailed controls for advanced configuration.

With accurate detection results, efforts can be concentrated on countering real threats instead of working on analyzing an endless stream of false alarms.

Manageability

Everyone recognizes the value of having the right tools for the right task, but unfortunately this is not always a given. IPS engines and Layer 2 Firewalls provide professional network administrators the tools they need to save time, reduce the likelihood of mistakes, and get a big picture of what is happening in the network.

While ease-of-use is one of the main goals for the product, IPS engines and Layer 2 Firewalls do not achieve it by cutting the available features. The system provides extensive inspection process tuning possibilities, detailed information for monitoring, advanced automation, and tools for complete remote management (including all software upgrades). The distributed architecture allows components to be located on separate machines and in different networks, even in different countries and continents–and still be easily managed as a single system.

An easy-to-use system makes helps the administrators’ concentrate on investigating the security threats instead of configuring the security systems.

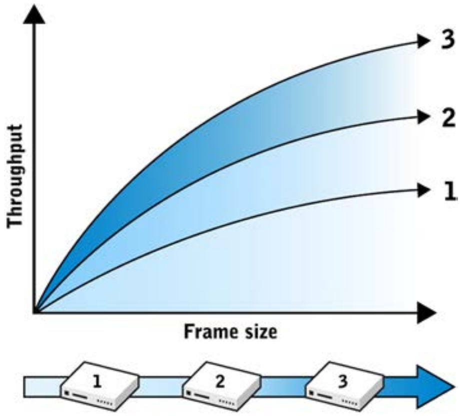
Scalability and High Availability

The constant introduction of new tools and services that rely on network access means that the volume of network traffic will keep growing over time even in companies that are not otherwise expanding. Also the IPS engines must be scalable to meet the growing demands and high-availability. Clustering answers this demand by combining devices together in a single, virtual entity.

Clustering also provides high availability for the intrusion detection services. If the system is down when an attack is made, attackers have a head start, allowing them to penetrate deeper into the network and cover up their tracks. If a node in an IPS Cluster stops processing traffic, the inspection can switch over to the other remaining IPS node transparently. If the active Layer 2 Firewall node in a Layer 2 Firewall Cluster fails, one of the standby nodes automatically takes over traffic inspection. This way, it is also possible to conduct online maintenance of individual IPS engines without corresponding interruptions in traffic inspection.

IPS engines can be transparently clustered together into a single virtual entity in which the performance of each node contributes to the total throughput. Clustering IPS engines allows new devices to be added flexibly as traffic volumes grow while retaining the existing equipment and configurations. Clustering improves performance by balancing the load intelligently between the clustered IPS nodes (Illustration 3.2).

Illustration 3.2 The Performance Benefits of Clustering IPS Engines



Optional high availability measures are also available for the Security Management Center. See the *McAfee SMC Reference Guide* for more information.

Scalability and high availability ensure that the system can adapt to growing needs, simplifies planned maintenance, and protects against hardware failure.

Disconnect Mode for IPS Engines and Layer 2 Firewalls

When IPS engines or Layer 2 Firewalls are deployed in inline mode, link failures may cause significant delays in traffic transfer if the link failure is not detected quickly enough. IPS engines and Layer 2 Firewalls support disconnect mode, which enables constant monitoring of link connections and minimizes delays. If a link goes down on one side of a pair of Inline Interfaces, the IPS engine or Layer 2 Firewall detects this, simulates cable disconnection on the other side, and takes the other side's link transmitter (TX) down. The IPS engine or Layer 2 Firewall continues to monitor the receiver (RX) side of a pair of Inline Interfaces. It detects when the link is up again and brings the transmitter (TX) back up accordingly. Disconnect mode is active by default on all IPS appliances that support the feature and on all McAfee NGFW appliances that are used in the IPS or Layer 2 Firewall role.

How IPS Engines and Layer 2 Firewalls Inspect Traffic

The are two main phases in the traffic inspection process. First, the IPS engine or Layer 2 Firewall inspects the network traffic for any anomalies. If the engine detects ambiguities in the traffic, it normalizes the traffic before inspecting it further. Traffic normalization also helps the engine in detecting attempts to use evasion techniques (for example, packet fragmentation, TCP segmentation, or combinations of evasions) to bypass inspection altogether.

In this process, known attacks are detected through attack signatures that are augmented with protocol awareness to form powerful attack *fingerprints*. Protocol awareness decreases the number of false positives compared to simple signatures. Each pattern is applied only to the correct type of traffic; for example, an attack that uses HTTP can be detected when the pattern is seen in HTTP traffic, but does not falsely match an e-mail message header transported over SMTP.

While fingerprinting accurately detects known attacks, it does not detect attacks that are not yet known. IPS and Layer 2 Firewall engines provide two types of anomaly detection to complement fingerprinting: *protocol analysis* and *statistical anomaly detection*.

- *Protocol analysis* identifies violations in network communications, such as unexpected data, wrong connection states, and additional or invalid characters. Detecting such violations is useful because many attacks purposely violate standards to trigger abnormal operating responses in vulnerable target systems.
- S*tatistical anomaly detection* gathers traffic statistics to detect events such as slow scans, unusual number of connections, and so on. This method tracks patterns based on frequency and sequence of events, or the occurrence of sets of related events within a period of time. For example, many connection attempts from one host to many ports and IP addresses is certainly a network scan of some kind.

How IPS Engines and Layer 2 Firewalls Respond to Incidents

There are various automatic responses that an IPS engine and a Layer 2 Firewall can take when it detects traffic of interest. For example, they can log the connection or actively filter out the traffic.

Several responses are available:

- As the mildest response, an event can be logged. The log entries can be used, for example, for generating statistical reports. This may be appropriate, for example, for tracking trends in normal network traffic patterns.
- A step up from a log entry is to generate an alert entry that can be escalated to administrators through multiple configurable alert channels including e-mail, mobile phone text messaging (SMS), and SNMP, in addition to being used like log entries.
- Additionally, logs and alerts can record the full packet headers and data payload for further analysis.

Note – Storing or viewing the packets' payload may be illegal in some jurisdictions due to laws related to the privacy of communications.

- Blacklisting makes it possible to block unwanted network traffic for a specified time. IPS engines and Layer 2 Firewalls can add entries to their own blacklists based on events in the traffic they inspect. They can also send blacklist requests to other Security Engines. Connections that match the blacklist are mainly stopped (depending on the enforcing component's policy).

The available responses on an IPS engine or Layer 2 Firewall depend on the engine's physical configuration. There are two possible configurations for IPS engines, both of which can be used on the same IPS engine at the same time (on different network interfaces). There are three possible configurations for Layer 2 Firewalls. The following table describes the installation modes for IPS engines and Layer 2 Firewalls.

Table 3.2 Installation Modes for IPS Engines and Layer 2 Firewalls

NGFW Role	Mode	Description
IPS	Inline	In an inline installation, the traffic flows through the IPS engine. The IPS engine has full control over the traffic flow and can be used to automatically block any traffic. An inline IPS engine can also enforce blacklisting commands received from other components. Fail-open network cards can be used to ensure traffic flow is not disrupted when the IPS engine is offline. An inline IPS engine also provide access control and logging for any Ethernet traffic (layer 2).
	Capture	In a capture installation, external equipment duplicates the traffic flow for inspection, and the IPS engine just "listens in". The IPS engine does not have direct control over the traffic flow, but it can respond to selected threats by sending packets that reset the connections. An IDS-only IPS engine can send blacklisting requests to other IPS engines, Layer 2 Firewalls, or Firewalls, but it cannot enforce blacklisting requests from other components.

Table 3.2 Installation Modes for IPS Engines and Layer 2 Firewalls (Continued)

NGFW Role	Mode	Description
Layer 2 Firewall	Inline	In an inline installation, the traffic flows through the Layer 2 Firewall. The Layer 2 Firewall has full control over the traffic flow and can be used to automatically block any traffic. An inline Layer 2 Firewall can also enforce blacklisting commands received from other components. An inline Layer 2 Firewall also provides access control and logging for any Ethernet traffic (layer 2).
	Capture (Passive Firewall)	In a capture (Passive Firewall) installation, external equipment duplicates the traffic flow for inspection to the Layer 2 Firewall, and the Layer 2 Firewall just “listens in”. The Layer 2 Firewall does not have direct control over the traffic flow, but it can respond to selected threats by sending packets that reset the connections. A Layer 2 Firewall in Passive Firewall mode can send blacklisting requests to other Layer 2 Firewalls, IPS engines, or Firewalls, but it cannot enforce blacklisting requests from other components.
	Passive Inline	In a passive inline installation, the traffic flows through the Layer 2 Firewall, but the Layer 2 Firewall is configured to only log connections. A Layer 2 Firewall in Passive Firewall mode can send blacklisting requests to other Layer 2 Firewalls, IPS engines, or Firewalls, but it cannot enforce blacklisting requests from other components.

Chapter 4

NGFW Deployment in IPS and Layer 2 Firewall Roles

This chapter provides general guidelines for deploying McAfee Next Generation Firewall from Intel Security (NGFW) in the IPS and Layer 2 Firewall roles, and for deploying Master Engines. It also illustrates some typical deployment scenarios.

The following sections are included:

Supported Platforms

McAfee NGFW engines in the IPS and Layer 2 Firewall roles can be run on the following general types of platforms:

- Purpose-built McAfee NGFW appliances.
- Standard Intel-compatible servers. Search for the version-specific *Hardware Requirements* at http://www.stonesoft.com/en/customer_care/kb/.
- Virtualization platforms that support the deployment of Open Virtual Format (OVF) templates. VMware is officially supported. Other virtualization platforms may also be supported. More information can be found in the *McAfee NGFW Installation Guide for IPS and Layer 2 Firewall Roles*.

The NGFW engine software includes an integrated, hardened Linux operating system. This eliminates the need for separate operating system installation, configuration, and patching.

NGFW appliances can also be used as Master Engines to host Virtual IPS engines or Virtual Layer 2 Firewalls. If the appliance has a license that allows the creation of Virtual Resources, the appliance can optionally be used as a Master Engine that provides resources for Virtual IPS engines or Virtual Layer 2 Firewalls.

- For more information about Master Engines and Virtual IPS Engines, see Master Engine and Virtual IPS Configuration (page 77).
- For more information about Master Engines and Virtual Layer 2 Firewalls, see Master Engine and Virtual Layer 2 Firewall Configuration (page 83).

Master Engines can be run on the following general types of platforms:

- Purpose-built McAfee NGFW appliances with 64-bit architecture.
- Standard Intel-compatible servers with 64-bit architecture.

General Deployment Guidelines

Table 4.1 summarizes the general deployment guidelines for IPS engines, Layer 2 Firewalls, and the McAfee Security Management Center. Naturally, there are valid reasons to make exceptions to these general rules depending on the actual network environment.

Table 4.1 General Guidelines for IPS and Layer 2 Firewall Deployment

Component	General Guidelines
Management Server	Position on a central site where it is physically accessible to the administrators responsible for maintaining its operation.
Log Servers	Place the Log Servers centrally and/or locally on sites as needed based on log data volume, administrative responsibilities, etc.
Management Clients	Management Clients can be used from any location that has network access to the Management Server and the Log Servers.
IPS Engines	Position IPS engine(s) at each location so that traffic in all the appropriate networks can be inspected. IPS engines can be clustered. Functionally, the IPS Cluster is equal to a single high-performance IPS engine. Cluster deployment involves setting up a heartbeat link between the IPS engines that allows the devices to track each others' operating status and agree on the division of work.
Layer 2 Firewalls	Position Layer 2 Firewall(s) at each location so that traffic in all the appropriate networks can be inspected. Layer 2 Firewalls can be clustered for high availability. Only one Layer 2 Firewall node in the Layer 2 Firewall Cluster is active at a time. If the active Layer 2 Firewall node goes offline, another Layer 2 Firewall node automatically starts processing traffic.
Master Engines	Position Master Engine(s) at locations where Virtual Security Engines are needed, such as at a hosting location for MSSP services, or between networks that require strict isolation. Master Engines can be clustered. A clustered Master Engine provides scalability and high availability. In a Master Engine Cluster, the Virtual Resource is active in one Master Engine at a time. Cluster deployment involves setting up a heartbeat link between the engines that allows the devices to track each others' operating status, agree on the division of work, and exchange information on traffic.

Positioning IPS Engines and Layer 2 Firewalls

IPS and Layer 2 Firewall engines pick up passing network traffic for inspection in real time. The positioning of the engines is the most critical part of the deployment. Each engine can inspect the network traffic of one or more network segments in IDS and IPS configurations.

The following table describes the modes for IPS engines and Layer 2 Firewalls.

Table 4.2 Modes for IPS Engines and Layer 2 Firewalls

NGFW Role	Default Policy	Mode	Description
IPS	Allows everything that is not explicitly denied in the policy.	Inline	In inline (IPS) mode, an IPS engine actively filters traffic. The IPS engine is connected as a “smart cable” between two network devices, such as routers and a switch. The IPS engine itself does not route traffic: packets enter through one port, are inspected, and exit through the other port that makes up the pair of Inline Interfaces. Fail-open network interface cards (NICs) are recommended on the IPS engine to allow network connectivity when the IPS engine is offline. An inline IPS engine can also transparently segment networks and control network access.
		Capture	In capture (IDS) mode, an IPS engine listens to network traffic that is replicated to the IPS engine through port mirroring (switch SPAN ports) or through dedicated network TAP devices.
Layer 2 Firewall	Denies everything that is not explicitly allowed in the policy.	Inline	In inline (IPS) mode, a Layer 2 Firewall engine actively filters traffic. The engine is connected as a “smart cable” between two network devices, such as routers. The engine itself does not route traffic: packets enter through one port, are inspected, and exit through the other port that makes up the pair of Inline Interfaces. Fail-open network interface cards (NICs) can only be used on the Layer 2 Firewall if the Failure Mode of the pair of Inline Interfaces is Normal. An inline Layer 2 Firewall can also transparently segment networks and control network access.
		Capture (Passive Firewall)	In capture (Passive Firewall) mode, a Layer 2 Firewall listens to network traffic that is replicated to the Layer 2 Firewall through port mirroring (switch SPAN ports).
		Passive Inline	In passive inline mode, a Layer 2 Firewall is installed directly inline between two network devices, such as routers and a switch, but does not actively filter traffic. An inline Layer 2 Firewall can be set to Passive Firewall mode by configuring the Layer 2 Firewall to only log connections.

The same IPS engine can be used for both IPS and IDS operation simultaneously. For example, an IPS engine can be deployed inline to examine traffic from one network to another and additionally capture traffic that stays within each network.

Take the following into consideration when you decide where to install the engines:

- The critical assets to be protected and the potential attack paths.
- The most suitable locations along the attack path for detecting and responding to attack attempts in order to protect the assets.
- The volume and profile of traffic to be inspected at each location.

Select the engine role based on the way the engine should deal with inspected traffic:

- Use a Layer 2 Firewall if traffic must be denied unless it is explicitly allowed.
- Use an IPS engine if traffic must be allowed unless it is specifically denied.

Illustration 4.1 Example of Positioning Engines in Different Network Segments

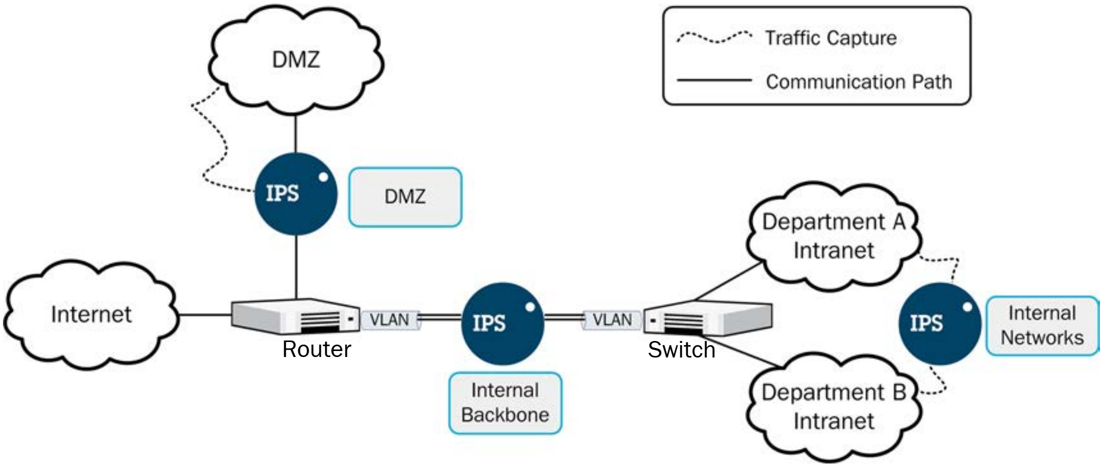


Illustration 4.1 outlines common deployment scenarios for IPS engines in general internal networks and in DMZ networks. Layer 2 Firewalls can be used in similar scenarios. The next few pages of this guide explain these scenarios in more detail. IPS engines and Layer 2 Firewalls are not necessarily needed at each of these points in all environments. A single IPS engine or a single Layer 2 Firewall can also cover several or even all scenarios simultaneously if the physical setup makes it practical.

Internal Networks

In internal networks, access is generally quite permissive for purely internal communications, but there are strict controls at the perimeter firewall that separates the internal network from public networks. Inbound traffic from public networks to internal networks is generally forbidden with few exceptions.

Table 4.3 Internal Network Considerations for IPS Engines and Layer 2 Firewalls

	Description	Considerations for IPS Engines and Layer 2 Firewalls
Main purpose	Network services and connectivity for authorized users. Back-end servers that serve other networks and user groups.	IPS engines and Layer 2 Firewalls can be used within internal networks and for strengthening the perimeter defense with additional layers of inspection.
Hosts	Mixed environment consisting of servers, laptops, desktops, network printers, copiers, etc.	IPS engines and Layer 2 Firewalls can control access between internal hosts that are not controlled by other devices. Connections between internal network zones are of particular interest for inspection.
Users	Authorized personnel. Access in and out of the network is controlled by a Firewall.	End-user-controlled devices can be distinguished from other hosts to create more accurate and fine-grained rules.
Traffic volume	Varies from low to high. Grows highest at network choke-points in large environments.	Installation at network choke-points where traffic levels are high requires high-performance hardware. Clustering and load balancing can be applied to increase performance and provide high availability in critical locations.
Traffic type	Diverse with a large number of different applications communicating within and in/out of the network.	A wide range of permitted applications means that the policy has a wide scope. Access control and inspection can be fine-tuned based on the security levels of the different network segments or zones. TLS inspection can be activated to inspect SSL/TLS encrypted traffic. The IPS engines and Layer 2 Firewalls can also detect and control Application use.
Network security	A "trusted network" where the users and the traffic are considered to be authorized.	The primary line of defense is at the perimeter, but it is possible that authorized users in the trusted network become willingly or accidentally involved in a security incident.

DMZ Networks

DMZ networks (demilitarized zone networks, also known as perimeter networks) allow inbound access to a wide range of users, but are quite unified environments in terms of devices. The services offered are limited in number as well and their allowed usage is often quite strictly defined.

Table 4.4 DMZ Considerations for IPS Engines

	Description	Considerations for IPS Engines
Main purpose	DMZs provide a limited number of services for external users. The services are often business-critical and open for public access.	DMZs are a tempting target for attacks because of their accessibility, importance, and visibility. IPS engines provide crucial protection in DMZs, unless the DMZs are already protected by firewalls.
Hosts	Often a uniform environment consisting mainly of servers. No outbound communication is usually initiated from the DMZ to the public networks.	Most sources are not trusted and IP address spoofing is a possibility. Internal networks may be considered more trustworthy if there is a Firewall that prevents IP address spoofing.
Users	Most services are public, but some services may also be offered to specific users. Administrators have wider privileges.	If the users can be reliably recognized, allowed and forbidden activities can be specified in great detail for each type of access.
Traffic volume	Low to medium, generally the full bandwidth of all Internet links combined (shared with other local networks). Traffic to other local networks may be high in volume.	Hardware requirements vary greatly depending on the environment. Clustering allows flexible adjustments to the inspection performance.
Traffic type	Rather uniform traffic, with only well-known applications and servers communicating within and into the networks.	The limited, well-defined set of protocols and applications means inspection can be tuned in great detail. If servers provide HTTPS services, decrypting the traffic for inspection may require heavy processing.
Network security	A network between the trusted and untrusted security zones allowing access for authorized and public use.	External access to services makes the servers in a DMZ a tempting target for attacks. Connections between the DMZs and other networks facilitate further attacks.

Positioning Security Management Center Components

The McAfee Security Management Center (SMC) consists of the Management Server and the necessary number of Log Servers and Management Clients. These can be positioned flexibly according to need. One Management Server can remotely manage a high number of Firewall, IPS, and Layer 2 Firewall engines. Optionally, you can install one or more additional Management Servers for a high availability setup. Only one Management Server is active at a time. The additional Management Servers function as standby Management Servers. You can also install one or more Web Portal Servers for view-only access to the system. Firewall, IPS, and Layer 2 Firewall engines are managed through the Management Server(s), so the Management Client never connects directly to the engines.

The general SMC deployment steps are as follows:

1. Position the Management Server at a central location where it can access all the other components and so that the Management Clients can connect to it.
2. Position Log Servers centrally and/or locally based on the log data volume requirements.
3. Position Management Clients freely where they are needed.

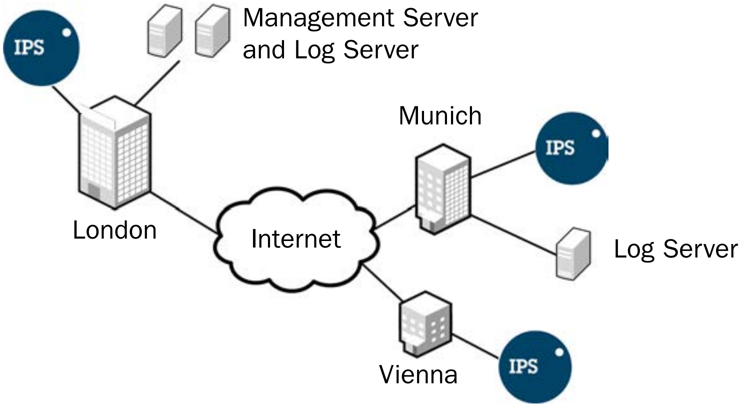
The SMC deployment considerations are described in detail in the *McAfee SMC Reference Guide*.

IPS Deployment Example

This section shows one way to deploy McAfee NGFW in the IPS role in an organization. The scenario presented here is not meant to be representative of a typical installation. The main focus here is to highlight some of the criteria that can be used in planning deployment. The example covers considerations that affect most installations, but is not an exhaustive list of all factors that you may need to consider. The IPS system could be deployed in alternative ways even in this example scenario, depending on issues that are not covered here, such as the physical layout of the individual local networks, the hardware available, and budget constraints.

This example explains the IPS deployment at a company that has three offices: headquarters in London, a branch office in Munich and a small satellite office in Vienna.

Illustration 4.2 The Example Company's Networks



All offices have IPS components. There are also SMC components at the two larger sites. The example company has some critical assets to protect and some of the networks experience a heavy traffic load. The example company has decided on a high-availability solution for most locations and acquired the following components:

- 3 IPS Engines
- 1 Management Server
- 2 Log Servers

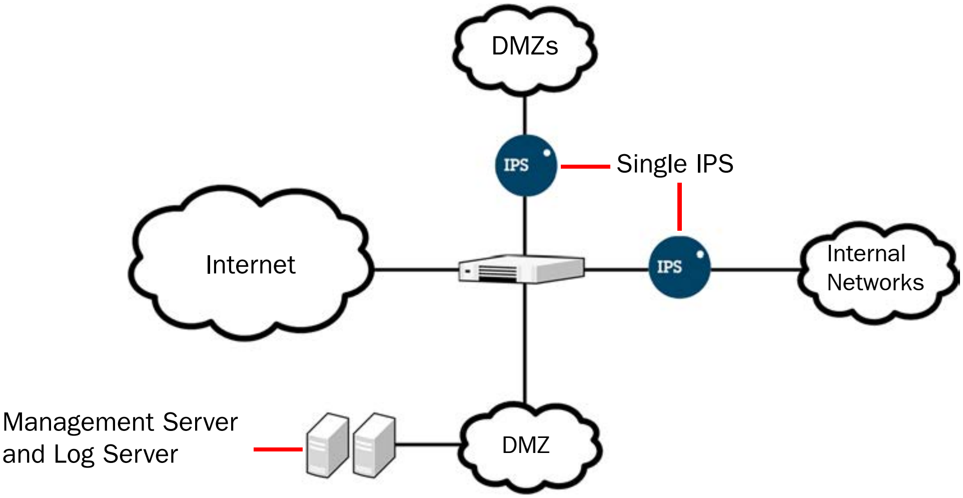
The next few pages show the placement of these components office by office and explain the criteria that the example company used to select them.

Example Large-Scale Installation

London

The example company's main office at London has a large number of end-users and servers. The servers host nearly all of the company's external services and receive a high volume of traffic. The large end-user base generates a high volume of network traffic as well. There are many different applications and protocols in use, resulting in a very diverse traffic pattern. The most important asset that the company wants to protect at its headquarters are the web servers hosting the company's online store. Because this is the main office, the main system administrators work at this site.

Illustration 4.3 Headquarters at London



In this case, the company has made the following decisions:

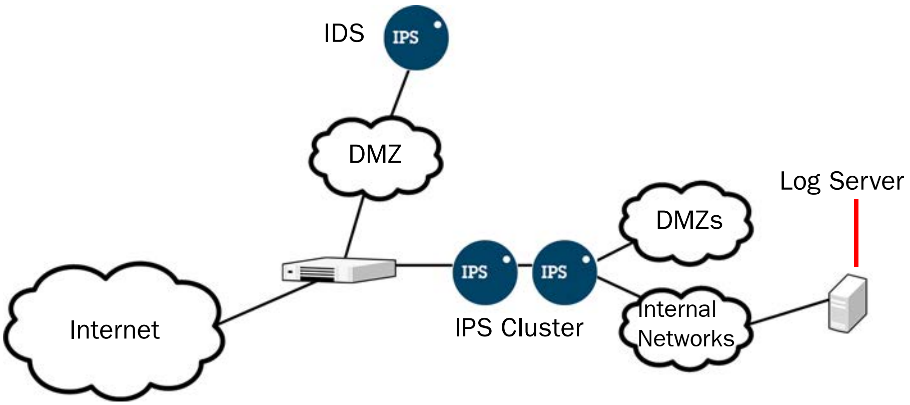
- Because most of the administrators are at this site, the Management Server that controls the whole distributed system is located here.
- There are many administrators and components, so there is also a Log Server here.
- Several DMZs for different services handle a high total volume of traffic. Part of the traffic is encrypted HTTPS, which uses significant processing power to decrypt for inspection. As the overall load is heavy, the company decided to protect the DMZs using a dedicated high-performance NGFW appliance.
- A separate single IPS is installed to protect the diverse high-volume communications of the internal networks.
- The Management Server and the Log Server are placed in a dedicated DMZ for security.

Example Medium-Scale Installation

Munich

The example company’s branch office at Munich has a moderate number of end-user clients. Although some services are only offered at the London headquarters and used remotely through a VPN, there are still many local servers, mostly for internal and partner use. There are also some administrators at this location who are responsible for the daily upkeep of the infrastructure at this office and the small satellite office in Vienna.

Illustration 4.4 Large Branch Office in Munich



In this case, the company has made the following decisions:

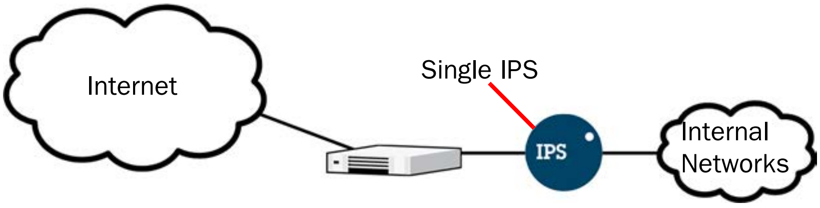
- Because there are administrators who browse logs extensively at this site, there is a dedicated Log Server here.
- One IDS is installed to inspect the network traffic in a DMZ that supports partner access.
- The IPS Cluster is placed in a dedicated DMZ for security.

Example Small-Scale Installation

Vienna

The example company’s small satellite office at Vienna has a relatively low number of end-user clients, and there are no servers of any major significance. Users rely mostly on the services at the Munich office, which they access through a VPN. Additionally, the users have direct Internet access for general web browsing. There are no local administrators; systems are managed remotely by administrators in Munich.

Illustration 4.5 Small Satellite Office in Vienna



In this case, the company has decided to install a Single IPS at the office to inspect the relatively low-volume traffic that the end-users' Internet and VPN-bound communications generate. Since there are no local administrators and the traffic volumes are low, the logs are sent to the Munich Log Server, making it quicker and easier for the responsible administrators there to view and manage the data.

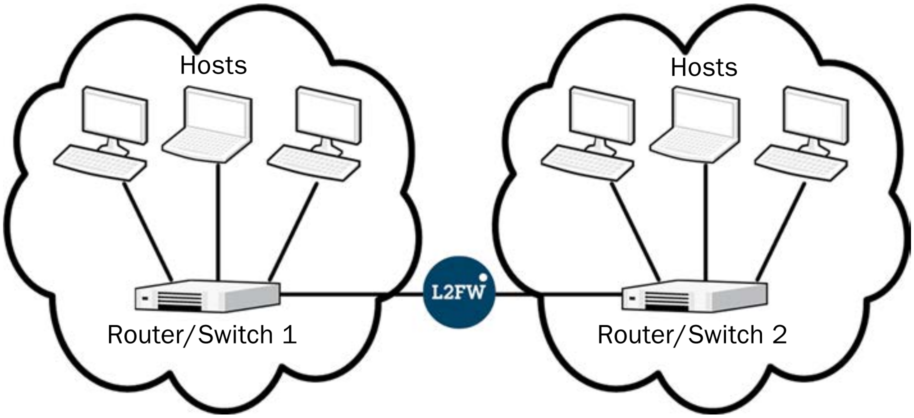
Layer 2 Firewall Deployment Example

This section shows an example of deploying Layer 2 Firewalls in an organization. The scenario presented here is not meant to be representative of a typical installation. The main focus here is to highlight some of the criteria that can be used in planning deployment. The example covers considerations that affect most installations, but does not comprise an exhaustive list of all the factors that you may need to consider. The Layer 2 Firewalls could be deployed in alternative ways even in this example scenario, depending on issues that are not covered here, such as the physical layout of the individual local networks, the hardware available, and budget constraints.

Example of Using a Single Layer 2 Firewall

This example uses a Single Layer 2 Firewall in an organization that has a very large internal network. Administrators want to prevent hosts connected to different switches in the same network segment from communicating directly at the protocol level. Using the Layer 2 Firewall makes it possible to implement access control for any Ethernet protocols between switches within the same network segment. There is no need to modify the network topology.

Illustration 4.6 Single Layer 2 Firewall in an Intranet



Network Configuration Scenarios for IPS Engines

This section presents the most important scenarios for deploying a Single IPS or an IPS Cluster in different network configurations.

Deploying IPS Engines in an IDS Configuration

One of the options in IDS mode is to use network TAP devices that copy packets for the IPS engines. In an IPS Cluster, all nodes must receive all packets. The nodes agree over the heartbeat link which node inspects which connections.

Illustration 4.7 Single IPS (Left) and IPS Cluster in IDS Mode With Network TAPs (Right)

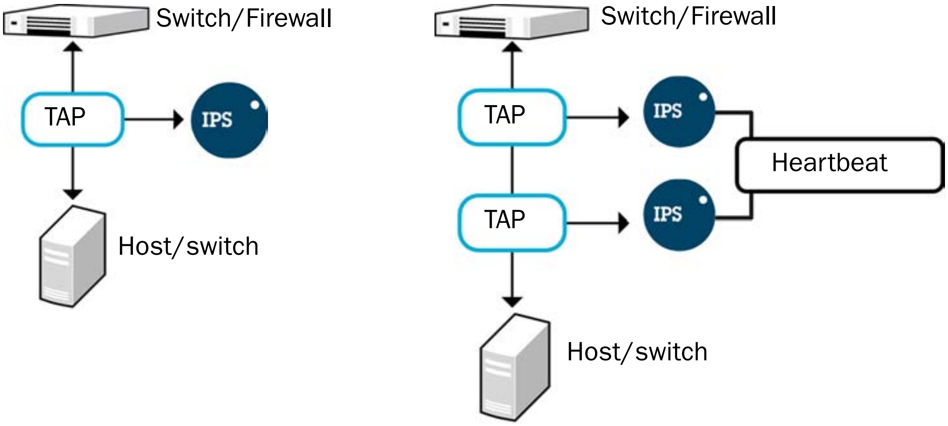


Illustration 4.8 Single IPS in IDS Mode With Network TAP and an Interface for Sending Resets

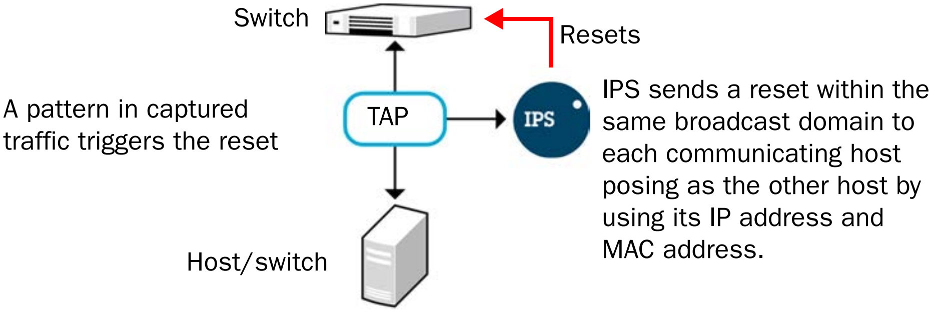
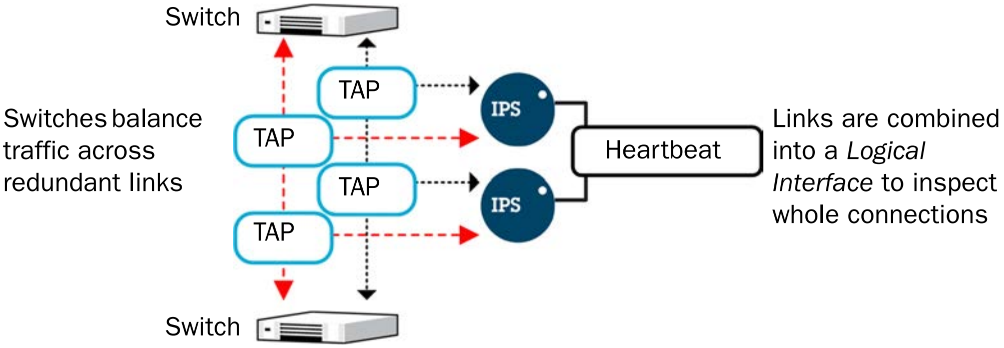

Switch
Resets
A pattern in captured traffic triggers the reset
TAP
IPS
IPS sends a reset within the same broadcast domain to each communicating host posing as the other host by using its IP address and MAC address.
Host/switch

Illustration 4.9 IPS Cluster in IDS Mode With Network TAPs on a Redundant Link

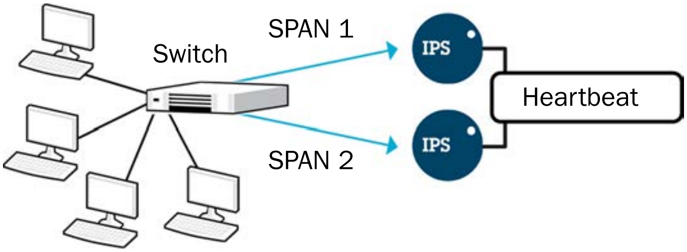


Switches balance traffic across redundant links

Links are combined into a *Logical Interface* to inspect whole connections

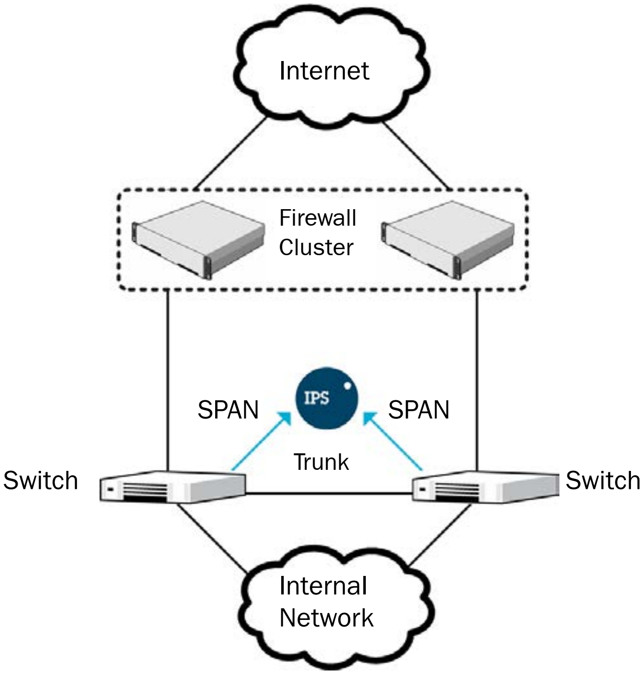
Packets can also be duplicated for inspection through a SPAN or mirror port on a switch/router. In an IPS Cluster, each node must be connected to a SPAN or mirror port of its own. A hub can be used to achieve a similar configuration when the low performance of a hub is not an issue (for example, in a basic testing environment), but hubs are generally not recommended.

Illustration 4.10 IPS Cluster in IDS Mode With SPAN/Mirror Ports



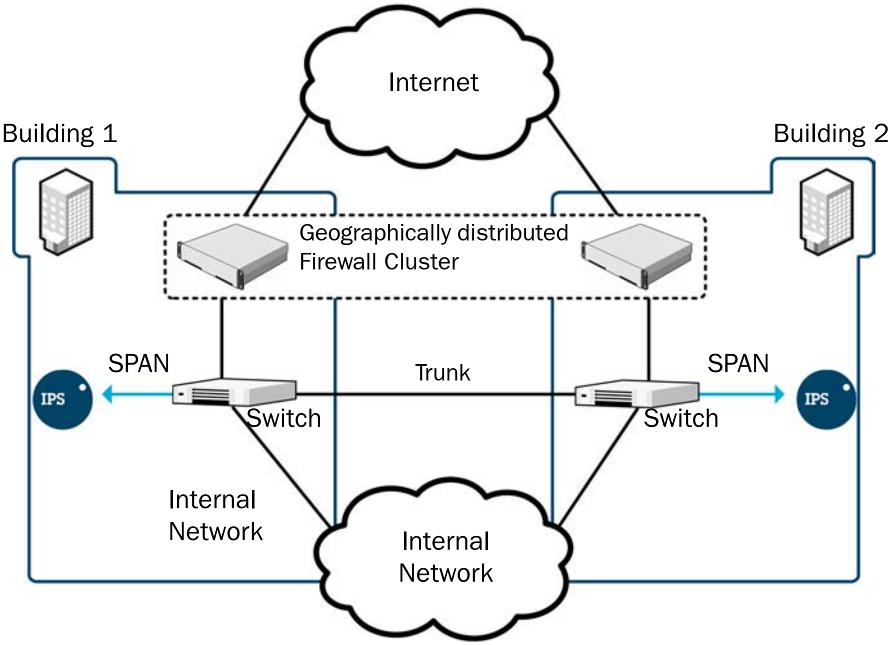
An IPS Cluster can be deployed alongside a Firewall Cluster. In this configuration, the IPS Cluster is in the same broadcast domain as the Firewall.

Illustration 4.11 IPS Connected to SPAN Ports Alongside Redundant Switches



In a redundant disaster-recovery setup, Firewall Cluster nodes can be far apart. The IPS engines are not clustered in this configuration, but they have identical policies.

Illustration 4.12 Single IPS Engines in a Distributed Disaster-Recovery Environment



Deploying IPS Engines in an IPS Configuration

In an inline IPS configuration, the IPS engines are installed directly in the traffic path. Fail-open network cards are recommended to allow traffic flow when the IPS engines are offline.

Caution – Always use standard cabling methods with an inline IPS engine. Use crossover cables to connect the appliance to hosts and straight cables to connect the appliance to switches. See Cabling Guidelines (page 47).

Illustration 4.13 Basic Inline Installations: Single Inline IPS Engine (Left) and Serial IPS Cluster (Right)

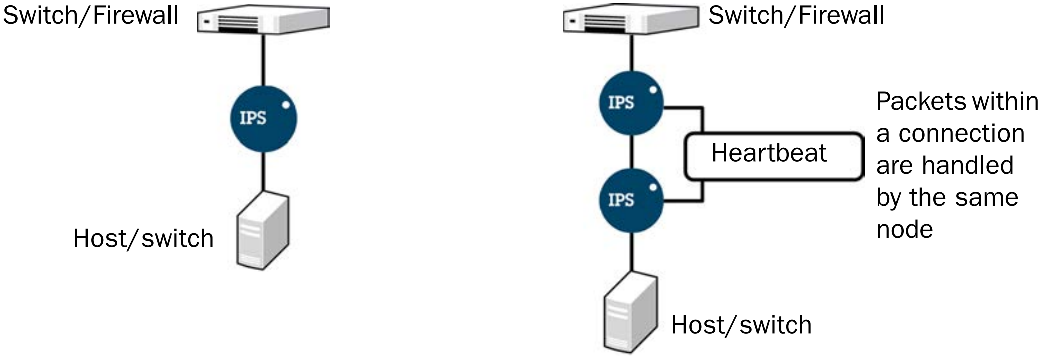
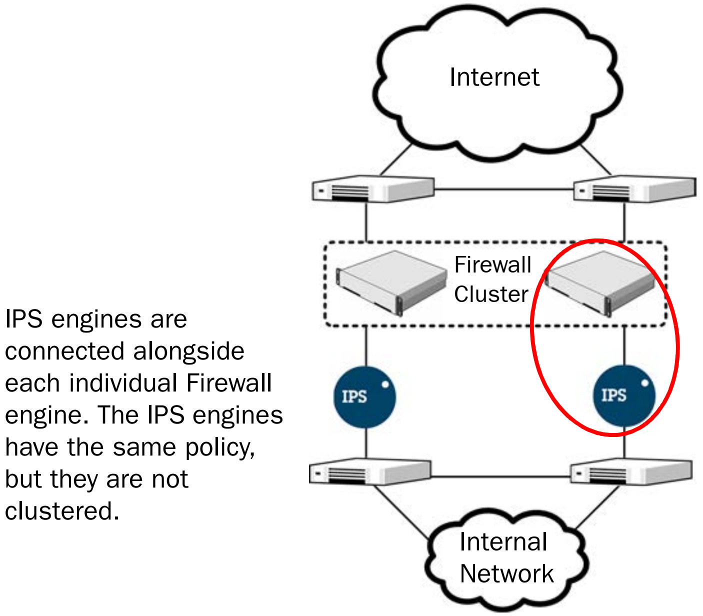


Illustration 4.14 Redundant Single Inline IPS Engines Alongside a Firewall Cluster


Internet
Firewall Cluster
IPS
IPS
Internal Network
IPS engines are connected alongside each individual Firewall engine. The IPS engines have the same policy, but they are not clustered.

In this deployment scenario, the Medium-Security Inspection Policy must be used on the IPS engines.

Network Configuration Scenarios for Layer 2 Firewalls

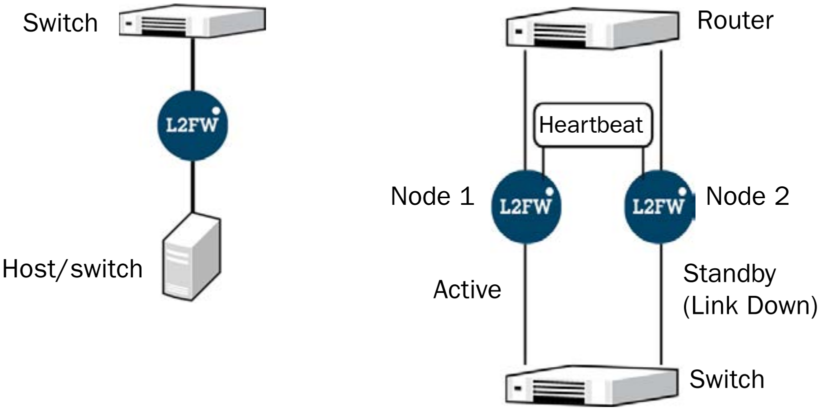
This section presents the most important scenarios for deploying a Single Layer 2 Firewall or a Layer 2 Firewall Cluster in different network configurations.

Deploying Layer 2 Firewalls in an IPS Configuration

In an IPS configuration, the Layer 2 Firewalls are installed inline directly in the traffic path.

Caution – Always use standard cabling methods with an inline Layer 2 Firewall. Use crossover cables to connect the appliance to hosts and straight cables to connect the appliance to switches. See Cabling Guidelines (page 47).

Illustration 4.15 Basic Inline Installations: Single Inline Layer 2 Firewall (Left) and Active/Standby Layer 2 Firewall Cluster (Right)



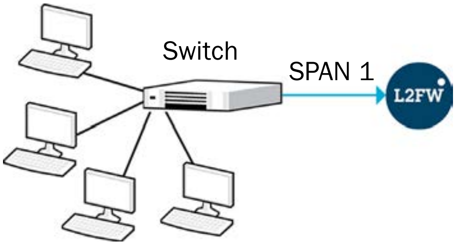
Deploying Layer 2 Firewalls in Passive Firewall Mode

Layer 2 Firewalls can be deployed in Passive Firewall mode in two ways:

- In capture mode to inspect packets that have been duplicated for inspection through SPAN or mirror ports.
- In passive inline mode by setting the engine to only log connections by default.

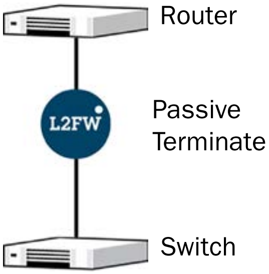
In a capture mode installation, packets are duplicated for inspection through a SPAN or mirror port on a switch/router. In a Layer 2 Firewall Cluster, each node must be connected to a SPAN or mirror port of its own.

Illustration 4.16 Passive Firewall: a Single Layer 2 Firewall in Capture Mode With SPAN/Mirror Ports



A Layer 2 Firewall can also be deployed in Passive Firewall mode in an inline configuration if the Only Log Connection mode is selected for the global Default Connection Termination in Access Rules setting in the Layer 2 Firewall engine properties.

Illustration 4.17 Passive Firewall: a Single Layer 2 Firewall in Passive Inline Mode



Cable Types

Follow standard cabling with inline IPS engines and Layer 2 Firewalls:

- Use straight cables to connect the Layer 2 Firewalls and IPS engines to switches.
- Use crossover cables to connect the Layer 2 Firewalls and IPS engines to hosts (such as routers or Firewalls).

Note – Fail-open network interface cards support Auto-MDIX, so both crossover and straight cables may work when the IPS engine is online. However, only the correct type of cable allows traffic to flow when the IPS engine is offline and the fail-open network interface card is in bypass state. It is recommended to test the IPS deployment in offline state to make sure that the correct cables are used.

Also, make sure the copper cables are correctly rated (CAT 5e or CAT 6 in gigabit networks).

Cabling for Master Engines that host Virtual IPS engines or Virtual Layer 2 Firewalls follows the same principles as the cabling for inline IPS engines and Layer 2 Firewalls.

Illustration 4.18 Correct Cable Types for Single IPS Engines

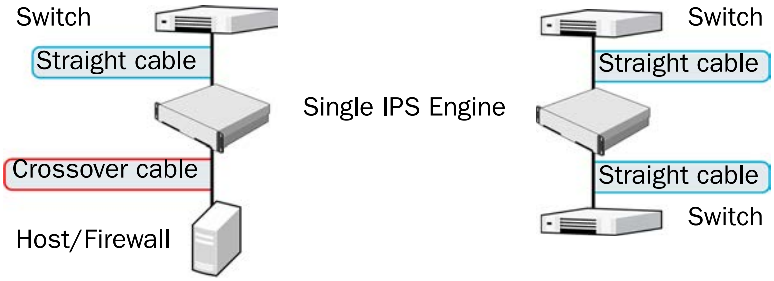


Illustration 4.19 Correct Cable Types for Single Layer 2 Firewalls

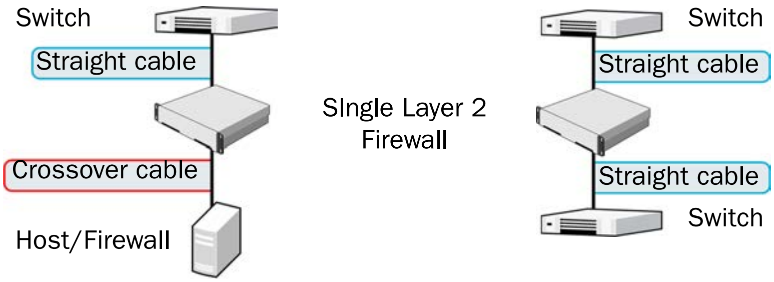


Illustration 4.20 Correct Cable Types for Serial IPS Clusters

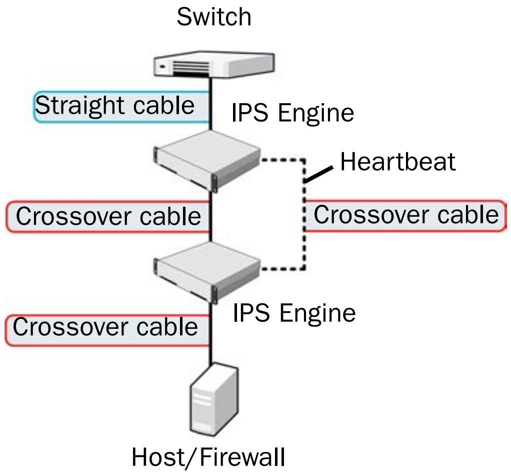


Illustration 4.21 Correct Cable Types for Active/Standby Layer 2 Firewall Clusters

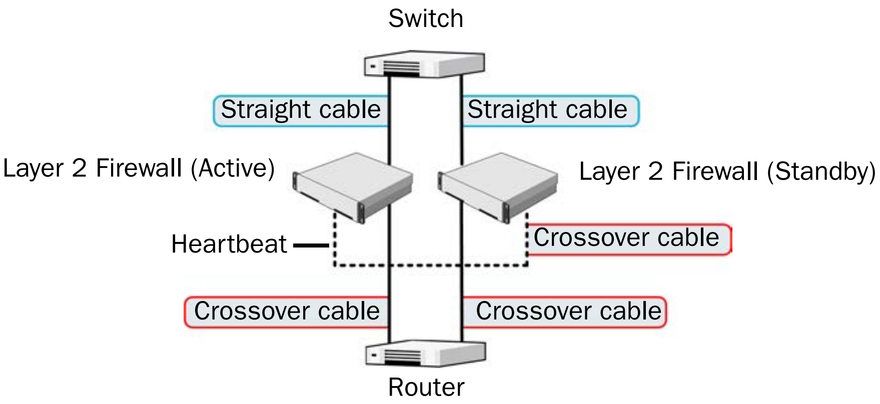


Illustration 4.22 Correct Cable Types for Serial Virtual IPS Clusters

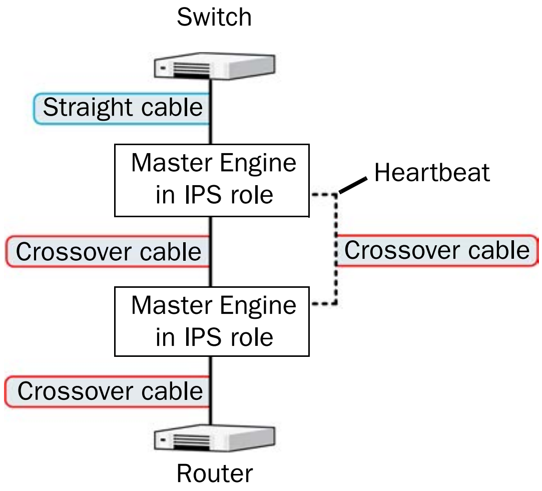
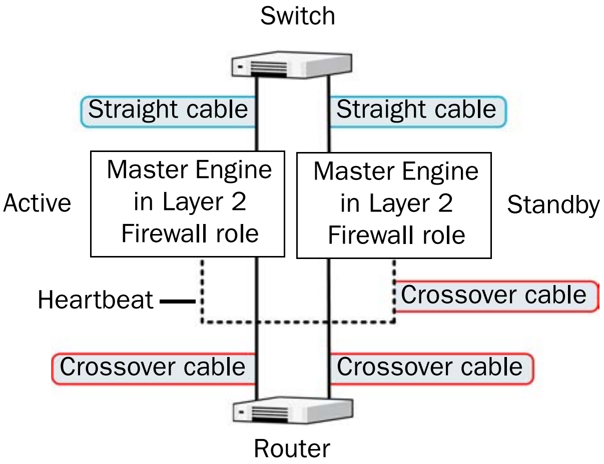


Illustration 4.23 Correct Cable Types for Active/Standby Virtual Layer 2 Firewall Clusters

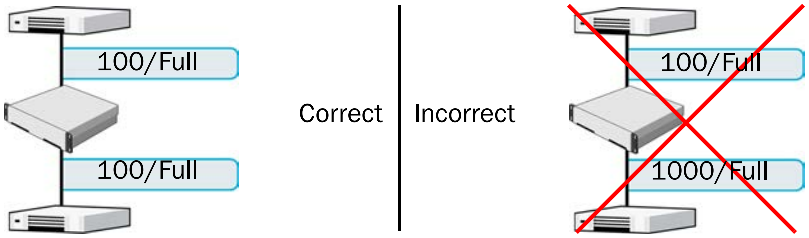


Speed and Duplex

Mismatched speed and duplex settings are a frequent source of networking problems. The basic principle for speed and duplex is that network cards at both ends of each cable must have identical settings. This principle also applies to the automatic negotiation setting: if one end of the cable is set to autonegotiate, the other end must also be set to autonegotiate and not to any fixed setting. Gigabit standards require interfaces to use autonegotiation. Fixed settings are not allowed at gigabit speeds.

For Inline interfaces, the settings must be identical on both links within each inline interface pair (identical settings on all four interfaces) instead of just matching settings at both ends of each cable (two + two interfaces). If one of the links has a lower maximum speed than the other link, the higher-speed link must be set to use the lower speed.

Illustration 4.24 Speed/Duplex Settings



SETTING UP SECURITY ENGINES

CHAPTER 5

IPS ENGINE CONFIGURATION

An IPS Cluster is a group of IPS nodes that work as a single logical entity to share the load of traffic processing. A Single IPS is an IPS engine that consists of a single node.

The following sections are included:

Overview of IPS Configuration

This chapter concentrates on configuration of network interfaces and clustering of IPS engines. The section Using IPS Engines (page 59) addresses other configuration tasks that you may define in the IPS elements' properties.

The IPS engine software can run on a McAfee NGFW appliance, on a standard server with an Intel-compatible processor, or on a virtualization platform.

IPS engines and Layer 2 Firewalls examine traffic in a very similar way. The main difference is that a Layer 2 Firewall drops packets that have not specifically been allowed to pass according to the engine's policy. An IPS engine, in contrast, by default allows packets to pass if the engine's policy does not specify a more specific action. For more information on Layer 2 Firewall engines, see Layer 2 Firewall Configuration (page 65).

Configuration of IPS Engines

IPS engines are configured and managed centrally through the Management Server. The Single IPS and IPS Cluster elements represent the IPS engine configuration on the Management Server.

Heartbeat Network for IPS Clusters

The nodes in an IPS cluster exchange status information through a *heartbeat network* using multicast transmissions. If an IPS node becomes unavailable, the other nodes of the cluster immediately notice this, and connections are reallocated to the available nodes. A dedicated network is recommended for at least the primary heartbeat communications.

Network Interfaces

The network interfaces of an IPS engine are identified by *Interface IDs*. During the configuration in the Management Client, you define the network interfaces of the IPS engine. During the engine configuration on the command line, the Interface IDs are mapped to the engine's physical interfaces. You must specify a unique Interface ID for each physical network interface.

Depending on whether you are configuring a Single IPS or an IPS Cluster, you can configure the following types of interfaces for each Interface ID in use:

Table 5.1 IPS Engine Interface Types

Interface Type	Description
Capture Interface	Capture Interfaces are used for listening to traffic that does not flow through the IPS engine. They are dedicated for capturing network traffic, and cannot be used for other purposes.
Inline Interface	Inline Interfaces handle traffic that flows through an IPS engine that is in inline mode. These Inline Interfaces cannot be simultaneously used for other purposes.
Normal Interface	Normal Interfaces are used for Management communications, sending log event information to the Log Server, and sending TCP Reset responses, as well as all communications in node-initiated connections. On IPS Clusters, Normal Interfaces are also used for the heartbeat between the nodes.

Configuration Workflow

The following sections provide an overview of the configuration tasks. Detailed step-by-step instructions can be found in the Management Client *Online Help* and the *McAfee SMC Administrator's Guide*.

Task 1: Create a Single IPS or IPS Cluster Element

To add a new Single IPS or IPS Cluster to the SMC, you must create an element that stores the configuration information related to the engines.

Task 2: Create Physical Interfaces

Physical interfaces represent the actual network interfaces on the engines. In an IPS Cluster, each physical interface definition represents a network interface on all nodes of the cluster.

The network interface numbering in the configuration is independent from the numbering of the physical interfaces. By default, the two numbering schemes are mapped in sequential order, but you can change the mapping freely using command line tools on the engine. This mapping can be done differently from node to node as long as you take care that the same interface on each node is correctly cabled to the same network.

Task 3: Define VLAN Interfaces

A *Virtual Local Area Network* (VLAN) is a logical grouping of hosts and network devices that appear as a single network segment regardless of the physical topology. IPS engines support *VLAN tagging* as defined in the IEEE 802.1q standard. One physical interface can support up to 4094 VLANs.

VLAN tagging can be used:

- to inspect VLAN tagged traffic (no VLAN Interface configuration required on the IPS engine)
- to define different inspection rules for different VLANs (requires defining VLAN Interfaces for the IPS engine)
- for the IPS engine's own management and event logging communications when the control IP address is on a VLAN Interface.

By default, all captured VLAN traffic is inspected in the same way as non-VLAN traffic. You only need to configure VLAN Interfaces for the IPS engine's Capture interfaces if you want to customize traffic inspection for the different VLANs. The traffic inspection is customized for the VLANs by defining different Logical Interfaces for the different VLAN Capture Interfaces. The Logical Interface elements are then used in the IPS Policy rules to define which rules are used for which VLANs.

When you use VLAN Inline Interfaces, the interface numbers must be different and the VLAN identifier must be identical in both of the Inline Interfaces. For example, 3.101 and 4.101 would be a valid pair of VLAN Inline Interfaces. Also, when a VLAN Interface is used for an Inline Interface, it cannot be simultaneously used for any other types of interfaces.

When you use VLAN with Capture Interfaces, the network interface used as the R*eset Interface* for sending TCP Reset responses must be defined in the Capture Interface's properties. The reset is automatically tagged for the same VLAN that triggers a reset. The Reset Interface must be connected to the same VLAN/Broadcast domain as the Capture Interface to reach the communicating hosts.

Task 4: Define Normal Interfaces

In a Single IPS, Normal Interfaces are used for communication between the IPS engine and the Management Server, for sending event information and traffic recordings to Log Servers, and as the Reset Interface for sending TCP Reset responses.

In an IPS Cluster, Normal Interfaces handle all traffic for which the end-point of the communication is a node itself. Normal Interfaces are used for the Heartbeat communication between the nodes, for communication between each individual node and the Management Server, for sending event information and traffic recordings to Log Servers, and for any other traffic between the node itself and some other host. Normal Interfaces in an IPS Cluster are also used as the Reset Interface for sending TCP Reset responses.

Each Single IPS engine or node in an IPS Cluster must have at least one Normal Interface defined. Multiple Normal Interfaces can be configured for the same physical network interface. It is recommended to create a separate Normal Interface that is used for communication with the Management Server rather than using the same Normal Interface for sending event information and traffic recordings to Log Servers, and for communication with the Management Server.

You can optionally assign an IP address to a Normal Interface. When a Normal Interface is used for communication with the Management Server, as the Heartbeat Interface in an IPS Cluster, or for communication with the Log Server, an IP Address is needed. When the same Normal

Interface that is used for communication with the Management Server and Log Server is also used as a Reset Interface for sending TCP Reset responses, it can have an IP address. When a Normal Interface is used *only* as a Reset Interface, it must not have an IP address.

All nodes in an IPS Cluster must have the same netmask value for the IP address of their respective Normal Interfaces. The IP addresses specified for each node are used whenever the nodes need to be contacted individually.

Task 5: Define Logical Interfaces

A Logical Interface is used in the Capture Interface and Inline Interface configuration to represent one or more network interfaces. A Logical Interface can represent any number or combination of physical interfaces or VLAN Interfaces, except that the same Logical Interface cannot be used to represent both Capture Interfaces and Inline Interfaces on the same IPS engine.

Logical Interfaces have one option, View Interface as One LAN. Selecting this option prevents the IPS engine from seeing a single connection as multiple connections when a switch passes traffic between different VLANs.

Logical Interfaces can also be used in IPS Policies to create rules that match based on which interface the traffic was picked up from. For example, you can create a different Logical Interface for each VLAN and use them to create rules that apply only to traffic from a specific VLAN.

A Logical Interface element called **System Communications** is automatically assigned to interfaces that have an IP address that is used as the primary or backup Control IP address. You can use the **System Communications** Logical Interface to represent all of the Control IP addresses in IPS Policies.

Task 6: Define Capture Interfaces

You must define Capture Interfaces if you want to use the IPS engine to inspect traffic that does not flow through the IPS engine.

Capture Interfaces have definitions for the corresponding *Logical Interface* that the interface belongs to. The Logical Interface represents one or more network interfaces that capture the traffic for inspection:

- When a Capture Interface is connected to a switch SPAN port, each Capture Interface is bound to one Logical Interface. More than one Capture Interface can optionally be bound to the same Logical Interface.
- When a network TAP device is used, two Capture Interfaces are bound to the same Logical Interface. The monitored traffic going to different directions is captured through these two related network interfaces and is then combined into a complete traffic flow on the Logical Interface.

You cannot select the same Logical Interface for a Capture Interface and an Inline Interface on the same IPS engine.

A Reset Interface can be selected for a Capture Interface to send TCP Reset responses for the traffic captured from the interface. The Reset Interface is a Normal Interface that can reach the communicating components with the TCP Reset (for example, a Normal Interface connected to the monitored network).

Task 7: Define Inline Interfaces

You must define Inline Interfaces if you want to place a Single IPS or Inline IPS Cluster directly in the traffic path so that any traffic that is to be inspected goes through the IPS engine. An Inline Interface is configured with two Interface IDs, representing two physical interfaces or two VLANs. Some NGFW appliances use a fail-open network card, so the Inline Interfaces must be configured for those specific ports. Inline Interfaces do not have an IP address or a MAC address visible to the network.

In addition to the Interface IDs, Inline Interfaces also have definitions for the corresponding Logical Interface that the interface belongs to. A single Logical Interface can represent one or more pairs of Inline Interfaces. The Logical Interface element can be used to represent the interfaces in the IPS Policy. You cannot select the same Logical Interface for a Capture Interface and an Inline Interface on the same IPS engine.

Task 8: Select Interface Options

The Interface Options allow you to define which IP addresses are used as the primary and backup Control IP address, which interfaces are used as the primary and backup Heartbeat Interface (*IPS Clusters only*), and the default IP address for outgoing traffic. By default, the first IP address you add to a Normal Interface is automatically selected as the primary Control IP address, the primary Heartbeat Interface (*IPS Clusters only*), and the default IP address for outgoing traffic. You can optionally change which physical interface is used for each of these purposes, and define a backup Control IP address and backup Heartbeat Interface (*IPS Clusters only*).

Task 9: Install the IPS Engines

During the engine installation, you map the physical interfaces to the interface IDs you define in the Management Client.

You can also install the engine automatically using a configuration saved on a USB memory stick. If you use the automatic engine configuration, interface IDs are automatically mapped to physical interfaces in sequential order. For example, Interface ID 0 is mapped to eth0, Interface ID 1 is mapped to eth1, and so on. The first physical interface (eth0) is always used as the Management interface. For this reason, Interface ID 0 must be defined as the Management interface in the Management Client when automatic engine configuration is used.

You also activate a basic IPS Policy (the initial configuration) that allows you to establish contact between the Management Server and the engine. After contacting the Management Server, the engine receives a certificate from the SMC for identification, and a trust relationship between the engine and the Management Server is established.

Task 10: Install an IPS Policy

After the IPS engine makes initial contact with the Management Server, only the interface used for the control connection with the Management Server is configured. You must install an IPS Policy using the Management Client to transfer the complete interface configuration to the IPS engine.

Using IPS Engines

The main points of IPS engine configuration are explained in the preceding sections of this chapter, but this section illustrates some additional concepts that are useful when working with IPS engines.

Contact Addresses for NATed Communications

In a situation where a device performs network address translation (NAT) between the communicating SMC components, you must specify contact addresses for the components. The *contact address* is the NATed address of the component that is contacted instead of the component's real IP address.

The contact addresses for the SMC components on each Site behind NAT are grouped into a *Location* element. The contact address for each component is defined in the element's properties based on the Location of the contacting component.

For example, when a Management Server contacts a IPS engine node through NAT, the Management Server uses the NATed contact address, not the IPS engine's real IP address. The NAT device in between performs the address translation from the NATed address to the IPS engine's real IP address as usual.

You create the Locations and add elements to the Locations based on how your network is set up. Then you define the Contact Addresses for each element for each Location in the properties of the elements. All SMC components in other Locations then use the addresses defined for their Location for contact.

Cluster Load Balancing

In an IPS Cluster, the recommended way to cluster the nodes is load-balanced clustering, where traffic is balanced between the nodes dynamically. Load-balanced clustering provides both fault tolerance and performance benefits.

When load-balanced clustering is used, the traffic arriving at the IPS Cluster is balanced across the nodes by means of a load-balancing filter. This filtering process distributes packets between the IPS Cluster nodes and keeps track of packet distribution. The IPS Cluster determines the packet ownership of the nodes by comparing the incoming packet with node-specific values based on the packet headers.

The IPS Cluster keeps track of which node is handling each ongoing connection. As a result, all packets that are part of a given connection can be handled by the same node. Some protocols use multiple connections, which are sometimes handled by different nodes, but this usually does not affect the processing of the traffic.

TCP Modes for Deep Inspection

IPS engines can handle TCP connections in two different modes: *Normal* mode and *Strict* mode.

Table 5.2 TCP Modes for Deep Inspection

Mode	Description
Normal	This is the default mode for handling TCP connections. It is the only available inspection mode when the IPS is used as an intrusion detection system (IDS). See NGFW Deployment in IPS and Layer 2 Firewall Roles (page 29) for more information. In Normal TCP mode, the IPS engine checks that the traffic proceeds according to the TCP protocol specification. The IPS engine does not need to see all the packets in a TCP connection (for example, the packets that initiate a new TCP connection). The IPS engine does not modify the packets and it does not enforce the packet direction. This means, for example, that SYN and SYN-ACK packets are allowed from the same interface.
Strict	Strict TCP mode provides enhanced protection against TCP evasion attempts. It is only available if the IPS engine is in inline mode. See NGFW Deployment in IPS and Layer 2 Firewall Roles (page 29) for more information. Strict TCP mode is used by default for handling TCP connections with the TLS Inspection and URL filtering features. Optionally, you can also enable Strict TCP mode for all TCP traffic in the Single IPS or IPS Cluster element properties. In Strict TCP mode, the IPS engine controls the progress of a TCP connection and checks that the TCP handshake proceeds according to the TCP protocol specification. The same Single IPS engine or IPS Cluster node must be able to see all the packets in the connection. The IPS engine also enforces the order of the packets and the packet direction (for example, SYN and SYN-ACK packets are not allowed from the same interface). The IPS engine also modifies the TCP packets' header, particularly the TCP window size. It can also remove TCP options that are not supported in Strict TCP mode (for example, timestamps) from the packets. In TLS Inspection, the IPS engine also modifies the TCP packet data (decrypts the packets for inspection and then encrypts the packets again). If an IPS engine cannot inspect the whole TCP connection (for example, the IPS engine goes offline), the connection is dropped if TLS inspection is applied to the connection. If URL filtering is used and the inspection of the connection starts in the middle of the connection (for example, the IPS engine goes from offline to online state), or the connection is transferred to another IPS engine in a high-availability environment, the inspection is done as in Normal TCP mode. When TLS inspection is first activated, the currently open TCP connections are inspected according to Normal TCP mode and Strict TCP mode is only applied to the new TCP connections. The same applies for all TCP connections if Strict TCP mode has been manually enabled in the Single IPS or IPS Cluster element’s properties.

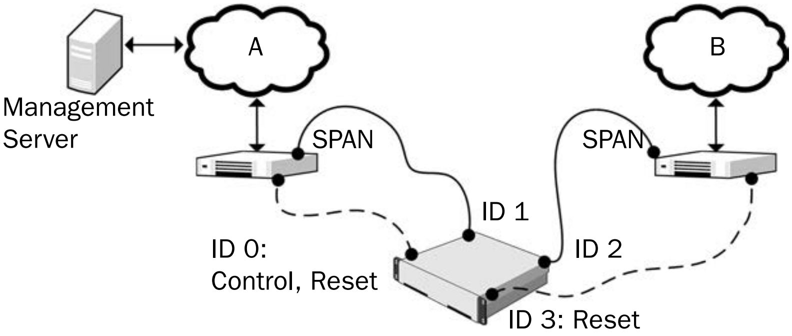
Examples of IPS Engine Configuration

The examples in this section illustrate some common uses for IPS engines and general steps on how each scenario is configured.

Configuring Capture Interfaces with SPAN

The administrator at company A wants to set up a Single IPS engine and deploy it in IDS configuration using SPAN ports on the switches to duplicate packets for inspection. Illustration 5.1 shows the interfaces of the IPS engine in IDS configuration.

Illustration 5.1 Capture Interfaces with SPAN



In this example, Interface ID 0 is a Normal Interface used for management connections, and sending TCP Reset responses for network segment A. Interface ID 1 is a Capture Interface for capturing network traffic from the network segment A switch for inspection. Interface ID 2 is a Capture Interface for capturing network traffic from the network segment B switch for inspection. Interface ID 3 is a Normal Interface used for sending TCP Reset responses for network segment B.

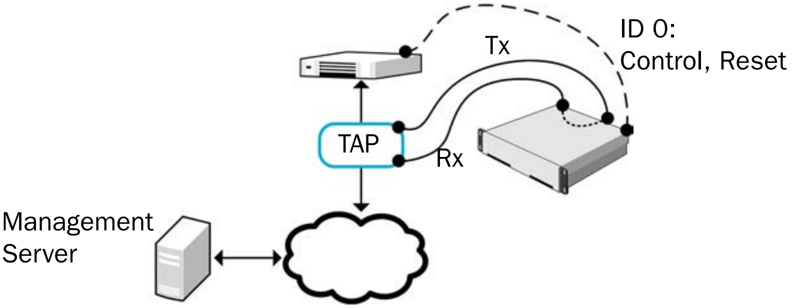
The administrator does the following:

1. Creates a Single IPS element and selects the Log Server to which it sends log data and the traffic recordings.
2. Defines Interface ID 0 as a Normal Interface and adds an IP address to it.
   - The IP address on Interface ID 0 is automatically selected as the Primary Control IP address because Interface ID 0 is the first Normal Interface with an IP address.
3. Defines Interface ID 3 as a Normal Interface without an IP address.
   - Because Interface ID 3 is used only as a Reset Interface, it must not have an IP address.
4. Defines Interface ID 1 as a Capture Interface and selects Interface ID 0 as the Reset Interface.
5. Defines Interface ID 2 as a Capture Interface and selects Interface ID 3 as the Reset Interface.
6. Saves the initial configuration of the engine in the Management Client.
7. Maps the interface IDs to the physical interfaces in the Engine Configuration Wizard on the engine's command line and makes initial contact with the Management Server.
8. Installs an IPS Policy in the Management Client to transfer the configuration to the engine.

Configuring Capture Interfaces with TAP

The administrator at company B wants to set up a Single IPS engine and deploy it in IDS configuration using a network TAP device. WireTAP copies transmitted (Tx) and received (Rx) packets from the monitored cable and forwards them to separate links for further analysis in the Single IPS engine. Illustration 5.2 shows the interfaces of the Single IPS engine in IDS configuration.

Illustration 5.2 Capture Interfaces with TAP



In this example, Interface ID 0 is a Normal Interface used for management connections, and sending TCP Reset responses. Interface ID 1 is a Capture Interface that listens to the received (Rx) packets from the network TAP. Interface ID 2 is a Capture Interface that listens to transmitted (Tx) packets from the network TAP. Interface IDs 1 and 2 share the same Logical Interface, which combines the traffic from both physical interfaces so that it can be inspected as a complete traffic flow.

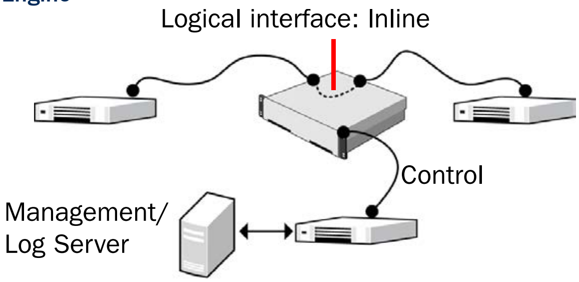
The administrator does the following:

1. Creates a Single IPS element and selects the Log Server to which it sends log data and traffic recordings.
2. Creates a Logical Interface called Capture for the two Capture Interfaces.
3. Defines Interface ID 0 as a Normal Interface and adds an IP address to it.
4. Defines Interface ID 1 and Interface ID 2 as Capture Interfaces, selects Interface ID 0 as the Reset Interface, and selects the Logical Interface called Capture for both.
5. Saves the initial configuration of the engine in the Management Client.
6. Connects the network cables to the appropriate NICs.
7. Maps the interface IDs to the physical interfaces in the Engine Configuration Wizard on the engine’s command line and makes initial contact with the Management Server.
8. Installs an IPS Policy in the Management Client to transfer the configuration to the engine.

Configuring Inline Interfaces

The administrator at Company C wants to set up a Single IPS engine and deploy it in the traffic path. Illustration 5.3 shows the interfaces of the inline Single IPS engine.

Illustration 5.3 Inline IPS Engine



In this example, the IP address on Interface ID 0 is configured as the Control IP address for management connections. Interface ID 1 and Interface ID 2 are an Inline Interface pair that share the same Logical Interface, called Inline. Traffic comes in through Interface ID 1. Any traffic that is allowed by the IPS engine leaves through Interface ID 2.

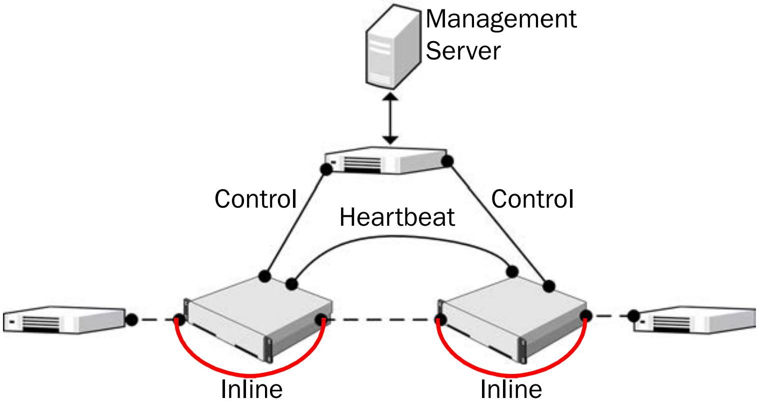
The administrator does the following:

1. Creates a Single IPS element and selects the Log Server to which it sends log data and traffic recordings.
2. Creates a Logical Interface called Inline for the Inline Interface pair.
3. Defines Interface ID 0 as a normal interface and adds an IP address to it.
4. Defines Interface IDs 1 and 2 as an Inline Interface pair and selects the Logical Interface called Inline for the pair.
5. Saves the initial configuration of the engine in the Management Client.
6. Connects the network cables to the appropriate NICs.
7. Maps the interface IDs to the physical interfaces in the Engine Configuration Wizard on the engine’s command line and makes initial contact with the Management Server.
8. Installs an IPS Policy in the Management Client to transfer the configuration to the engine.

Setting up an Inline Serial IPS Cluster

The administrators at Company D want to set up an IPS Cluster in a serial inline deployment. Illustration 5.4 shows the interfaces of the inline IPS Cluster.

Illustration 5.4 Inline Serial IPS Cluster



In this example, the IPS Cluster consists of two nodes. Interface ID 0 is a Normal Interface used for the heartbeat communication between the nodes. Interface ID 1 is a Normal Interface used for communication with the Management Server. Interface ID 2 and Interface ID 3 are an Inline Interface pair that share one Logical Interface, called Inline. Traffic enters each IPS node through Interface ID 2 and leaves through Interface ID 3.

The administrators:

1. Create an IPS Cluster element and select the Log Server to which the IPS Cluster sends event data and traffic recordings.
2. Define Interface ID 0 as a Normal Interface and add IP addresses for each of the nodes. The IP address on Interface ID 0 is automatically selected as the Primary Control IP address, the Primary Heartbeat Interface, and the Log communication source IP Address.
3. Define Interface ID 1 as a Normal Interface and add IP addresses for each of the nodes.
4. Define Interface IDs 2 and 3 as an Inline Interface pair and select the Logical Interface called Inline for the pair.
5. Select Interface ID 0 as the Primary Heartbeat Interface and select the IP address on Interface ID 1 as the Primary Control IP address in the Interface Options.
6. Save the initial configuration of the engine in the Management Client.
7. Connect the Heartbeat and Inline Interfaces between the nodes with crossover cables, and the rest of the interfaces with straight cables.
8. Map the interface IDs to the physical interfaces in the Engine Configuration Wizard on the engine's command line and makes initial contact with the Management Server.
9. Install an IPS Policy on each of the nodes in the Management Client to transfer the configuration to the IPS Cluster engine.

CHAPTER 6

LAYER 2 FIREWALL CONFIGURATION

A Layer 2 Firewall is a basic firewall engine that provides access control and deep inspection of traffic.

The following sections are included:

Overview of Layer 2 Firewall Configuration

This chapter concentrates on the settings related to configuring network interfaces and clustering of Layer 2 Firewalls. The section Using Layer 2 Firewalls (page 71) addresses other configuration tasks that you may do in the Layer 2 Firewall elements' properties.

Layer 2 Firewalls are basic firewalls with a limited set of functionalities. They provide access control and deep inspection of traffic. More advanced firewall features such as VPNs, authentication, anti-spam, and anti-virus are not supported on Layer 2 Firewalls.

You can cluster Layer 2 Firewalls. Only one Layer 2 Firewall node is active at a time in a Layer 2 Firewall Cluster. If the active node goes offline, one of the standby nodes automatically becomes active and starts processing traffic.

The Layer 2 Firewall software can run on a McAfee NGFW appliance, on a standard server with an Intel-compatible processor, or on a virtualization platform.

Layer 2 Firewalls and IPS engines examine traffic in a very similar way. The main difference is that a Layer 2 Firewalls drops packets that have not specifically been allowed to pass according to the engine's policy. An IPS engine, in contrast, by default allows packets to pass if the engine's policy does not set a more specific action. For more information on IPS engines, see IPS Engine Configuration (page 53).

Configuration of Layer 2 Firewalls

Layer 2 Firewalls are configured and managed centrally through the Management Server. The Single Layer 2 Firewall and the Layer 2 Firewall Cluster elements represent the engine's configuration on the Management Server.

Network Interfaces

The network interfaces of a Layer 2 Firewall are identified by *Interface IDs*. During the configuration in the Management Client, you define the network interfaces of the Layer 2 Firewall. During the engine configuration on the command line, the Interface IDs are mapped to the engine's physical interfaces. You must specify a unique Interface ID for each physical network interface.

You can configure the following types of physical interfaces for each Interface ID in use:

Table 6.1 Layer 2 Firewall Interface Types

Interface Type	Description
Capture Interface	Capture Interfaces are used for listening to traffic that does not flow through the Layer 2 Firewall. They can only be used for capturing network traffic and cannot be used for other purposes.
Inline Interface	Inline Interfaces handle traffic that flows through a Layer 2 Firewall that is in inline mode. They cannot be used for other purposes.

Table 6.1 Layer 2 Firewall Interface Types (Continued)

Interface Type	Description
Normal Interface	Normal Interfaces are used for Management communications, sending log event information to the Log Server, and sending TCP Reset responses, as well as for all communications in node-initiated connections. On Layer 2 Firewall Clusters, Normal Interfaces are also used for the heartbeat between the nodes.

Configuration Workflow

The following sections provide an overview of the configuration tasks. Detailed step-by-step instructions can be found in the Management Client *Online Help* and the *McAfee SMC Administrator's Guide*.

Task 1: Create a Single Layer 2 Firewall or Layer 2 Firewall Cluster Element

To add a new Single Layer 2 Firewall or Layer 2 Firewall Cluster to the McAfee Security Management Center, you must create an element that stores the configuration information related to the engines.

Task 2: Create Physical Interfaces

Physical interfaces represent the actual network interfaces on the engines. In a Layer 2 Firewall Cluster, each physical interface definition represents a network interface on all nodes of the cluster.

The network interface numbering in the configuration is independent from the numbering of the physical interfaces. By default, the two numbering schemes are mapped in sequential order, but you can change the mapping freely using command line tools on the engine. This mapping can be done differently from node to node as long as you take care that the same interface on each node is correctly cabled to the same network.

Task 3: Define VLAN Interfaces

A *Virtual Local Area Network* (VLAN) is a logical grouping of hosts and network devices that appear as a single network segment regardless of the physical topology. Layer 2 Firewalls support *VLAN tagging* as defined in the IEEE 802.1q standard. One physical interface can support up to 4094 VLANs.

VLAN tagging can be used:

- to inspect VLAN tagged traffic (no VLAN interface configuration required on the Layer 2 Firewall)
- to define different inspection rules for different VLANs (requires defining VLAN interfaces for the Layer 2 Firewalls)
- for the Layer 2 Firewall's own management and event logging communications when the control network interface is connected to a VLAN trunk.

By default, all VLAN traffic is inspected in the same way as non-VLAN traffic. You only need to configure VLAN Interfaces for the physical interfaces if you want to customize traffic inspection for the different VLANs. The traffic inspection is customized for the VLANs by defining different Logical Interfaces for the different VLAN Interfaces. The Logical Interface elements are then used in the Layer 2 Firewall Policy rules to define which rules are used for which VLANs.

When you use VLANs with Inline Interfaces, the interface numbers must be different and the VLAN identifier must be identical in both of the Inline Interfaces. For example, 3.101 and 4.101 would be a valid pair of VLAN Inline Interfaces. Also, when a VLAN Interface is used for an Inline Interface, it cannot be simultaneously used for any other types of interfaces.

Task 4: Define Normal Interfaces

In a Single Layer 2 Firewall, Normal Interfaces are used for communication between the Layer 2 Firewall and the Management Server, and for sending event information and traffic recordings to Log Servers.

In a Layer 2 Firewall Cluster, Normal Interfaces handle all traffic for which the end-point of the communication is a node itself. Normal Interfaces are used for the Heartbeat communication between the nodes, for communication between each individual node and the Management Server, for sending event information and traffic recordings to Log Servers, and for any other traffic between the node itself and some other host.

Each Single Layer 2 Firewall or node Layer 2 Firewall Cluster must have at least one Normal Interface defined. Multiple Normal Interfaces can be configured for the same physical network interface. It is recommended to create a separate Normal Interface that is used for communication with the Management Server rather than using the same Normal Interface for sending event information and traffic recordings to Log Servers and for communication with the Management Server.

You can optionally assign an IP address to a Normal Interface. When a Normal Interface is used for communication with the Management Server, as the Heartbeat Interface in a Layer 2 Firewall Cluster, or for communication with the Log Server, an IP Address is required.

All nodes in a Layer 2 Firewall Cluster must have the same netmask value for the IP address of their respective Normal Interfaces. The IP addresses specified for each node are used whenever the nodes need to be contacted individually.

Task 5: Define Logical Interfaces

A Logical Interface is used in the Inline Interface and Capture Interface configuration to represent one or more network interfaces. A Logical Interface can represent any number or combination of physical interfaces or VLAN Interfaces, except that the same Logical Interface cannot be used to represent both Capture Interfaces and Inline Interfaces on the same Layer 2 Firewall.

Logical Interfaces have one option, View Interface as One LAN. Selecting this option prevents the Layer 2 Firewall from seeing a single connection as multiple connections when a switch passes traffic between different VLANs.

Logical Interfaces can also be used in Layer 2 Firewall Policies to create rules that match based on which interface the traffic was picked up from. For example, you can create a different Logical Interface for each VLAN and use them to create rules that apply only to traffic from a specific VLAN.

A Logical Interface element called **System Communications** is automatically assigned to interfaces that have an IP address that is used as the primary or backup Control IP address. You can use the **System Communications** Logical Interface to represent all of the Control IP addresses in Layer 2 Firewall Policies.

Task 6: Define Inline Interfaces

You must define Inline Interfaces if you want to place a Single Layer 2 Firewall or Layer 2 Firewall Cluster directly in the traffic path so that any traffic to be inspected goes through the Layer 2 Firewall. An Inline Interface is configured with two Interface IDs that represent two physical interfaces or two VLANs. Some McAfee NGFW appliances have a fail-open network card, so the Inline Interfaces must be configured for those specific ports. Inline Interfaces do not have an IP address or a MAC address visible to the network.

In addition to the Interface IDs, Inline Interfaces also have definitions for the corresponding Logical Interface that this interface belongs to. A single Logical Interface can represent one or more pairs of Inline Interfaces. The Logical Interface element can be used to represent the interfaces in the Layer 2 Firewall Policy. You cannot select the same Logical Interface for a Capture Interface and an Inline Interface on the same Layer 2 Firewall.

Task 7: Define Capture Interfaces

Capture Interfaces are used to configure a Layer 2 Firewall in Passive Firewall mode. In Passive Firewall mode, the Layer 2 Firewall only listens to traffic that does not flow through it.

Capture Interfaces have definitions for the corresponding Logical Interface that the interface belongs to. The Logical Interface represents one or more network interfaces that capture the traffic for inspection. Each Capture Interface is bound to one Logical Interface. More than one Capture Interface can optionally be bound to the same Logical Interface.

You cannot select the same Logical Interface for a Capture Interface and an Inline Interface on the same Layer 2 Firewall.

A Reset Interface can be selected for a Capture Interface to send TCP Reset responses for the traffic captured from the interface. The Reset Interface is a Normal Interface that can reach the communicating components with a TCP Reset (for example, a Normal Interface connected to the monitored network).

Task 8: Select Interface Options

The Interface Options allow you to define which IP addresses are used as the primary and backup Control IP address, which interfaces are used as the primary and backup Heartbeat Interface (*Layer 2 Firewall Clusters only*), and the default IP address for outgoing traffic. By default, the first IP address you add to a Normal Interface is automatically selected as the primary Control IP address, the primary Heartbeat Interface (*Layer 2 Firewall Clusters only*), and the default IP address for outgoing traffic. You can optionally change which physical interface is used for each of these purposes, and define a backup Control IP address and backup Heartbeat Interface (*Layer 2 Firewall Clusters only*).

Task 9: Install the Layer 2 Firewall Engines

During the engine installation, you map the physical interfaces to the interface IDs you define in the Management Client.

You can also install the engine automatically using a configuration saved on a USB memory stick. If you use the automatic engine configuration, interface IDs are automatically mapped to physical interfaces in sequential order. For example, Interface ID 0 is mapped to eth0, Interface ID 1 is mapped to eth1, and so on. The first physical interface (eth0) is always used as the Management interface. For this reason, Interface ID 0 must be defined as the Management interface in the Management Client when automatic engine configuration is used.

When you install the engine, you also activate a basic Layer 2 Firewall Policy (the initial configuration) that allows you to establish contact between the Management Server and the engine. After contacting the Management Server, the engine receives a certificate from the SMC for identification, and a trust relationship between the engine and the Management Server is established.

Task 10: Install a Layer 2 Firewall Policy

After the Layer 2 Firewall engine makes initial contact with the Management Server, only the interface used for the control connection with the Management Server is configured. You must install a Layer 2 Firewall Policy using the Management Client to transfer the complete interface configuration to the Layer 2 Firewall.

Using Layer 2 Firewalls

The main points of Layer 2 Firewall configuration are explained in the preceding sections of this chapter. This section illustrates some additional concepts that are useful when working with Layer 2 Firewalls.

Contact Addresses for NATed Communications

In a situation where a device performs network address translation (NAT) between the communicating SMC components, you must specify contact addresses for the components. The *contact address* is the NATed address of the component that is contacted instead of the component's real IP address.

The contact addresses for the SMC components on each Site behind NAT are grouped into a *Location* element. The contact address for each component is defined in the element's properties based on the Location of the contacting component.

For example, when a Management Server contacts a Layer 2 Firewall through NAT, the Management Server uses the NATed contact address, not the Layer 2 Firewall's real IP address. The NAT device in between performs the address translation from the NATed address to the Layer 2 Firewall's real IP address.

You create the Locations and add elements to the Locations based on how your network is set up. Then you define the Contact Addresses for each element for each Location in the properties of the elements. All SMC components in other Locations then use the addresses defined for their Location for contact.

Passive Firewall Mode

Layer 2 Firewalls can be configured in Passive Firewall mode. In Passive Firewall mode, the engine captures network traffic for inspection but does not actively filter traffic.

The most common way to configure a Layer 2 Firewall in Passive Firewall mode is to define Capture Interfaces for listening to network traffic that does not flow through the Layer 2 Firewall. If you configure only Capture Interfaces, the engine always functions in Passive Firewall mode. You can also use a Layer 2 Firewall that has Inline Interfaces in Passive Firewall mode. To do this, you configure the engine to only log connections.

TCP Modes for Deep Inspection

Layer 2 Firewalls can handle TCP connections in two different modes: *Normal* mode and *Strict* mode.

Table 6.2 TCP Modes for Deep Inspection

Mode	Description
Normal	This is the default mode for handling TCP connections. In Normal TCP mode, the Layer 2 Firewall checks that the traffic proceeds according to the TCP protocol specification. The Layer 2 Firewall does not need to see all the packets in a TCP connection (for example, the packets that initiate a new TCP connection). The Layer 2 Firewall does not modify the packets and it does not enforce the packet direction. This means, for example, that SYN and SYN-ACK packets are allowed from the same interface.
Strict	Strict TCP inspection mode provides enhanced protection against TCP evasion attempts. Strict TCP mode is used by default for handling TCP connections with the TLS Inspection feature. Optionally, you can also enable Strict TCP mode for all TCP traffic in the Single Layer 2 Firewall or Layer 2 Firewall Cluster element properties. In Strict TCP mode, the Layer 2 Firewall controls the progress of a TCP connection and checks that the TCP handshake proceeds according to the TCP protocol specification. The same Single Layer 2 Firewall or node in a Layer 2 Firewall Cluster must be able to see all the packets in the connection. The Layer 2 Firewall engine also enforces the order of the packets and the packet direction (for example, SYN and SYN-ACK packets are not allowed from the same interface). The Layer 2 Firewall engine also modifies the TCP packets' header, particularly the TCP window size. It can also remove TCP options that are not supported in Strict TCP mode (for example, timestamps) from the packets. In TLS Inspection, the Layer 2 Firewall also modifies the TCP packet data (decrypts the packets for inspection and then encrypts the packets again). If a Layer 2 Firewall cannot inspect the whole TCP connection (for example, the Layer 2 Firewall goes offline), the connection is dropped if TLS inspection is applied to the connection. When TLS inspection is first activated, the currently open TCP connections are inspected according to Normal TCP mode and Strict TCP mode is only applied to the new TCP connections. The same applies for all TCP connections if Strict TCP mode has been manually enabled in the Single Layer 2 Firewall or Layer 2 Firewall Cluster element’s properties.

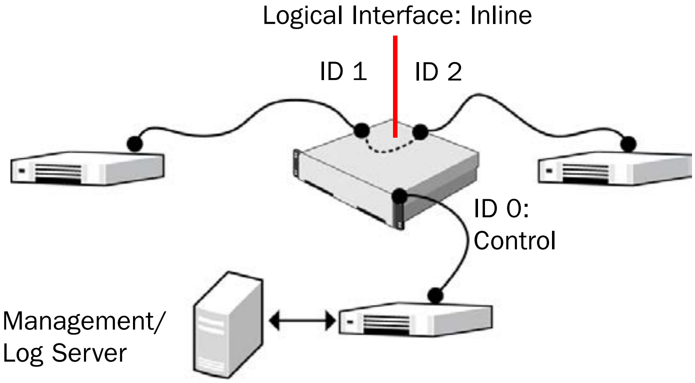
Examples of Layer 2 Firewalls

The examples in this section illustrate some common uses for Layer 2 Firewalls and general steps on how each scenario is configured.

Configuring Inline Interfaces in Inline Mode

The administrator at Company A wants to set up a Layer 2 Firewall and deploy it in the traffic path in inline mode. Illustration 6.1 shows the interfaces of the inline Layer 2 Firewall.

Illustration 6.1 Inline Layer 2 Firewall



In this example, the IP address on Interface ID 0 is configured as the Control IP address for management connections. Interface ID 1 and Interface ID 2 are an inline interface pair that share the same Logical Interface, called Inline. Traffic comes in through Interface ID 1. Any traffic that is allowed by the Layer 2 Firewall leaves through Interface ID 2.

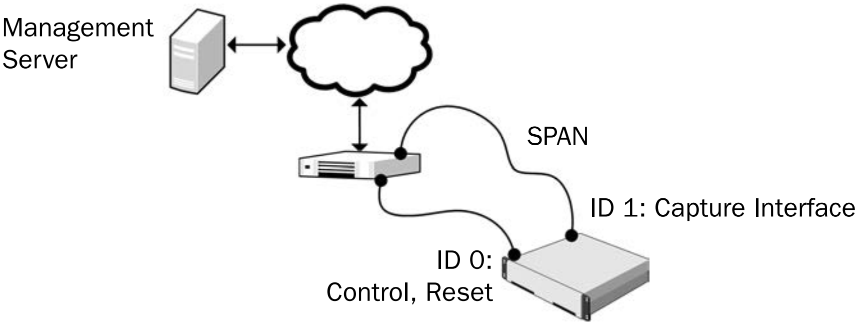
The administrator does the following:

1. Creates a Single Layer 2 Firewall element and selects the Log Server to which the Layer 2 Firewall engine sends its log data.
2. Creates a Logical Interface called Inline for the Inline Interface pair.
3. Defines Interface ID 0 as a normal interface and adds an IP address to it.
4. Defines Interface IDs 1 and 2 as an inline interface pair and selects the Logical Interface called Inline for the pair.
5. Saves the initial configuration of the engine in the Management Client.
6. Connects the network cables to the appropriate physical interfaces on the engine.
7. Maps the interface IDs to the physical interfaces in the Engine Configuration Wizard on the engine's command line and makes initial contact with the Management Server.
8. Installs a Layer 2 Firewall Policy in the Management Client to transfer the configuration to the engine.

Configuring Capture Interfaces in Passive Firewall Mode

The administrator at company B wants to set up a Single Layer 2 Firewall and deploy it in Passive Firewall mode using SPAN ports on the switch to duplicate packets for inspection. Illustration 6.2 shows the interfaces of the Layer 2 Firewall in Passive Firewall mode with Capture Interfaces.

Illustration 6.2 Capture Interfaces with SPAN



In this example, Interface ID 0 is a Normal Interface used for management connections and sending TCP Reset responses. Interface ID 1 is a Capture Interface used for capturing network traffic from the network switch for inspection.

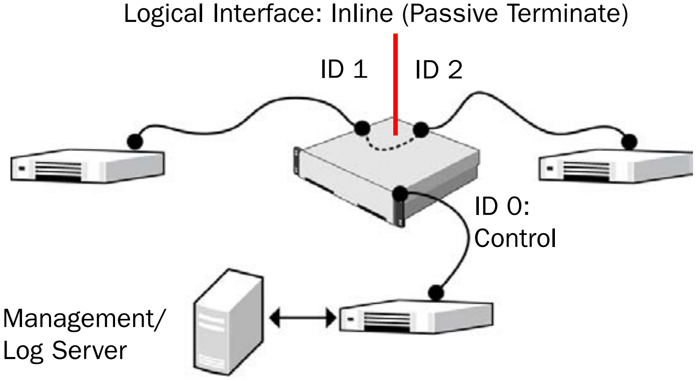
The administrator does the following:

1. Creates a Single Layer 2 Firewall element and selects the Log Server to which it sends log data and the traffic recordings.
2. Defines Interface ID 0 as a Normal Interface and adds an IP address to it.
   - The IP address on Interface ID 0 is automatically selected as the Primary Control IP address because Interface ID 0 is the first Normal Interface with an IP address.
3. Defines Interface ID 1 as a Capture Interface and selects Interface ID 0 as the Reset Interface.
4. Saves the initial configuration of the engine in the Management Client.
5. Maps the interface IDs to the physical interfaces in the Engine Configuration Wizard on the engine’s command line and makes initial contact with the Management Server.
6. Installs a Layer 2 Firewall Policy in the Management Client to transfer the configuration to the engine.

Configuring Inline Interfaces in Passive Firewall Mode

The administrator at company C wants to set up a Single Layer 2 Firewall and deploy it in Passive Firewall mode in an inline configuration. Illustration 6.3 shows the interfaces of the Single Layer 2 Firewall in Passive Firewall mode with Inline Interfaces.

Illustration 6.3 Inline Interfaces in Passive Firewall Mode



In this example, the IP address on Interface ID 0 is configured as the Control IP address for management connections. Interface ID 1 and Interface ID 2 are an inline interface pair that share the same Logical Interface, called Inline (Passive Terminate). Traffic comes in through Interface ID 1 and leaves through Interface ID 2.

The administrator does the following:

1. Creates a Single Layer 2 Firewall element and selects the Log Server to which the Layer 2 Firewall engine sends its log data.
2. Creates a Logical Interface called Inline (Passive Terminate) for the Inline Interface pair.
3. Defines Interface ID 0 as a normal interface and adds an IP address to it.
4. Defines Interface IDs 1 and 2 as an inline interface pair and selects the Logical Interface called Inline for the pair.
5. Configures the Layer 2 Firewall engine to only create Terminate (passive) log entries for all connections that match the Access rules with the Discard action in the Layer 2 Firewall Policy and all the Inspection rules with the Terminate action in the Inspection Policy.
6. Saves the initial configuration of the engine in the Management Client.
7. Connects the network cables to the appropriate physical interfaces on the engine.
8. Maps the interface IDs to the physical interfaces in the Engine Configuration Wizard on the engine’s command line and makes initial contact with the Management Server.
9. Installs a Layer 2 Firewall Policy in the Management Client to transfer the configuration to the engine.

CHAPTER 7

MASTER ENGINE AND VIRTUAL IPS CONFIGURATION

A *Virtual Security Engine* is a logically-separate engine that runs as a virtual engine instance on a physical engine device. A *Virtual IPS Engine* is a Virtual Security Engine in the IPS role. A *Master Engine* is a physical engine device that provides resources for Virtual Security Engines.

The following sections are included:

Overview of Master Engine and Virtual IPS Engine Configuration

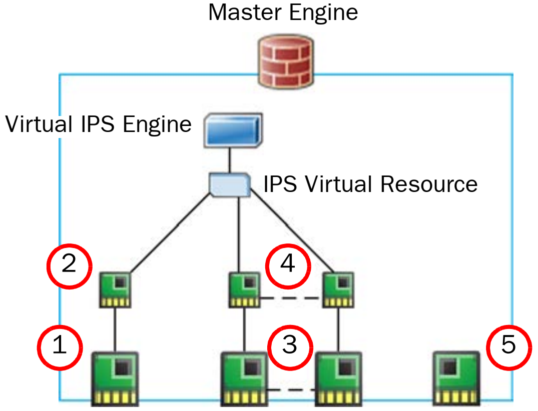
This chapter focuses on Virtual Security Engines in the IPS role. Virtual Security Engines in the IPS role are configured using Virtual IPS elements in the Management Client.

Using Virtual IPS engines allows the same physical engine device to support multiple policies or policies that involve overlapping IP addresses. This is especially useful in a Managed Security Service Provider (MSSP) environment, or in a network environment that requires strict isolation between networks.

Configuration of Master Engines and Virtual IPS Engines

Any NGFW engine that has a license that allows the creation of Virtual Resources can be used as a Master Engine.

Illustration 7.1 Master Engine and Virtual IPS Engine Architecture



One physical Master Engine can host multiple Virtual IPS engines. A *Virtual Resource* element defines the set of resources on the Master Engine that are allocated to a Virtual IPS engine. Virtual Resource elements associate Virtual IPS engines with Physical Interfaces or VLAN Interfaces on the Master Engine. The license for the Master Engine defines how many Virtual Resources can be created. The number of Virtual Resources limits the number of Virtual IPS engines: one Virtual IPS engine at a time can be associated with each Virtual Resource.

Master Engines can have two types of interfaces: interfaces for the Master Engine's own traffic, and interfaces that are used by the Virtual IPS engines hosted on the Master Engine. In the example above, the Master Engine has the following kinds of interfaces:

1. Capture Interface for hosted Virtual IPS engine traffic.
2. VLAN Interface for hosted Virtual IPS engine traffic.
3. Inline Interface pair for hosted Virtual IPS engine traffic.
4. Inline VLAN Interface pair for hosted Virtual IPS engine traffic.
5. Normal Interface for the Master Engine's own traffic.

Configuration Workflow

The following sections provide an overview of the configuration tasks. Detailed step-by-step instructions can be found in the Management Client *Online Help* and the *McAfee SMC Administrator's Guide*.

Task 1: Create a Master Engine Element

To introduce a new Master Engine to the SMC, you must define a Master Engine element that stores the configuration information related to the Master Engine and the Virtual Resources. Each Master Engine can support one Virtual Security Engine role (Firewall/VPN, IPS, or Layer 2 Firewall). You must select the role for the Virtual Security Engines that the Master Engine will host. You cannot change the role of the Virtual Security Engine hosts after you create the Master Engine element.

By default, Master Engine elements are created with two nodes. You can optionally add or remove nodes. Each Master Engine can consist of 1 to 16 nodes.

Task 2: Create Virtual Resource Element(s)

Virtual Resource elements associate Virtual IPS engines with Physical Interfaces or VLAN Interfaces on the Master Engine. When you select the same Virtual Resource for a Physical Interface or VLAN Interface on the Master Engine and for a Virtual IPS engine, the Virtual IPS engine is automatically associated with the Master Engine.

Task 3: Configure Master Engine Interfaces

You can add Physical Interfaces and VLAN Interfaces to a Master Engine. If you want to use a Physical Interface to host a Virtual IPS engine, you can define the interface as an Inline Interface or as a Capture Interface. You must select a Virtual Resource for interfaces that are used to host Virtual IPS engines. The same Virtual Resource can be used on more than one Master Engine interface to allocate multiple interfaces to the same Virtual IPS engine. If you want the Virtual IPS engine to have multiple interfaces, you must use the same Virtual Resource on more than one Master Engine interface.

If you want to use a Physical Interface or VLAN Interface for the Master Engine's own traffic, you must define the interface as a Normal Interface. You can configure several IPv4 addresses on each Physical Interface that is defined as a Normal Interface, or any VLAN Interface that belongs to a Normal Interface. Because the interfaces for the Master Engine's own traffic are only used for system communications, only IPv4 addresses are supported on interfaces for the Master Engine's own traffic.

By default, the Physical Interface definitions for the Master Engine are mapped to the actual network interfaces on the Master Engine hardware in numerical order. If necessary, you can change the mapping using command line tools on the Master Engine. This mapping can be done differently from one Master Engine node to another as long as you make sure that the interface that represents the same network interface on each Master Engine node is correctly cabled to the same network.

Task 4: Create a Virtual IPS Element

Virtual IPS elements store the configuration information related to the Virtual IPS engines. Selecting a Virtual Resource for the Virtual IPS engine associates the Virtual IPS engine with the Master Engine where the Virtual Resource is used.

Task 5: Configure Virtual IPS Interfaces

Physical Interfaces in the properties of a Virtual IPS engine represent interfaces allocated to the Virtual IPS engine in the Master Engine. All communication between Virtual IPS engines and the SMC is proxied by the Master Engine.

Physical Interfaces for the Virtual IPS engine are automatically created based on the interface configuration in the Master Engine properties. The number of Physical Interfaces depends on the number of interfaces allocated to the Virtual IPS engine in the Master Engine. You can optionally modify the automatically created Physical Interfaces.

In addition to the automatically created Physical Interfaces, you can add VLAN Interfaces if the creation of VLAN Interfaces for Virtual IPS engines is enabled in the Master Engine properties.

By default, the interface definitions for the Virtual IPS engine are mapped to interfaces on the Master Engine in the order in which the interfaces are created on the Master Engine.

Task 6: Install a Policy

There is no separate type of policy for Master Engines. Master Engines use Firewall Policies regardless of the role of the Virtual Security Engines that they host. After you have modified Physical Interfaces, VLAN Interfaces, or interface mapping for Virtual IPS engines in the Master Engine properties, you must install or refresh the policy on the Master Engine to transfer changes.

Virtual IPS engines use IPS Polices. You can install a different IPS Policy on each Virtual IPS engine. After you have modified Physical Interfaces or VLAN Interfaces in the Virtual IPS engine properties, you must install or refresh the policy on the Virtual IPS engine to transfer the changes.

Using Master Engines and Virtual IPS Engines

The main points of Master Engine and Virtual IPS engine configuration are explained in the preceding sections of this chapter. This section illustrates some additional concepts for working with Master Engines and Virtual IPS engines:

- Moving a Virtual IPS Engine to a Different Master Engine
- Using Master Engines and Virtual IPS Engines With Domains

Moving a Virtual IPS Engine to a Different Master Engine

The Virtual Resource selected in the properties of a Virtual IPS engine element determines the Master Engine to which the Virtual IPS engine belongs. To move a Virtual IPS engine to a different Master Engine, you select a Virtual Resource that is associated with a different Master Engine in the Virtual IPS engine properties. The move takes effect only after you refresh the policy on the Master Engine.

Using Master Engines and Virtual IPS Engines With Domains

If there are multiple administrative Domains, Virtual IPS engines associated with the same Master Engine can belong to different Domains. However, the Master Engine must either belong to the Shared Domain or to the same Domain as the associated Virtual IPS engines. For example, the Master Engine can belong to the Shared Domain, while each associated Virtual IPS engine belongs to a different Domain.

CHAPTER 8

MASTER ENGINE AND VIRTUAL LAYER 2 FIREWALL CONFIGURATION

A *Virtual Security Engine* is a logically-separate engine that runs as a virtual engine instance on a physical engine device. A *Virtual Layer 2 Firewall* is a Virtual Security Engine in the Layer 2 Firewall role. A *Master Engine* is a physical engine device that provides resources for Virtual Security Engines.

The following sections are included:

Overview of Master Engine and Virtual Layer 2 Firewall Configuration

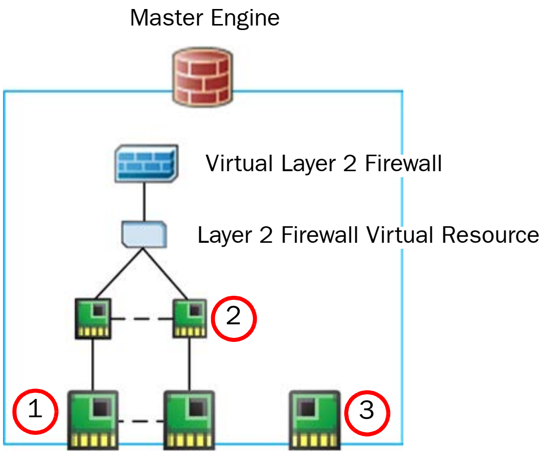
This chapter focuses on Virtual Security Engines in the Layer 2 Firewall role. Virtual Security Engines in the Layer 2 Firewall role are configured using Virtual Layer 2 Firewall elements in the Management Client.

Using Virtual Layer 2 Firewalls allows the same physical engine device to support multiple policies or policies that involve overlapping IP addresses. This is especially useful in a Managed Security Service Provider (MSSP) environment, or in a network environment that requires strict isolation between networks.

Configuration of Master Engines and Virtual Layer 2 Firewalls

Any NGFW engine that has a license that allows the creation of Virtual Resources can be used as a Master Engine.

Illustration 8.1 Master Engine and Virtual Layer 2 Firewall Architecture



One physical Master Engine can host multiple Virtual Layer 2 Firewalls. A *Virtual Resource* element defines the set of resources on the Master Engine that are allocated to a Virtual Layer 2 Firewall. Virtual Resource elements associate Virtual Layer 2 Firewalls with Physical Interfaces or VLAN Interfaces on the Master Engine. The license for the Master Engine defines how many Virtual Resources can be created. The number of Virtual Resources limits the number of Virtual Layer 2 Firewalls: one Virtual Layer 2 Firewall at a time can be associated with each Virtual Resource.

Master Engines can have two types of interfaces: interfaces for the Master Engine's own traffic, and interfaces that are used by the Virtual Layer 2 Firewalls hosted on the Master Engine. In the example above, the Master Engine has the following kinds of interfaces:

1. Inline Interface for hosted Virtual Layer 2 Firewall traffic.
2. Inline VLAN Interface for hosted Virtual Layer 2 Firewall traffic.
3. Normal Interface for the Master Engine's own traffic.

Configuration Workflow

The following sections provide an overview of the configuration tasks. Detailed step-by-step instructions can be found in the Management Client *Online Help* and the *McAfee SMC Administrator's Guide*.

Task 1: Create a Master Engine Element

To introduce a new Master Engine to the SMC, you must define a Master Engine element that stores the configuration information related to the Master Engine and the Virtual Resources.

Each Master Engine can support one Virtual Security Engine role (Firewall/VPN, IPS, or Layer 2 Firewall). You must select the role for the Master Engine when you create a new Master Engine element. The selected role determines which types of interfaces can be configured for the Master Engine.

By default, Master Engine elements are created with two nodes. You can optionally add or remove nodes. Each Master Engine can consist of 1 to 16 nodes.

Task 2: Create Virtual Resource Element(s)

Virtual Resource elements associate Virtual Layer 2 Firewalls with Physical Interfaces or VLAN Interfaces on the Master Engine. When you select the same Virtual Resource for a Physical Interface or VLAN Interface on the Master Engine and for a Virtual Layer 2 Firewall, the Virtual Layer 2 Firewall is automatically associated with the Master Engine.

Task 3: Configure Master Engine Interfaces

You can add Physical Interfaces and VLAN Interfaces to a Master Engine. If you want to use a Physical Interface or VLAN Interface to host a Virtual Layer 2 Firewall, you can define the interface as an Inline Interface or as a Capture Interface. You must select a Virtual Resource for interfaces that are used to host Virtual Layer 2 Firewalls. The same Virtual Resource can be used on more than one Master Engine interface to allocate multiple interfaces to the same Virtual Layer 2 Firewall. If you want the Virtual Layer 2 Firewall to have multiple interfaces, you must use the same Virtual Resource on more than one Master Engine interface.

If you want to use a Physical Interface or VLAN Interface for the Master Engine's own traffic, you must define the interface as a Normal Interface. You can configure several IPv4 addresses on each Physical Interface that is defined as a Normal Interface, or any VLAN Interface that belongs to a Normal Interface. Because the interfaces for the Master Engine's own traffic are only used for system communications, only IPv4 addresses are supported on interfaces for the Master Engine's own traffic.

By default, the Physical Interface definitions for the Master Engine are mapped to the actual network interfaces on the Master Engine hardware in numerical order. If necessary, you can change the mapping using command line tools on the Master Engine. This mapping can be done differently from one Master Engine node to another as long as you take care that the interface that represents the same network interface on each Master Engine node is correctly cabled to the same network.

Task 4: Create a Virtual Layer 2 Firewall Element

Virtual Layer 2 Firewall elements store the configuration information related to the Virtual Layer 2 Firewalls. Selecting a Virtual Resource for the Virtual Layer 2 Firewall associates the Virtual Layer 2 Firewall with the Master Engine where the Virtual Resource is used.

Task 5: Configure Virtual Layer 2 Firewall Interfaces

Physical Interfaces in the properties of a Virtual Layer 2 Firewall represent interfaces allocated to the Virtual Layer 2 Firewall in the Master Engine. All communication between Virtual Layer 2 Firewalls and the SMC is proxied by the Master Engine.

Physical Interfaces for the Virtual Layer 2 Firewall are automatically created based on the interface configuration in the Master Engine properties. The number of Physical Interfaces depends on the number of interfaces allocated to the Virtual Layer 2 Firewall in the Master Engine. You can optionally modify the automatically-created Physical Interfaces.

In addition to the automatically-created Physical Interfaces, you can add VLAN Interfaces to Virtual Layer 2 Firewalls if the creation of VLAN Interfaces for Virtual Layer 2 Firewalls is enabled in the Master Engine Properties.

The interface definitions for the Virtual Layer 2 Firewall are mapped to interfaces on the Master Engine in the order in which the interfaces are created on the Master Engine.

Task 6: Install a Policy

Master Engines use Firewall Policies, regardless of the role of the Virtual Security Engines they host. After you have modified Physical Interfaces, VLAN Interfaces, or interface mapping for Virtual Layer 2 Firewalls in the Master Engine properties, you must install or refresh the Firewall Policy on the Master Engine to transfer the changes.

Virtual Layer 2 Firewalls use Layer 2 Firewall Policies. You can install a different Layer 2 Firewall Policy on each Virtual Layer 2 Firewall. After you have modified Physical Interfaces or VLAN Interfaces in the Virtual Layer 2 Firewall properties, you must install or refresh the policy on the Virtual Layer 2 Firewall to transfer the changes.

Using Master Engines and Virtual Layer 2 Firewalls

The main points of Master Engine and Virtual Layer 2 Firewall configuration are explained in the preceding sections of this chapter. This section illustrates some additional concepts for working with Master Engines and Virtual Layer 2 Firewalls:

- Moving a Virtual Layer 2 Firewall to a Different Master Engine
- Using Master Engines and Virtual Layer 2 Firewalls With Domains

Moving a Virtual Layer 2 Firewall to a Different Master Engine

The Virtual Resource selected in the properties of a Virtual Layer 2 Firewall element determines the Master Engine to which the Virtual Layer 2 Firewall belongs. To move a Virtual Layer 2 Firewall to a different Master Engine, you select a Virtual Resource that is associated with a different Master Engine in the Virtual Layer 2 Firewall properties. The move becomes effective when you refresh the policy on the Master Engine.

Using Master Engines and Virtual Layer 2 Firewalls With Domains

If there are multiple administrative Domains, Virtual Layer 2 Firewalls associated with the same Master Engine can belong to different Domains. However, the Master Engine must either belong to the Shared Domain or to the same Domain as the associated Virtual Layer 2 Firewalls. For example, the Master Engine can belong to the Shared Domain, while each associated Virtual Layer 2 Firewall belongs to a different Domain.

CHAPTER 9

ROUTING AND ANTISPOOFING

Routing defines which network interface IPS engines, Layer 2 Firewalls, and Master Engines select to reach a particular destination address. Antispoofing is the process of defining which addresses are considered valid source addresses for the network(s) connected to each interface.

The following sections are included:

Overview of Routing and Antispoofing

Routing information is used for deciding which network interface is used to reach any given destination address. For the most part, the SMC automates the routing and antispoofing configuration. Much of the configuration is generated automatically based on the IP addresses of the network interfaces.

IPS engines, Layer 2 Firewalls, and Master Engines need modifications in their routing and antispoofing configuration only if other SMC components are located in some other network than the directly connected networks. IPS engines, Layer 2 Firewalls, and Master Engines that host Virtual IPS engines or Virtual Layer 2 Firewalls do not forward traffic from one network to another.

Configuration of Routing and Antispoofing

The routing and antispoofing information is displayed and configured graphically in interface-based trees in the Routing view and Antispoofing view. The routing information is stored on the Management Server. The information is transferred to the engines when the policies are installed or refreshed.

Usually, only a default route is needed for the IPS engines, Layer 2 Firewalls, and Master Engines. These are used when the engines need to open connections to some network other than the directly connected networks. No routes need to be defined if an IPS engine, Layer 2 FIrewall, or Master Engine communicates only in its local IP network.

IP address *spoofing* is an attack where the source IP address in a packet is modified to gain unauthorized access or to cause a *denial-of-service* (DoS). Such attacks can be prevented with antispoofing rules. The antispoofing configuration is generated automatically based on the routing tree. Antispoofing is always enforced on all interfaces. You can change the antispoofing configuration, but in most environments, there is no need to do so.

New Networks are automatically added to routing when you change the engine's interface properties. However, none of the Networks are automatically removed, so you must check your Routing view for obsolete entries and clear them manually. This is to prevent the system from removing currently unused route definitions that you may want to reuse or keep for easy rollback to previous definitions. All additions and deletions in the Routing view are automatically reflected in the Antispoofing view. Manual definitions in the Antispoofing view are preserved regardless of routing changes.

Routing and antispoofing can only be configured for interfaces that have IP addresses. Capture Interfaces and Inline Interfaces cannot have IP addresses. For this reason, it is not possible to configure routing or antispoofing for these types of interfaces. Capture Interfaces and Inline Interfaces are the only type of interfaces that you can define for Virtual IPS engines and Virtual Layer 2 Firewalls. Because of this, it is not possible to configure routing or antispoofing for Virtual IPS engines or Virtual Layer 2 Firewalls. On Master Engines, routing and antispoofing can only be configured for the Master Engine's system communications interfaces.

Default Elements

Networks that correspond to the IP addresses of each interface are automatically added to the routing and antispoofing configurations of IPS, Layer 2 Firewall, and Master Engine elements.

The system includes a default network element called *Any network*, which is needed to define the default route for IPS engines, Layer 2 Firewalls, and Master Engines.

Configuration Workflow

The following sections provide an overview of the configuration tasks. Detailed step-by-step instructions can be found in the Management Client *Online Help* and the *McAfee SMC Administrator's Guide*.

Task 1: Add Router(s)

A Router element represents the next-hop gateway device that forwards packets to the network(s) you define.

Task 2: Add Network(s)

The Network element represents the IP addresses of a network or a subnetwork to which the Router forwards the traffic.

- To add a default route, add the default **Any network** element to the Router you just created.
- If you need to forward traffic to a network that is not directly connected and it cannot be reached through the default gateway, you must define a Network element for this network and add it under the Router.

Task 3: Refresh the Policy

To transfer the routing changes, upload the policy on the IPS engine, Layer 2 Firewall, or Master Engine. The Management Server sends all necessary configuration information when uploading a policy.

CHAPTER 10

BANDWIDTH MANAGEMENT AND TRAFFIC PRIORITIZATION

Bandwidth management involves defining how the available network link bandwidth is divided between different types of communications to ensure that important traffic is not delayed by less important traffic when the network is congested.

Traffic prioritization is used either independently or in addition to bandwidth management to ensure quick delivery of time-critical communications.

The following sections are included:

Overview of Bandwidth Management and Traffic Prioritization

This chapter explains the two Quality of Service (QoS) features: bandwidth management and traffic prioritization. Both features are configured using the same tools. You can use both bandwidth management and traffic prioritization together, or bandwidth management or traffic prioritization individually for any given type of traffic.

The Layer 2 Firewall or IPS engine can also read and write DiffServ Code Point (DSCP) markers in type of service (ToS) fields. This allows you to integrate the Layer 2 Firewall or IPS engine with other network equipment that implements QoS management in your own or your ISP’s network.

Bandwidth Management

Bandwidth management means giving a guaranteed minimum portion of the available bandwidth to some types of traffic and setting limits for how much of the total available bandwidth each type of traffic is allowed to consume at any given time. You can set a limit, a guarantee, or both for any given type of traffic.

Bandwidth management features can be used to ensure the quality of time-critical communications (even under normal network load), to prepare for malfunctions, and to restrict the total bandwidth needed.

Note – Bandwidth management applies to outbound traffic only. Engines can only indirectly limit the bandwidth use of incoming traffic. See Managing Bandwidth of Incoming Traffic (page 102).

Traffic Prioritization

Even under normal traffic conditions, temporary traffic peaks sometimes occur. With many communications, slight delays caused by queuing traffic are not noticeable to the user of the service. However, some connections, such as streaming audio or video, are extremely time-critical, and even relatively minor delays cause a noticeable reduction in service quality.

Normally, when packets are queued, they are sent onwards in the same order in which the packets were received. To change this behavior, you can assign priority values to the traffic. For example, you can assign time-critical connections a high priority. High-priority packets are placed before any lower-priority packets in the queue, allowing the fastest possible delivery.

Active Queue Management (AQM) reduces the volume of dropped or retransmitted packets when there is network congestion. AQM monitors the average queue size and uses a scheduling algorithm to determine the statistical probability for dropping incoming packets. If the queue is almost empty, all packets are accepted. As the queue size increases, the probability for dropping incoming packets also increases. When the queue is full, all packets are dropped.

Effects of Bandwidth Management and Prioritization

Bandwidth management and traffic prioritization improve the quality of service for important traffic, but this may decrease the quality of service for traffic that you define as less important.

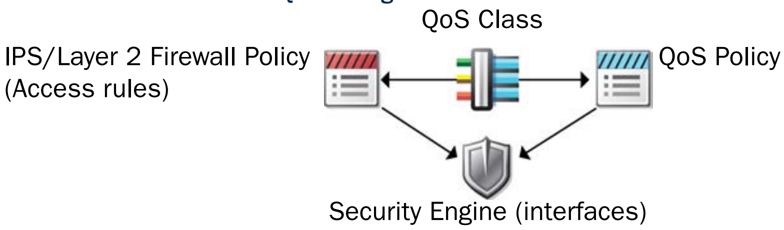
Usually the traffic management process allows all connections to proceed, although some traffic may occasionally slow down when the bandwidth limits are reached. If there is prolonged congestion in the network, lower priority traffic eventually starts to time out. If you set priorities without setting any maximum limits or minimum guarantees for bandwidth, high-priority traffic may even use all available bandwidth, blocking all lower-priority traffic.

In most situations, the guaranteed minimum bandwidths given to important connections allow traffic to proceed. However, even traffic with bandwidth guarantees may not get through if the network links are not able to maintain their defined throughputs or if the volume of traffic continuously exceeds the throughput. Make sure that your bandwidth limits and guarantees are granular enough to account for situations where a part of the bandwidth is lost. Keep track of the total bandwidth use so that you can increase the throughput before problems appear.

Caution – Inappropriate bandwidth limits and guarantees only disturb traffic. Make sure the guarantees and limits you set are appropriate for the volume of each type of traffic.

Configuration of Limits, Guarantees, and Priorities for Traffic

Illustration 10.1 Elements in the QoS Configuration



Bandwidth management and traffic prioritization are configured in *QoS Policies*, which contain rules for the bandwidth guarantees, limits, and priorities you want to set for each type of traffic. The QoS Policies do not contain any traffic profile: to define which QoS rule affects which type of traffic, you use the same *QoS Class* element in the QoS Policy and in one or more Access rules.

The *QoS Mode* for each interface defines how QoS is applied to the interface. You can select a QoS Mode and define a bandwidth for traffic in the properties of a Normal Interface, Inline Interface, or VLAN Interface.

There are two ways the QoS Class can be applied to a packet:

- If the traffic contains a DSCP code when the traffic enters the engine, and DSCP handling and throttling or full QoS are enabled, the engine checks whether the interface that the packets use to enter the engine has a QoS Policy. If the DSCP Match/Mark tab of the QoS Policy defines a QoS Class match for that code, the selected QoS Class is applied to the traffic. See Communicating DSCP Markers (page 101).

- When the traffic is inspected against the IPS or Layer 2 Firewall policy, the traffic may match an Access rule that includes a QoS Class. The QoS Class specified in the QoS Class cell is always applied to the traffic, overwriting any other QoS Class the traffic may have been assigned. Access rules are not needed if you only want to use DSCP handling and throttling.

Using the QoS Class as a matching criterion, the engine checks if the interface that the packet uses to exit the engine has a QoS Policy. If the QoS Policy contains a rule with the same QoS Class defined, the QoS rule is applied to the connection and the packets are dropped, sent directly, or sent into the queue, depending on the QoS rules and the current traffic situation.

Default Elements

There are three default QoS Classes: High Priority, Normal Priority, and Low Priority. These are used in the default QoS Policy, **Prioritize**.

The Prioritize QoS Policy is a sample policy that contains simple rules for prioritizing traffic according to the three default QoS Classes. High Priority traffic is assigned the highest possible priority value of 1, the Normal Priority value is 8, and Low Priority is assigned the lowest possible value of 16. The default Prioritize policy does not provide any bandwidth guarantees or limits.

Caution – If you set priorities without setting any bandwidth limits or guarantees, high-priority traffic may use all available bandwidth, blocking all lower-priority traffic.

If the default Prioritize policy is sufficient for you, you can use the default QoS Classes and the Prioritize policy as they are. Just add the QoS Classes to Access rules as explained in Task 3: Assign QoS Classes to Traffic (page 98). Then, configure the interface(s) to use the Prioritize QoS Policy. See Task 4: Define QoS for Interfaces (page 99).

If you want to define bandwidth guarantees or limits, or if you want to have more control over the traffic priorities, you must configure QoS as explained in the configuration workflow below.

Configuration Workflow

The following sections provide an overview of the configuration tasks. Detailed step-by-step instructions can be found in the Management Client *Online Help* and the *McAfee SMC Administrator's Guide*.

Task 1: Define QoS Classes

QoS Classes can be used to collect QoS statistics about traffic, or to create a link between the Access rules and the QoS Policy. When traffic matches an Access rule, the QoS Class defined in the rule is applied to the traffic.

You can create as many QoS Classes as necessary. The QoS Policy must not have overlapping rules. For this reason, you must create one QoS Class for each rule you plan to add in the QoS Policy (each type of traffic must have its own QoS Class). The same QoS Class can be used in multiple Access rules, so several Access rules can point matching traffic to the same QoS rule.

Task 2: Define QoS Policies

QoS Policies are tables of QoS rules and DSCP Match/Mark rules. If you only want to collect QoS statistics about traffic, you do not need to define a QoS Policy.

Because the QoS rules are separate from the Access rules, you can flexibly design the rules. For example, you can create different QoS Policies for different interfaces of the same engine.

All cells in the QoS rules are applied to outgoing packets. When Full QoS is used, packets that do not match a QoS rule are handled with priority 8 (middle of the scale) without any bandwidth guarantees or limits. The cells in the QoS rules are explained in the tables below.

Table 10.1 QoS Rule Cells

Cell	Explanation
ID	An identifier that shows the order of the rules. The number changes as you add, remove, and move rules.
QoS Class	A list of the defined QoS Classes, which match the QoS rules to traffic. The QoS Class is assigned to traffic in Access rules or through the DSCP Match cell in the QoS Policy (see Table 10.2).
Guarantee	Sets the minimum bandwidth given to this type of traffic under any conditions. The guarantee can be set in kilobits per second or as a percentage of the available bandwidth.
Limit	Sets the maximum bandwidth that this type of traffic is allowed to consume at any single moment as kilobits per second or as a percentage of the available bandwidth.
Priority	Assigns this type of traffic a number that determines the order in which the engine sends packets onwards if there is not enough bandwidth available to send all packets onwards directly, so that packets have to be queued. The priority is a number between 1 (highest priority) and 16 (lowest priority). Higher-priority packets are inserted in the queue ahead of any lower-priority packets already in the queue.
Weight	The weight of the QoS Class controls the distribution of bandwidth between QoS Classes with the same priority after the Guarantees for the QoS Classes are reached. The weight of the QoS Class is entered as a value from 0 to 100. The relative weight of each QoS Class is displayed in parentheses as a percentage.
Latency	The average time packets are held in the queue for Active Queue Management (AQM). The engine makes a best effort to handle the packets within the specified time, but the Latency value is not a guarantee.
Comment	A short free-form comment for your reference.
Rule Name	Contains a rule tag and optionally a rule name. Name: You can optionally also add a name for the rule, which is displayed alongside the rule tag. Rule tag: (Not editable) The unique identifier of the rule in this policy. It contains a static part that does not change when rules are added, removed, or moved, and a changing part that indicated the version of the rule.

All cells in the DSCP Match/Mark rules except the DSCP Match cell are applied to outgoing packets. If packets do not match any DSCP Match/Mark rule, any DSCP markers in the traffic are preserved, but have no effect on how the engine handles the traffic. The cells in the DSCP Match/Mark rules are explained below.

Table 10.2 DSCP Match/Mark Rule Cells

Cell	Explanation
ID	An identifier that shows the order of the rules. The number changes as you add, remove, and move rules.
QoS Class	A list of the defined QoS Classes, which match the QoS rules to traffic. The QoS Class is assigned to traffic in Access rules or through the DSCP Match cell (see below).
DSCP Match	Assigns the rule's QoS Class to traffic when the DSCP code (ToS field) defined in this cell is seen in traffic. The value specified in this cell is the only option that is applied on the interface that the packets use to enter the engine.
DSCP Mark	Defines the DSCP code (ToS field) that is written to packets that match this DSCP Match/Mark rule when the packets exit the engine. The DSCP Mark allows you to communicate the priority of this traffic to other devices that support QoS. You can also use the cell to clear the DSCP classification set by other devices by entering **0** as the value (shown in the policy as 0x00).
Comment	A short free-form comment for your reference.
Rule Name	Contains a rule tag and optionally a rule name. Name: (*Optional*) Name or description for the rule. Displayed alongside the rule tag. Tag: (*Not editable*) The unique identifier of the rule in this policy. It contains a static part that does not change when rules are added, removed, or moved, and a changing part that indicated the version of the rule.

Task 3: Assign QoS Classes to Traffic

The same QoS Class can appear in several Access rules. You can insert a QoS Class in an Access rule that allows traffic or in an Access rule that uses the Continue action to set the same QoS Class for several rules. This way, you can assign a specific QoS Class to any traffic that you can match with a single Access rule. If you only want to collect QoS statistics about traffic, you only need to define Access rules to assign a QoS Class to the traffic.

If you only want to map existing DSCP codes in traffic to QoS Classes, no Access rules are required. However, the same traffic must not match an Access rule that sets a QoS Class, since the Access rule overwrites the QoS Class that is assigned based on the DSCP code.

Task 4: Define QoS for Interfaces

The *QoS Mode* for each interface defines how QoS is applied to the interface. By default, No QoS is selected. You can select a QoS Mode and define a bandwidth for traffic in the properties of a Normal Interface, Inline Interface, or VLAN Interface. You can select different QoS Modes for each interface. It is not mandatory to use QoS on all interfaces of the same engine. The following table describes how QoS is applied for each QoS Mode.

Table 10.3 QoS Modes

QoS Mode	Explanation
No QoS	None of the QoS features are applied to the interface.
QoS Statistics Only	Allows the collection of QoS Class-based counters without activating any other QoS functionality. A QoS Policy is not required. The QoS Class for the packet comes from the QoS Classes that are applied in the Access rules. See Collecting QoS Class-Based Statistics (page 102).
DSCP Handling and Throttling	QoS guarantees, limits, and priorities are applied to the traffic either according to the QoS Classes set by the Access rules or by a DSCP Match in the QoS Policy. DSCP markers are read and/or written for the traffic according to the QoS Policy. You must specify bandwidth guarantees and limits in the QoS Policy in kilobits per second. Only a QoS Policy is required.
Full QoS	QoS guarantees, limits, and priorities are applied to the traffic according to the QoS Policy. DSCP markers are read and/or written for the traffic according to the QoS Policy. You can specify bandwidth guarantees and limits in the QoS Policy in kilobits per second or as a percentage of the available bandwidth. A QoS Policy and bandwidth definitions for the interface are required.

When you use Full QoS on an interface, you must define the available throughput in the Interface Properties for each Normal Interface, Inline Interface, or VLAN Interface whose throughput you want to manage. There is no automatic mechanism for finding out how much bandwidth each interface has. The throughput must always correspond to the actual throughput the interface offers to connecting clients, that is, the outbound bandwidth of an Internet link that is connected to the interface. If there are VLANs on a Physical Interface, the settings are only available in the properties of each VLAN.

Note – When you define the throughput of an interface, the engine always scales the traffic to this throughput. Take special care that you set the throughput value correctly.

Using Bandwidth Management and Traffic Prioritization

Bandwidth management and traffic prioritization are generally used for the following purposes:

- To ensure the quality of service for time-critical communications during occasional traffic peaks. This may be necessary even if there is generally ample bandwidth available, since even very short periods of congestion may degrade the quality of some types of communications.
- To prepare for severe congestion caused by the loss of network links when there are technical problems.
- To restrict non-essential services to reduce the total bandwidth needed if it is not possible to increase throughput of the network links or add new links. For example, important services can be given priority at the expense of less important services.

Designing QoS Policies

Each QoS Class can appear in only one (active) rule on each tab of a QoS Policy. The same QoS Class can be used on both the QoS tab and the DSCP Match/Mark tab of the same QoS Policy. The order of the QoS rules does not matter. The classification of the traffic is made using Access rules, so the purpose of the QoS Policy is to determine which limit, guarantee, or priority traffic marked with a certain QoS Class receives.

Except for the QoS Class, all other cells for rules on the QoS tab are optional, but at least one of the other cells must be filled for the rule to have any effect on the traffic. None of the cells exclude any of the other cells, so you are free to select which cells you want to use for any given QoS Class. It is not necessary to define the use of all available bandwidth in your QoS Policy. The bandwidth outside the guarantees as well as any bandwidth within the guarantees that is not used for actual traffic at any given time is used to handle the traffic that has no specific QoS rule on the normal first-come-first-served-basis using the medium priority of 8.

Caution – If your guarantees are equal to the total throughput of an interface, any traffic without a guarantee is completely blocked if all guarantees are fully used.

When you save the QoS Policy, the system checks if there are contradictions within each rule, such as a rule that sets a limit that is lower than the guarantee for the same rule. When you refresh the engine’s configuration, the QoS Policies defined for the engine’s interfaces are checked again, comparing the QoS rules to the throughput values you have set for the interfaces. At this point, the values may be automatically scaled down if the sum of all the guarantees in the QoS Policy exceeds the interface’s throughput. However, the values are never scaled up.

The values for the bandwidth limits and guarantees can be entered either in kilobits or as percentages of the total throughput of the interface. Technically, nothing prevents you from using both ways of entering values even in the same rule. However, it is recommended to use one way of entering the values consistently throughout each QoS Policy. Using mixed methods of entering the values makes it more difficult for the administrators to read the QoS policy and may prevent the system from checking for issues when you save the QoS Policy, as the throughput(s) of the interfaces are not known at this point. If the QoS Policy cannot be checked when you save it, it is checked when the IPS or Layer 2 Firewall Policy is installed. Mistakes in the QoS Policy may prevent you from installing or refreshing the policy.

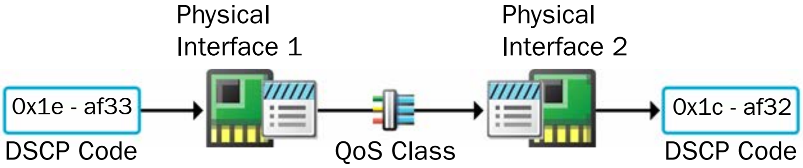
Communicating DSCP Markers

You and your ISP may have routers that also make decisions on handling the packets based on the priority of the traffic. DSCP (DiffServ type of service field) markers in the traffic are a standard way to indicate priorities in network traffic. You can communicate traffic priorities between the engine and other network equipment using DSCP to ensure the best possible quality of service for important traffic.

It is possible to apply a DSCP mark to a particular type of traffic without configuring Access rules to apply a QoS Class to the traffic. The matching is done based only on the QoS Policy. When a packet that matches a particular protocol comes in, the DSCP markers are read and assigned a QoS Class according to the DSCP Match/Mark rules of the QoS Policy. When the packet is sent out, a DSCP mark is written in packets based on the QoS Class according to the DSCP Match/Mark rules of the QoS Policy on the interface through which the traffic leaves the engine.

Two QoS Policies on two Physical Interfaces can be used together to “translate” between two different DSCP schemes as shown in Illustration 10.2.

Illustration 10.2 Translating Between Two DSCP Schemes



In the illustration above, the packets arrive at Physical Interface 1. The engine reads the existing DSCP value and compares it to the QoS Policy assigned to Physical Interface 1. The policy has a DSCP Match rule for the DSCP marker with an associated QoS Class, which is then assigned to this traffic.

Note – The same traffic must not match any Access rule with a QoS Class definition, because the QoS Class in the Access rule overrides the QoS Class that is assigned based on the DSCP marker.

When the packets are sent out through Physical Interface 2, the engine checks the QoS Policy assigned to this Physical Interface. In this QoS Policy, there is a DSCP Match/Mark rule that defines that traffic with the assigned QoS Class will be marked with a DSCP marker specified in the rule, and the engine overwrites the original DSCP marker before sending the packets onwards.

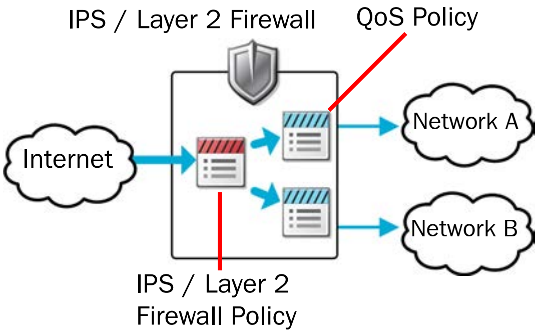
Managing Bandwidth of Incoming Traffic

Bandwidth management and prioritization usually help manage the quality of service for traffic going out through Internet links, which are often the choke point in a corporate network due to the costs associated with increasing the bandwidth.

Outbound traffic can be controlled easily and accurately with QoS Policies, since the engine has full control of what traffic it forwards. Controlling incoming traffic is more difficult, since by the time the engine sees the traffic, the packets have already travelled through the congested links and taken their share of the limited bandwidth. Still, you may be able to limit some types of incoming traffic in a limited way. In this case, only limits apply. To set guarantees and priorities for traffic, you must consider other solutions, such as arranging with your ISP(s) that they implement traffic management before the traffic is passed to your Internet links.

To limit the bandwidth incoming traffic consumes, you can apply the QoS Policy on the engine's interfaces connected to the *internal* network. This arrangement is shown in Illustration 10.3.

Illustration 10.3 Applying QoS to Incoming Traffic



In the illustration above, traffic is checked against the policy and allowed traffic is assigned a QoS Class. At the interfaces connected to the internal network, the QoS Policies limiting the bandwidth use are enforced as the traffic is sent onwards.

Limiting the bandwidth of incoming traffic in this way requires that the application that is the source of the traffic scales down the transmissions to match the available bandwidth. If an application does not scale down its bandwidth use, any limits you set have no effect, and the only option is to control the traffic before it reaches the engine (by your ISP).

Collecting QoS Class-Based Statistics

QoS Class-based statistics items are available even when QoS is not used for bandwidth management and traffic prioritization. Selecting the “QoS Statistics Only” QoS Mode for an interface allows the collection of QoS Class-based counters without activating any other QoS functionality. No QoS Policy is needed in this case, but you must define Access rules to apply QoS Classes to traffic. QoS Class-based statistics items can be used in Overviews and Reports.

TRAFFIC INSPECTION

In this section:

CHAPTER 11

SITUATIONS

Situation elements collect together the information that identifies and describes detected events in the traffic (or in the operation of the system). Situations contain the context information, that is, a pattern that the system is to look for in the inspected traffic.

The following sections are included:

Overview of Situations

Situations define the traffic patterns and events you want to detect in the Inspection Policy. The patterns and events are defined by selecting a *Context* for the Situation. The Context contains the information on the traffic to be matched, and the options you can set for the matching process.

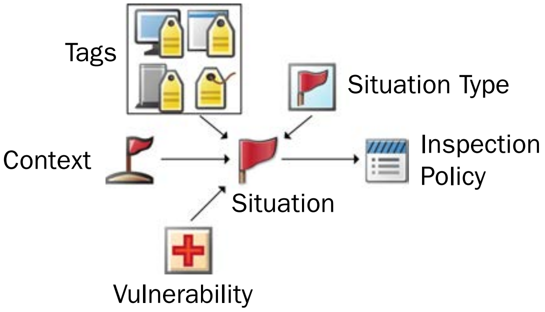
Situations also provide a description that is shown in the logs, and a link to relevant external information (CVE/BID/MS/TA) in the form of a *Vulnerability* element attached to the Situation.

The Inspection Policy defines how the Situations are matched to traffic and what kind of action the engine takes when a match to a particular Situation is found. Correlation Situations are a special type of Situations that group together event data to find patterns in that data.

Configuration of Situations

The illustration below shows how Situations and the related elements are used together.

Illustration 11.1 A Situation and the Associated Elements



The Situation element uses different elements to form a representation of the traffic that you want to detect in your Inspection Policy. The purpose of these elements is as follows:

- The **Tag** elements help you to create simpler policies with less effort. Tag elements represent all Situations that are associated with that Tag. For example, using the Tag “Windows” in a rule means that the rule matches all the Situations that concern Windows systems.
- The **Situation Type** elements define the general category of the Situation and the branch of the Rules tree under which the Situation appears (Attacks, Successful Attacks, etc.). One Situation Type can be associated with each Situation.
- The **Context** element defines the traffic patterns the Situation detects. The Context binds the Situation to a certain type of traffic and gives you a set of options or a field for entering a regular expression.
- The **Vulnerability** element associates your custom Situation with a commonly known vulnerability. It allows you to attach a description of the Vulnerability and references to public vulnerability databases (which are shown in the Logs view if a match is found).

The Context is the only mandatory element in a Situation. However, it is recommended to consistently associate all relevant Tags with each custom Situation you create. The vulnerability description is not mandatory, but it is helpful to have it for Situations that detect some publicly known issue.

Situation Contexts

Context elements are protocol-specific, so they define what the Situation element matches. They provide a framework for defining the parameters of each Situation. The parameters are entered as a regular expression or through a set of fields and options that you can adjust, depending on the Context element selected. The properties of each Context provide assistance on filling in the parameters for the Contexts.

The sections below explain the types of Context elements available and how they can be configured.

Note – The details related to the Contexts in your system may be different from what is described here because the Contexts may have been updated through dynamic update packages after this guide was published. Read the release notes of each update package you import to see which elements are affected.

Correlation Contexts

Correlation Contexts define the patterns for matching groups of related events in traffic. There are five types of Correlation Contexts:

Table 11.1 Correlation Context Types

Correlation Context Type	Description
Compress	Combines repeated similar events into the same log entry, reducing clutter in the Logs view. **Example**: There is a custom Situation for detecting suspicious access to a file server. An attacker is likely to browse through many files, triggering an alert entry for each file. An Event Compress Situation can be used to combine Situations together when the suspect's IP address is the same.
Count	Finds recurring patterns in traffic by counting how many times certain Situations occur within the defined period, so that action can be taken if the threshold values you set are exceeded. **Example**: A Situation that detects access to a system could normally trigger just a log entry, but the Event Count Situation could be used to blacklist connections when access by any single host is too frequent.
Group	Finds event patterns in traffic by keeping track of whether all events in the defined set of Situations match at least once in any order within the defined time period. **Example**: Individual attempts to exploit different vulnerabilities in a software product in use on your server may not be too alarming if you know that your system is patched against those vulnerabilities. However, when several such events are found in a short period of time, it becomes more likely that someone is trying to systematically attack the server and already knows that the server is running that particular piece of software. A Situation that belongs to the Group Context can detect this.
Match	Allows you to use Filters to filter event data produced by specific Situations.

Table 11.1 Correlation Context Types (Continued)

Correlation Context Type	Description
Sequence	Finds event patterns in traffic by keeping track of whether all events in the defined set of Situations match in a specific order within the defined time period. **Example**: Clients may use a certain type of request (e.g., “give file X”) to fetch a file from a file server. When administrators log in to the same server, a successful administrator login can be seen in the traffic as a certain type of response (e.g., “full access granted”). However, a vulnerability in the server software may allow an attacker to send a specially crafted file fetch request that looks like a valid “give file x” command, but actually causes the server to give the attacker administrator access. This is seen as a normal-looking “full access granted” response from the server. The Event Sequence Situation can detect when a “give file X” Situation match is followed by a “full access granted” Situation match, which cannot be any legitimate traffic.

Detailed descriptions of the parameters for each of the Correlation Contexts can be found in Situation Context Parameters (page 229).

DoS Detection Contexts

The DoS Detection Contexts provide parameters for detecting DoS (Denial of Service) events in network traffic.

Scan Detection Contexts

The Scan Detection Contexts provide parameters for detecting attempts to scan which IP addresses are in use or which ports are open in your systems.

Protocol-Specific Contexts

The protocol-specific Contexts are used to detect a particular characteristic in the network traffic. For example, you can detect a certain option number used in IP packets, or set the maximum length for particular arguments in FTP commands. You can also use the HTTP URL Filter to allow or deny access to specific websites (*not supported on Layer 2 Firewalls*).

For Contexts that have particular values to be filled in (instead of a regular expression), the parameters you define in the Contexts often actually determine what is considered normal, so that anything above/below/outside/not matching these values is considered a match for the Situation. In some cases, you may define what the Situation *does not* match.

Effective modifications to the protocol-specific Contexts require you to be familiar with the protocols in question and how the traffic in your network uses those protocols.

File Contexts

The File Contexts are used to detect malicious or suspicious content in transferred files regardless of the transport protocol used. When a file is detected, the file is inspected to identify the file type. Once the file type is identified, more specific inspection can be applied to the file.

System Contexts

The System Contexts are used for errors and other system events. They are internal to the SMC, and they cannot be modified in any way.

Default Elements

There are many predefined Contexts, Situations, Tags, and Vulnerabilities available, which are imported and updated from dynamic update packages. This also means that the set of elements changes whenever you update your system with new definitions. Both Situation elements and Context elements have a comment and a longer description that you can view in the Management Client (in the Info panel or in the Properties dialog for the element) to see what each element is meant for.

The Release Notes of each dynamic update package list the new elements that the update introduces. If your Management Server can connect to the McAfee website, you can view the release notes directly through the Management Client.

Configuration Workflow

The following sections provide an overview to the configuration tasks. Detailed step-by-step instructions can be found in the *Online Help* of the Management Client and the *McAfee SMC Administrator’s Guide* PDF.

Task 1: Create a Situation Element

You can create new Situations in addition to using the predefined ones. You can create a Situation element to detect individual events or a Correlation Situation element to detect a group of related events. Situation elements can also be defined automatically based on Snort rules when you import a Snort rules library. See Importing Snort Rules Libraries (page 160) for more information.

A Situation element collects together the related elements and settings and sets the severity value for the Situation. The severity value can be set between Info (the least severe) to Critical (the most severe). You can use the severity value to restrict which Situations added to the Situations cell are considered in Inspection Exceptions and Alert Policies. For example, if a rule matches a large range of Situations you can create separate rules for less severe and more severe Situations.

Task 2: Add a Context for the Situation

Adding a Context to a Situation allows you to define what kinds of patterns you want to look for in the traffic. For example, you can specify that you want to look for a certain character sequence in an HTTP stream from the client to the server.

Note – With the exception of whitelisted URLS in URL Filtering, Situations are identified only by the element name. Avoid matching the same pattern in different Situation elements. Situations with duplicate patterns can make the policy difficult to read and manage.

When you select a Context you get a set of options or a field for entering a regular expression as parameters for the Context. The parameters define the pattern you want to look for in the traffic. The syntax for SMC regular expressions is explained in Regular Expression Syntax (page 235). Correlation Situation parameters are explained in Situation Context Parameters (page 229).

Other types of context parameters are not listed in this guide. They concentrate on some aspect of a particular kind of network traffic, and using them requires basic knowledge of the underlying network protocols. For more information on what a particular Context is used for, see the Properties dialog of the Context in question.

Task 3: Associate Tags and/or Situation Types with the Situation

You can use Tag elements to group Situations and Situation Types to classify Situations. You can use predefined Tags or create new ones according to any criteria (for example, create a Tag for grouping together related services). Situation Types are predefined, and you cannot create new Situation Types. You can associate multiple Tags with one Situation, but only one Situation Type can be associated with each Situation.

You can use the Tags and/or Situation Types to represent a group of Situations in the Rules and Exceptions of the Inspection Policy. This allows you to match a rule to all Situations that contain the Tag or Situation Type. Situations that are associated with a Situation Type are automatically included in the Rules tree. See Inspection Policies (page 151) for more information.

Note – If a Tag or Situation Type you add to a Situation is in use in some Inspection Policy, the new Situation is automatically included in the policy when you save the Situation, and the engines start matching traffic to the Situation when you refresh the policy.

Task 4: Associate the Situation with a Vulnerability

Vulnerabilities provide a short description of the event that has matched. Vulnerability information is included in dynamic update packages, so all Situations provided by McAfee that are related to a known vulnerability are linked to a Vulnerability element. When you create your own Situations, you can associate them with an existing Vulnerability or a custom Vulnerability element.

You can add up to four references to public vulnerability databases to your custom Vulnerabilities (CVE/BID/MS/TA). System vulnerabilities can have an unlimited number of references to any reference system, and can have multiple references to the same reference system. The reference information is also shown in the Logs view.

Using Situations

Situations are used for defining what you want to detect with the Inspection Policy. Situations are generally used for:

- Detecting malicious patterns in traffic. The Situations supplied by McAfee in dynamic update packages concentrate on such known vulnerabilities and exploits.
- Reducing the number of alert and log entries you receive (using Correlation Situations).
- Detecting some other traffic patterns that you want to record. For example, you may be interested in the use of certain applications.

Although the general workflow requires ensuring that a Situation you want to use is included in the Inspection Policy, you may often not actually insert the Situation into the rule, but use a Tag or Situation Type element instead to represent a whole group of Situations.

Examples of Custom Situations

The examples in this section illustrate some common uses for Situations and the general steps on how each scenario is configured.

Detecting the Use of Forbidden Software

Company A has an IPS engine deployed in between their internal network and the Internet. The IPS engine uses a policy that is based on the IPS Template policy.

The administrators find out that some of the internal users have installed a piece of software on their computers that the company's security policy forbids. They consider this software a security risk.

The administrators decide that they would like to detect the use of the software so that they can find out which users have installed it. The administrators find one simple but distinctive characteristic in the software: when launched, the software in question always connects to a particular address to check for updates using HTTP. The administrators:

1. Create a new custom Situation element with the name "Software X".
2. Add the **HTTP Client Stream** Context to the Situation and type in a regular expression that contains the address they want the Situation to find using the SMC regular expression syntax (see Regular Expression Syntax (page 235)).
3. Add one of the default Situation Types under **Traffic Identification** to the Situation.
4. Select the correct options for logging the traffic in the Rules tree in the Inspection Policy and install the policy on the IPS engine.
5. Open the Logs view and filter the view using the "Software X" Situation as the filtering criteria.
6. See which computers use the forbidden software and take action based on which IP addresses are shown in the logs.

Counting Events

Company B has a Firewall and an IPS engine that monitor traffic to a DMZ network. The DMZ contains a server that provides information to Company B's partners. A while ago, users started complaining that the service had slowed down.

Upon investigation, Company B's administrators found out that the traffic had grown dramatically even though the number of users and the data available had stayed the same. They found out that one of the partners had made a misconfigured script that frequently copied several large catalogs from Company B's server to their own server and had given the script to a few other partners as well. As a first step, the administrators decide to immediately stop excessive queries to the server. The administrators:

1. Create a custom Situation for detecting access to the catalog files.
2. Create a custom Correlation Situation, attach the **Count** Context to it, and define the settings for the Count Context to detect when there are more than 5 requests per minute to any of the files from the same source address.

Table 11.2 Context Settings for the Example Correlation Situation

Field	Option
Correlated Situations	Custom Situation
Time Window	60
Alarm Threshold	5
Log Fields Enabled	Select
Log Names	Src Addr

3. Insert the Correlation Situation in the Inspection Policy with blacklisting as the Action. The traffic from the offending hosts will be stopped at the Firewall.
4. Refresh the Inspection Policy on the IPS engine.

Preventing Access to Forbidden Websites

The Administrators at Company C have noticed that employees frequently visit certain websites that are not related to their work. They want to block access to these websites to prevent employees from accessing them at work. To do this, they:

1. Create a new Situation element.
2. Add the **Website Access Control** Context to the Situation.
3. Specify the addresses they want to prevent access to. Access to the specified addresses will be blocked.
4. Refresh the Inspection Policy on the IPS engine.

CHAPTER 12

IPS AND LAYER 2 FIREWALL POLICIES

IPS and Layer 2 Firewall Policy elements are containers for the lists of rules that determine how the IPS and Layer 2 Firewall engines inspect traffic. The Policy elements include Template Policies, Policies, and Sub-Policies.

The following sections are included:

Overview of IPS and Layer 2 Firewall Policies

IPS and Layer 2 Firewall Policy elements store rules according to which the IPS and Layer 2 Firewall engines examine the traffic. This chapter introduces you to how the IPS and Layer 2 Firewall engines use these elements. It also explains how you can build a purposeful and efficient policy hierarchy using the different policy elements. The basics of building the actual traffic handling rules that are contained in the policy elements are discussed in the chapters that follow. See Ethernet Rules (page 129), Access Rules (page 137), and Inspection Policies (page 151).

Policy Hierarchy

The policy structure in IPS and Layer 2 Firewalls is a hierarchical structure based on templates, which allows you to:

- Reuse rules without duplicating them.
- Assign and enforce editing rights of different parts of a single policy to different administrators.
- Reduce the resource consumption of the engines.
- Make policies easier to read.

IPS engines and Layer 2 Firewalls have separate Template Policies, Policies, and Sub-Policies.

The template and policy hierarchy is flattened when the Policy is transferred to the engines, so the policy will look the same to the engines regardless of how it is organized on the Management Server (as long as the rules are in the same order). You can also create sections of conditional IPv4 Access rules that you can insert into the other policy elements. The engine may skip the processing of a conditional block of rules based on whether or not certain common matching criteria is found in the packet being examined.

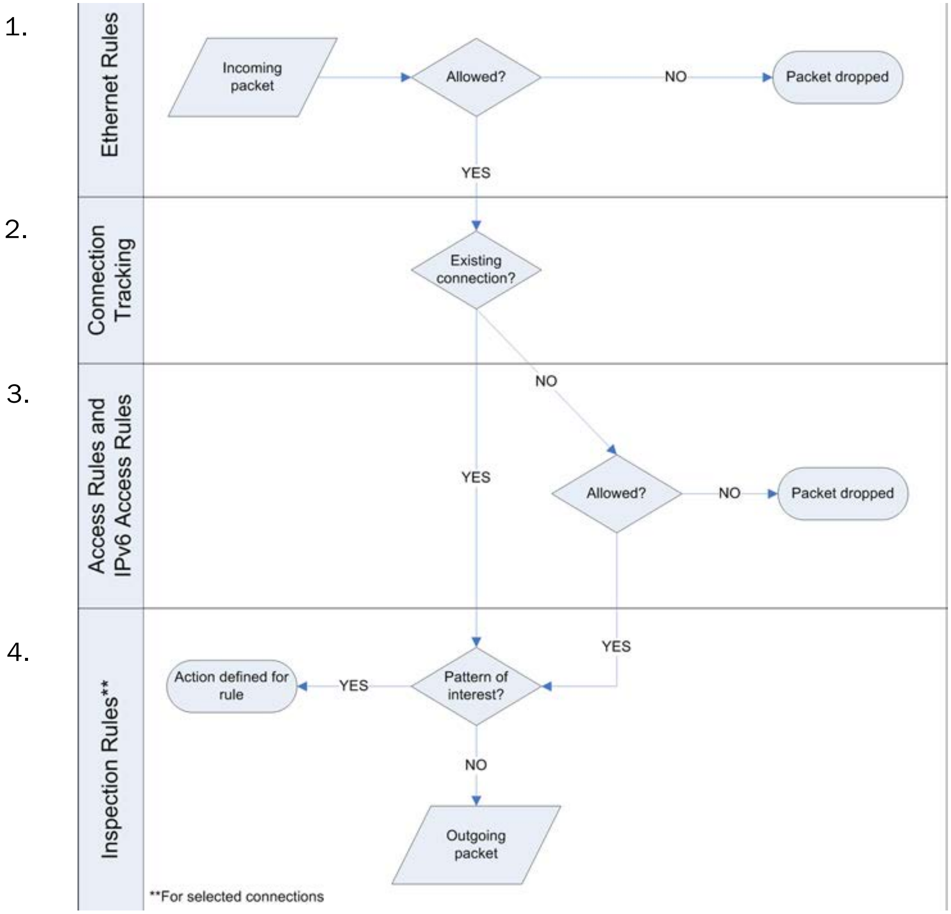
If your environment is simple and you do not need the benefits outlined above, you have the option of creating a very simple policy hierarchy. You can start, for example, with a single custom IPS Policy or Layer 2 Firewall Policy built on one of the pre-defined IPS Template Policies or Layer 2 Firewall Template Policies. You can use the same IPS Policy on any number of IPS engines and Virtual IPS engines, and the same Layer 2 Firewall Policy on any number of Layer 2 Firewalls and Virtual Layer 2 Firewalls.

How IPS Engines and Layer 2 Firewalls Examine Traffic

An IPS or Layer 2 Firewall engine matches traffic to different protocols and then checks the definitions for known vulnerabilities and other threats for that protocol. The protocol is assigned first, before the deep inspection. An engine in inline mode can also filter traffic based on protocols, IP addresses, and the interface that received the traffic without analyzing the traffic for threat patterns. IPS engines and Layer 2 Firewalls can be installed either in inline mode or in capture mode. See NGFW Deployment in IPS and Layer 2 Firewall Roles (page 29) for more information.

Illustration 12.1 shows how an engine in inline mode inspects traffic.

Illustration 12.1 Packet/Connection Handling in an inline IPS or Layer 2 Firewall Engine



1. The engine checks Ethernet frames against the Ethernet rules in the policy. The packet is processed until it matches an Ethernet rule that tells whether to allow or to discard the packet.
2. The engine checks the current connection tracking information to see if the packet is part of an established connection (for example, a reply packet to a request that has been allowed).
3. If the packet is not part of an existing connection, the packet is matched to IPv4 or IPv6 Access rules according to the IP protocol used.
   - If the traffic is tunneled using IP-in-IP or Generic Routing Encapsulation (GRE), the traffic can be checked against the IPv4 and/or IPv6 Access rules several times according to the number and type of layers in the tunnel and the settings of the engine.
   - The processing of the packet continues until the packet matches a rule that tells the engine to allow or discard the packet. If the packet does not match any Access rule, the

final action depends on the engine type. An IPS engine allows the packet to pass through, while a Layer 2 Firewall drops the packet.

4. The engine matches connections that are selected for deep packet inspection in the IPv4 or IPv6 Access rules against the Inspection rules.
   - The Inspection rules are used to look for patterns of interest in allowed connections. The patterns may indicate potential attacks, exploits, or other possible threats. Alternatively, they can be any other patterns that might be of interest, such as multiple login attempts, use of peer-to-peer or instant messaging software, or protocol violations in the traffic.
   - If a pattern in traffic matches a pattern defined in a rule, the action(s) defined in the rule are taken.

Configuration of Policy Elements

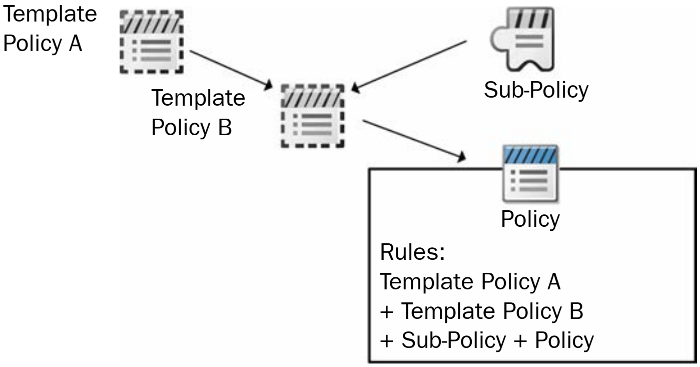
IPS engines and Virtual IPS engines use IPS Policies, and Layer 2 Firewalls and Virtual Layer 2 Firewalls use Layer 2 Firewall Policies. Master Engines always use Firewall Policies regardless of the role of the Virtual Security Engines they host.

There are four kinds of IPS and Layer 2 Firewall Policy elements:

- A *Template Policy* is a policy that can be used as the basis for Policy or other Template Policy elements. The rules in the Template Policy are copied as *inherited rules* into the Policies and Template Policies that are based on the Template Policy. You can modify the inherited rules only by editing the original Template Policy from which the rules were inherited.
- An *Inspection Policy* element is a set of Inspection rules that are referenced from the Inspection tab of Policy and Template Policy elements. You can use the same Inspection Policy in multiple Policy and Template Policy elements.
- A *Sub-Policy* element is a section of IPv4 Access rules that you can insert into Policies and Template Policies. The rules in the Sub-Policy are conditional rules that allow you to define matching criteria that determines whether the Sub-Policy applies to a connection. You can modify the rules by editing the Sub-Policy where the rules belong.
- A *Policy* element is an element that gathers together all the rules from the different policy elements (the rules added directly to the IPS Policy or Layer 2 Firewall Policy, the rules from the higher-level Template Policy, and possibly rules from one or more Sub-Policies). An IPS Policy is always based on an IPS Template Policy element. A Layer 2 Firewall Policy is always based on a Layer 2 Firewall Template Policy element. IPS Policies and Layer 2 Firewall Policies are the only type of policy elements that can be installed on engines.

The hierarchy of how rules are inherited between different policy elements is shown in Illustration 12.2.

Illustration 12.2 Rule Inheritance



In Illustration 12.2, Template Policy A is the basis for Template Policy B, so Template Policy B contains all the rules defined in Template Policy A. Template Policy B also contains all the rules in an Sub-Policy, as well as rules defined directly in Template Policy B. The example Policy inherits the following rules:

- All the rules in Template Policy A.
- All the rules in Template Policy B.
- All the rules in the Sub-Policy.

Inherited rules cannot be edited in the policy that inherits the rules. For example, to change rules inherited from Template Policy A, administrators must have privileges to edit the Template Policy A in which the rules were originally defined.

A hierarchy such as the one outlined above is useful to:

- Reduce the need for creating the same or similar rule in several policies. For example, any rule added to Template Policy A is also added to any policy created based on Template Policy A. The next time Policies based on Template Policy A are installed on engines, the new rule is used on all the engines without the need to directly modify each individual Policy.
- Restrict the editing rights of administrators. For example, administrators who are granted rights only to Policy elements cannot edit the rules inherited from the Policy Templates. Their actions have no effect on rules that are placed above the row where the Template Policy allows them to insert new rules. In the hierarchy shown in the illustration above, the insert point(s) for the Policy are defined in Template Policy B, which in turn can be edited only in the place where there is an insert point in Template Policy A.
- Reduce the likelihood of mistakes affecting important communications. Template Policies can be reserved for defining only the rules for essential communications, so that most daily editing is done in the lower-level Policies. If the Template Policy is properly designed, the rules in the Template Policy cannot be overridden by any rules in the lower-level policy. Good organization also makes policies easier to read, further reducing the risk of errors.
- Improve processing performance. With the help of Sub-Policies, whole blocks of rules may be skipped during processing when a connection does not match the rule that directs the traffic processing to the Sub-Policy. This reduces the processor load, which may lead to improved throughput if the processor load is constantly very high.

Default Elements

The default policy elements are introduced when you import and activate a recent dynamic update package (for example, during the installation). The elements may change when you install newer update packages.

None of the default policy elements can be modified. However, you can make copies of the default policies if you need to create a modified version.

The following table describes the default policy elements for IPS and Layer 2 Firewall engines.

Table 12.1 Default Policy Elements for IPS and Layer 2 Firewall Engines

Element Type	Default Element Name	Description
IPS Template Policy	IPS Template	A Template Policy that contains the predefined Access rules necessary for the IPS engine to communicate with the SMC and some external components. The predefined Access rules are explained in Access Rules (page 137). The IPS Template Policy uses Inspection rules from the High-Security Inspection Policy. The IPS Template Policy provides an easy starting point for determining what kinds of rules your system needs.
IPS Policy	Customized High-Security Inspection IPS Policy	An IPS Policy that is based on the IPS Template. The Customized High-Security Inspection IPS Policy contains a set of customized rules that were used when the IPS was tested at ICSA Labs and NSS Labs.
	Default IPS Policy	An IPS Policy that is based on the IPS Template. The Default IPS Policy does not add any rules to those defined in the IPS Template. It allows you to install the predefined rules in the IPS Template on the IPS engine right after installation (since Template Policies cannot be installed on the engines).
Layer 2 Firewall Template Policy	Layer 2 Firewall Template	A Template Policy that contains the predefined Access rules necessary for the Layer 2 Firewall to communicate with the SMC and some external components. The predefined Access rules are explained in Access Rules (page 137). The Layer 2 Firewall Template uses Inspection rules from the No Inspection Policy. The rules in the No Inspection Policy do not enforce inspection.
	Layer 2 Firewall Inspection Template	A Template Policy that is based on the Layer 2 Firewall Template. It uses Inspection rules from the High-Security Inspection Policy. The Layer 2 Firewall Inspection Template enables deep inspection for all traffic.

Table 12.1 Default Policy Elements for IPS and Layer 2 Firewall Engines (Continued)

Element Type	Default Element Name	Description
Inspection Policy	No Inspection Policy	An Inspection Policy with a set of Inspection rules that do not enforce inspection.
	Medium-Security Inspection Policy	An Inspection Policy with a set of Inspection rules for detecting common threats. The Medium-Security Inspection Policy logs Situations categorized as Suspected Attacks but allows the traffic to pass. The Medium-Security Inspection Policy is suitable for Firewall, Layer 2 Firewall, Master Engine, and Virtual Security Engine deployments. It is also suitable for inline IPS deployments in asymmetrically-routed networks and IPS deployments in capture mode. The risk of false positives is low in production use.
Inspection Policy (*cont.*)	High-Security Inspection Policy	An Inspection Policy with a set of Inspection rules for detecting common threats. The High-Security Inspection Policy terminates Suspected Attacks with an alert. The High-Security Inspection Policy is suitable for Firewall, Layer 2 Firewall, inline IPS, Master Engine, and Virtual Security Engine deployments where extended inspection coverage and strong evasion protection is required. The risk of false positives is moderate in production use. The High-Security Inspection Policy terminates a connection if the engine cannot see the whole connection. We recommended that you use the High-Security Inspection Policy as a starting point for your Inspection Policies.
	Customized High-Security Inspection Policy	An Inspection Policy that is based on the High-Security Inspection Policy and contains a set of customized Inspection rules. The High-Security Inspection Policy is an example of a highly customized Inspection Policy for network environments in which unconditional inspection coverage and evasion protection are required. The risk of false positives is high in production use. The High-Security Inspection Policy was used when the IPS was tested at ICSA Labs and NSS Labs. It provides an example of a customized Inspection Policy.

You can add new Template Policies without basing them on any existing Template Policy. However, in most cases we recommend using the IPS Template as the starting point for your customized IPS Template Policies and IPS Policies, and the Layer 2 Firewall Template as the starting point for your customized Layer 2 Firewall Template Policies and Layer 2 Firewall Policies.

Situations are the central elements in the Inspection rules of your Inspection Policies. The Situation elements detect exploit attempts against known vulnerabilities and other commonly known security threats. Because dynamic updates include new and updated Situations, new patterns in traffic may begin to match when a new dynamic update is activated and you refresh the Inspection Policy.

In most environments we recommend using the High-Security Inspection Policy as the starting point for Inspection Policies. The High-Security Inspection Policy provides extended inspection coverage. It also protects the network against evasions, which are attempts to disguise attacks in order to avoid detection and blocking by network security systems. The only difference between the rules in the High-Security Inspection Policy and the Medium-Security Inspection Policy is in the way the Inspection rules handle Situations that are categorized as Suspected Attacks. The High-Security Inspection Policy terminates Suspected Attacks with an alert, whereas the Medium-Security Inspection Policy only logs Suspected Attacks.

Situations that belong to the Suspected Attacks category contain basic fingerprints. Suspected Attacks also contain traffic patterns that may indicate malicious activities but are not any verified attack patterns. Suspected Attacks can catch zero-day attacks (attacks that are not yet publicly known), but may sometimes block some legitimate traffic if the traffic pattern happens to resemble malicious activities.

Configuration Workflow

The following sections provide an overview of the configuration tasks. Detailed step-by-step instructions can be found in the Management Client *Online Help* and the *McAfee SMC Administrator's Guide*.

Policy elements are merely containers for the actual traffic handling rules. When you have decided on a policy hierarchy, you can populate the policy elements with the actual rules for handling the traffic. See Ethernet Rules (page 129), Access Rules (page 137), and Inspection Policies (page 151).

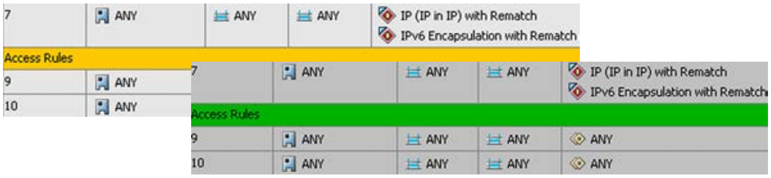
Task 1: Create a Template Policy

IPS Template Policy elements are used as a basis for other IPS Policies and IPS Template Policies. *Layer 2 Firewall Template Policy* elements are used as a basis for Layer 2 Firewall Policies and other Layer 2 Firewall Template Policies.

Every IPS Policy and Layer 2 Firewall Policy that you create is based on a Template Policy. You can also base several policies on the same Template Policy. Although it is possible to create custom Template Policies without basing them on any of the pre-defined templates, we recommend basing your IPS Policies on the IPS Template and your Layer 2 Firewall Policies on the Layer 2 Firewall Template. Updated versions of the Templates are included in dynamic update packages, so the policies that inherit rules from the pre-defined Templates are automatically updated when you activate a new dynamic update. Policies based on a copy of the pre-defined templates or a completely different Template are not automatically updated.

When editing the policies, the main difference between Template Policies and Policies are the special rows called *Insert Points*. Insert points are shown in both Template Policies and Policies, but you can add them only to Template Policies. The insert points added to templates mark the places where new rules can be added to Policies that are based on the templates. If you create a new Template Policy and do not base the Template Policy on any pre-defined Template Policy, you must add insert points on the Ethernet and Access tabs to be able to add rules on each tab.

Illustration 12.3 Insert Point in a Template and the Inheriting (Template) Policy



7	ANY	ANY	ANY	IP (IP in IP) with Rematch IPv6 Encapsulation with Rematch
Access Rules				
9	ANY			
10	ANY			

7	ANY	ANY	ANY	IP (IP in IP) with Rematch IPv6 Encapsulation with Rematch
Access Rules				
9	ANY	ANY	ANY	ANY
10	ANY	ANY	ANY	ANY

Illustration 12.3 shows an example of what the same insert point looks like in an IPS Template Policy and in the inheriting IPS Policy elements. The color of the insert point indicates whether the insert point has been added in the IPS Template Policy for inheritance to lower levels (orange) or whether it has been inherited from the higher-level template (green). Only the orange insert points are inherited to lower-level policy elements, so you must add at least one new insert point to each template you create to make the lower-level policies editable. When you add the first new rule to the green insert point, the rule replaces the insert point. Any number of rules can then be added directly above and below that first rule.

Rules defined in the Template Policy itself are not editable in lower-level policies that use the template. Such inherited rules are shown only on your request and they have a gray background. Only the actual rules are inherited from a higher-level template into the lower-level policies and templates. The rights to edit policies and Template Policies are defined separately.

Because the IPS and Layer 2 Firewall engines read rules in order from top down, rules above the insert point in the higher-level template cannot be cancelled by anything a lower-level policy adds below the insert point.

Task 2: Create a Policy

An IPS Policy or a Layer 2 Firewall Policy is the element that gathers together all the rules from the different policy elements: the rules inherited from the Template Policy that is used as the basis of the policy, rules from one or more Sub-Policies added to the policy, rules added directly to the policy, and the rules from the Inspection Policy that is referenced from the Inspection tab in the policy.

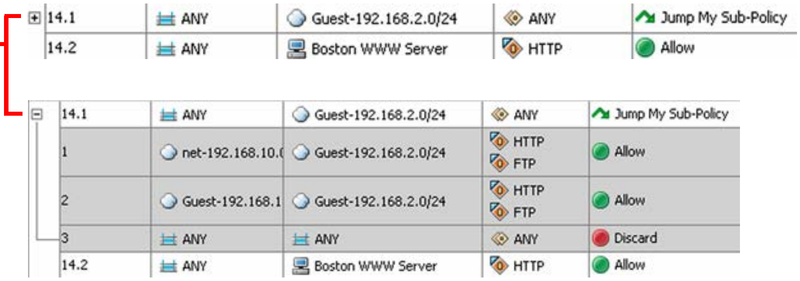
Task 3: Create a Sub-Policy

Sub-Policies are sections of IPv4 Access rules that you can insert into IPS Policies, IPS Template Policies, Layer 2 Firewall Policies, Layer 2 Firewall Template Policies, and even other IPS Sub-Policies or Layer 2 Firewall policies. The Sub-Policies are not based on any template. Apart from IPv4 rules, you cannot insert any other types of rules into IPS Sub-Policies or Layer 2 Firewall Policies.

Using Sub-Policies can make the engines process traffic faster. You can also use Sub-Policies to organize rules. Sub-Policies may make reading the policies and assigning administrator editing rights easier. For example, you can give some administrators the rights to edit only certain IPS Sub-Policies or Layer 2 Firewall Sub-Policies without giving editing rights to IPS Policies or Layer 2 Firewall Policies.

A Sub-Policy is inserted into some other policy element by adding a *Jump rule* to the policy element. The Jump rule directs connections that match the criteria defined in the Jump rule for inspection against the Sub-Policy.

Illustration 12.4 An Example of an IPS Sub-Policy in Use



A Jump rule inserts the Sub-Policy, which is shown as an expandable section.

14.1	ANY	Guest-192.168.2.0/24	ANY	Jump My Sub-Policy
14.2	ANY	Boston WWW Server	HTTP	Allow

14.1	ANY	Guest-192.168.2.0/24	ANY	Jump My Sub-Policy
1	net-192.168.10.(	Guest-192.168.2.0/24	HTTP FTP	Allow
2	Guest-192.168.1	Guest-192.168.2.0/24	HTTP FTP	Allow
3	ANY	ANY	ANY	Discard
14.2	ANY	Boston WWW Server	HTTP	Allow

The illustration above shows the same Jump rule in a policy in the collapsed and the expanded state. The rules of the Sub-Policy are shown on a gray background, because they can be edited only within the Sub-Policy itself, not in the Policy that uses the rules.

For example, you could create a Sub-Policy for checking traffic destined to a group of servers located in one particular network. The Jump rule could then use the destination network as a criteria for directing connections for checking against the Sub-Policy. Any connection that was destined to some other network would not get matched against any of the rules in the Sub-Policy. This makes the Access rule processing faster, because the engine can skip a whole Sub-Policy by comparing a connection to just one simple rule for any non-matching connection. If the Sub-Policy rules were inserted directly into the main Policy, the engine would have to compare all connections to all those rules individually. Naturally, the performance benefit gained depends on the number and complexity of the rules that can be placed in a Sub-Policy and how heavily stressed the engine is to begin with. Therefore, Sub-Policies are mainly useful in the policies of inline IPS or Layer 2 Firewall engines that are used extensively for IPv4 packet filtering.

Task 4: Install the Policy

After creating or modifying an IPS Policy or Layer 2 Firewall Policy, you must transfer the changes to the engines using the Management Client.

The policy transfer includes more information than just the rules in the IPS Policy or Layer 2 Firewall Policy. Whenever you update the engine's configuration, you must refresh the IPS Policy or the Layer 2 Firewall Policy to make the changes take effect. After you have modified Physical Interfaces or VLAN Interfaces in the Master Engine properties, you must refresh the policy on the Master Engine to transfer changes. After you have modified Physical Interfaces or VLAN Interfaces in the Virtual IPS or Virtual Layer 2 Firewall engine properties, you must refresh the policy on the Virtual IPS or Virtual Layer 2 Firewall engine to transfer the changes.

If the installation of a policy fails for some reason, the system automatically rolls back to the previously installed configuration.

Using Policy Elements and Rules

The main points of using policy elements are explained in the preceding sections of this chapter. The sections below illustrate additional points that are useful to know when working with policies and rules:

- Validating the Policy
- User Responses
- Connection Tracking vs. Connectionless Packet Inspection (page 124)
- Policy Snapshots (page 126)
- Continue Rules (page 126)
- Adding Comments to Rules (page 126)
- Naming Rules (page 127)

Validating the Policy

The number of rules in a policy may grow quite large over time. It may become quite difficult, for example, to notice configuration errors in a policy. To make policy management easier and make sure that the policy does not contain misconfigured rules, you can automatically validate the policy while editing and during the policy installation. There are different criteria for validating the policy. You can, for example, check the policy for duplicate and empty rules or check if there are rules that cannot ever match traffic.

User Responses

The User Response element allows you to send a configurable reply to the client instead of just ending the TCP connection when an HTTP connection is terminated or blacklisted. The reply can be a custom error message, or an HTTP redirect to a specified URL. The User Response is selected in the Action options in Access rules and in the Exceptions in the Inspection Policy.

Connection Tracking vs. Connectionless Packet Inspection

Connection tracking means that the engine keeps a record of all currently open connections (stateful inspection). With connection tracking, the engine can verify that the connection proceeds according to the protocol standards. Connection tracking is also required for enforcing some other connection-related settings. By default, connection tracking is on.

However, it is not necessary to keep track of the state of certain kinds of connections (for example, SNMP traps). Some applications may also communicate in a non-standard way that prevents them from being allowed through the engine when connection tracking is used. For those connections, you can disable connection tracking in the Access rules. This allows the engine to function as a simple packet filter for those connections. This also prevents the use of some features that require connection tracking.

When connection tracking is off, each packet that you want to allow must match an Access rule that allows the packet. This means that even if two-way communications are always opened one way, both directions of communication must be explicitly allowed in Access rules. Reply packets are not recognized, so they are not automatically allowed through. If done carelessly, turning connection tracking off reduces the benefits you gain from your engine and may even weaken security. You may have other options: in some cases using the correct Protocol Agent helps.

Note – Before disabling connection tracking, always check if there is a Protocol Agent for the protocol in question. The Protocol Agents can pass connections that require special handling when connection tracking is on, which is always a more secure option.

When connection tracking is enabled in an Access rule, the Service cell of the rule must contain one of the protocols supported for connection tracking (ICMP, TCP, UDP, or another protocol that belongs to the IP protocol suite). ICMP and UDP are stateless protocols that do not contain any connection data. However, ICMP and UDP packets contain data that the engine can use to build virtual connections. The engine can also build virtual connections based on the IP address and IP protocol data in other types of IP packets.

You can choose between several connection tracking modes, depending on the traffic type and how strictly you want the connections to be tracked. The effect of the connection tracking mode set in the Access rules depends on the traffic type. The available options are explained in Table 12.2.

Table 12.2 Connection Tracking Modes in Access Rules

Option	Explanation
Inherited from Continue Rule(s)	The connection tracking setting defined in Continue rule(s) higher up in the policy is used. The additional options available for connection tracking are explained in the next table. **Note!** If connection tracking is disabled in Continue rule(s) higher up in the policy, the IPS or Layer 2 Firewall engine behaves as described in the Off (Not recommended) explanation below.

Table 12.2 Connection Tracking Modes in Access Rules (Continued)

Option	Explanation
Off (Not recommended)	Connection tracking is disabled. The IPS or Layer 2 Firewall engine operates as a simple packet filter and allows packets based on their source, destination, and port. You must add separate Access rules that explicitly allow the reply packets. NAT cannot be applied to traffic allowed without connection tracking. **Note!** Turn off logging in the rule if you disable connection tracking. When connection tracking is off, a log entry is generated for each packet. This may put considerable strain on the engine, network, and the Log Server.
Defined in Engine Properties	The IPS or Layer 2 Firewall engine allows or discards packets according to the Connection Tracking mode selected in the engine properties. Protocols that use a dynamic port assignment must be allowed using a Service with the appropriate Protocol Agent for that protocol (in Access rules and NAT rules).
Normal	The Normal mode is the default Connection Tracking mode for Layer 2 Firewalls. The IPS or Layer 2 Firewall engine drops ICMP error messages related to connections that are not currently active in connection tracking (unless explicitly allowed by a rule in the policy). A valid, complete TCP handshake is required for TCP traffic. The IPS or Layer 2 Firewall engine checks the traffic direction and the port parameters of UDP traffic. If the Service cell in the rule contains a Service that uses a Protocol Agent, the IPS or Layer 2 Firewall engine also validates TCP and UDP traffic on the application layer. If a protocol violation occurs, the packet that violates the protocol is dropped.
Strict	The IPS or Layer 2 Firewall engine allows only TCP traffic that strictly adheres to the TCP standard as defined in RFC 793. The IPS or Layer 2 Firewall engine also checks the sequence numbers of the packets in pre-connection establishment states and for RST and FIN packets, and drops packets that are out of sequence. If the Service cell in the rule contains a Service that uses a Protocol Agent, the IPS or Layer 2 Firewall engine also validates the traffic on the application layer. If a protocol violation occurs, the packet that violates the protocol is dropped.
Loose	The Loose mode is the default Connection Tracking mode for IPS engines. Allows some connection patterns that are not allowed in the Normal mode. Can be used, for example, if routing is asymmetric and cannot be corrected or if the use of dynamic routing protocols causes the IPS or Layer 2 Firewall engine to receive non-standard traffic patterns. This is the recommended connection tracking mode for Layer 2 Firewalls, IPS engines, Virtual Layer 2 Firewalls, and Virtual IPS engines when they are configured by default to only log connections instead of terminating them. See the *McAfee SMC Administrator's Guide* for more information on the Default Connection Termination settings for each engine type.

If Connection Tracking is on, you can also set the **Idle Timeout** for connections. The timeout is meant for clearing the engine's records of old connections that the communicating hosts leave hanging. The timeout concerns only idle connections, so connections are not cut because of timeouts while the hosts are still communicating. The timeout defined for an Access rule overrides the default idle timeout value that is set for the protocol in the engine's properties.

Caution – Setting excessively long timeouts for a large number of connections may consume engine resources and degrade engine performance and stability. Be especially careful when defining timeouts for ICMP and UDP. The ICMP and UDP virtual connections do not have closing packets, so the engine keeps the records for the ICMP and UDP connections until the end of the timeout.

Connection tracking options in Access rules also allow you to override the limit for concurrent connections from a single source and/or destination IP address defined on the Advanced tab in the Security Engine properties, in Virtual Resource properties, and in the properties of some interface types. When the set number of connections is reached, the next connection attempts are blocked by the engine until a previously open connection is closed.

Changes in the connection tracking mode affect how existing connections are handled when you install or refresh the policy. When you install or refresh the policy on an engine, the engine checks if the existing connections are still allowed in the policy. If the connection tracking mode changes from Loose to Strict, existing virtual ICMP connections are only allowed if they began with a valid packet (for example, not with a response packet). In addition, if the mode changes from Normal to Strict, existing TCP connections are only allowed if all the packets in the connection have been seen. In all other cases, changes in connection tracking mode do not affect existing ICMP, TCP, and UDP connections at policy installation.

Policy Snapshots

A *Policy Snapshot* is a stored record of a policy configuration. A Policy Snapshot is stored in an engine's upload history whenever a policy is installed or refreshed on the engine. The Policy Snapshots allow you to check which IPS Policies and other configuration information were uploaded, and when they were uploaded. You can also compare any two Policy Snapshots and see the differences between them.

Continue Rules

The Continue action for a rule sets default options (such as logging options) for the traffic matching process. Options set in Continue rules are used for subsequent rules that match the same criteria as the Continue rule, unless the rules are specifically set to override the options. Continue rules are also very useful in the hierarchical structure of the policies. IPS Template Policies and Layer 2 Firewall Template Policies are particularly convenient for setting options with a Continue rule, because all the Policies and Template Policies that use the template inherit the option settings you have specified. Continue rules are explained in detail in Configuring Default Settings for Several Rules (page 146).

Adding Comments to Rules

Each policy can include a large number of rules. Adding comments provides administrators with useful information and also makes it easier to read policies. You can add comments to all types of rules. In rule tables, you can add comments in the rule's Comment cell. You can also a Rule Section, which begins with a comment row and can include one or more rules.

The Rule Section automatically includes all the rules below the Rule Section until the next Rule Section in the policy. You can expand and collapse the Rule Sections as necessary. The comment row in a Rule Section is displayed against a colored background (with configurable colors). This makes Rule Sections particularly useful in visually separating the sections of rules within a single policy.

Naming Rules

In addition to comments, it is possible to specify an optional name or short description for Access rules and Ethernet rules in IPS and Layer 2 Firewall Policies, Exceptions in Inspection Policies, and rules in QoS Policies and Alert Policies. Names help administrators identify individual rules in large rule tables. You can also search for a rule by its name. If a rule has been named, the name is displayed in the Logs view as well.

Example of Policy Element Use

This section illustrates a common use for the different policy elements and general steps on how the scenario is configured.

Restricting Administrator Editing Rights

Company A is implementing a distributed network with multiple sites, one central office where most of the administrators work, and a number of branch offices in different countries. The branch offices mostly have IT staff with only limited networking experience, but who are still responsible for the day-to-day maintenance of the network infrastructure and the IPS engines at their site. They must be able to, for example, add and remove Access rules for testing purposes without always contacting the main administrators.

The administrators decide to limit the privileges of the branch office IT staff so that they are not able to edit the policies of the IPS engines at any of the other sites. The administrators:

1. Create a new IPS Template Policy based on the pre-defined IPS Template.
2. Add rules to the IPS Template Policy using Alias elements to cover the essential services that each of these sites have.
   - Using a common IPS Template Policy for all the branch offices eliminates the need to make the same changes in several policies, easing the workload.
3. Create a new IPS Policy based on the new template for each of the branch office sites.
   - Although a single IPS Policy for all sites could work, in this case the administrators decide against it, as separate policies are needed for the separation of editing rights. The policies are based on the same template, so rules can still be shared without duplicating them manually.
4. Grant each IPS Policy to the correct IPS engine elements.
   - After this, only the correct IPS Policy can be installed on each IPS engine. No other policy is accepted.
5. Create new accounts with restricted rights for the branch office administrators and grant the correct IPS engine element and IPS Policy to each administrator.
   - The branch office administrators are now restricted to editing one IPS Policy and can install it on the correct IPS engine.
   - The branch office administrators are not allowed to edit the template the IPS Policy is based on, nor can they install any other policies on any other IPS engines.

Administrator rights are explained in more detail in the *McAfee SMC Reference Guide*.

Chapter 13

Ethernet Rules

Ethernet rules are lists of matching criteria and actions that define whether Ethernet protocol traffic is allowed or discarded.

The following sections are included:

Overview of Ethernet Rules

Ethernet rules are used by inline IPS engines or Layer 2 Firewalls in Transparent Access Control mode, by inline Virtual IPS engines, and by Virtual Layer 2 Firewalls. Ethernet rules define which Ethernet protocol packets the engines stop, and which packets are allowed through. Ethernet rules can also be used by IPS engines, Layer 2 Firewalls, Virtual IPS engines, and Virtual Layer 2 Firewalls in capture mode to define how Ethernet protocol traffic is logged. The Ethernet rules are stored in policy elements, which are discussed in IPS and Layer 2 Firewall Policies (page 113).

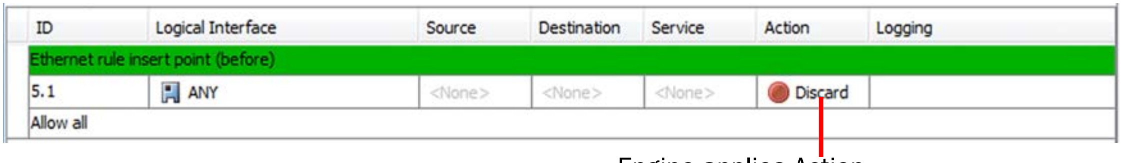
The traffic matching in Ethernet rules is based on the Source and Destination MAC Address in the packets. Any Ethernet network traffic, such as ARP, RARP, IPv6, Cisco Discovery Protocol (CDP), and Spanning Tree Protocol (STP), can be checked against the Ethernet rules. Ethernet traffic can be allowed or discarded. For IPS engines or Layer 2 Firewalls in capture mode, only the Allow action is available.

Regardless of the action taken, a matching rule can also create a log or alert entry.

Configuration of Ethernet Rules

Ethernet rules are configured on the **Ethernet** tab inside IPS and Layer 2 Firewall Policy and Template Policy elements. IPS and Layer 2 Firewall Sub-Policies cannot contain Ethernet rules.

Illustration 13.1 Newly Inserted Ethernet Rule - Main Cells



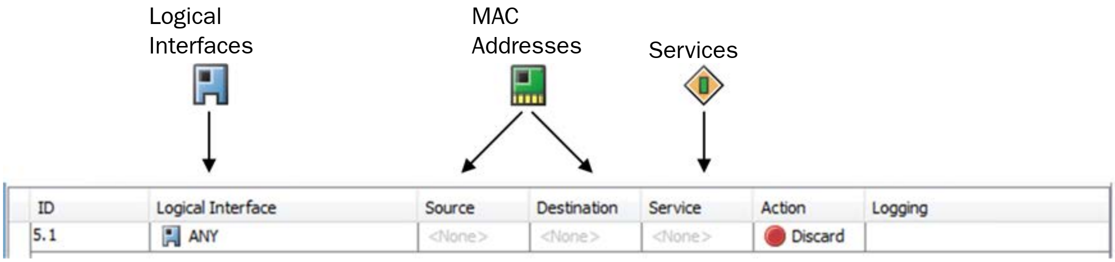
ID	Logical Interface	Source	Destination	Service	Action	Logging
Ethernet rule insert point (before)						
5.1	ANY	<None>	<None>	<None>	Discard	
Allow all						

Engine applies Action
when it finds a match

Illustration 13.1 displays an Ethernet rule that has just been inserted into the policy. The **Source**, **Destination**, and **Service** cells are set to **NONE**, so this rule never matches until they are changed to **ANY** or some more specific value. The **Logical Inteface** cell is also matched against traffic, but it is not mandatory to change it if you want the rule to apply regardless of the interface. The other editable cells specify further conditions and options for handling matching connections. It is not necessary to modify all cells in each rule.

Before starting to build policies, make sure you understand the network element types available and how you can use them efficiently to define the resources that you want to protect and control. The illustration below shows the types of elements that you can use in Ethernet rules.

Illustration 13.2 Elements in Ethernet Rules



ID	Logical Interface	Source	Destination	Service	Action	Logging
5.1	ANY	<None>	<None>	<None>	● Discard	

The table below explains briefly what each Ethernet rule cell does and which elements you can use in the rules. More information on each cell is included in the task flow later in this chapter.

Table 13.1 Ethernet Rule Cells

Cell	Explanation
ID	*(Not editable)* Automatically assigned ID number that indicates the order of the rules in the policy. The rules are matched against traffic in the order of the ID numbers.
Logical Interface	Matches the rule based on which interface the traffic is picked up from. The same logical interface may be assigned to one or several interfaces as configured in the properties of the IPS engine or Layer 2 Firewall. This cell accepts only Logical Interface elements.
Source	Elements containing the MAC addresses that the rule matches. Both the Source and the Destination defined must match the source and destination of a packet for the packet to match the rule. The Source and Destination cells accept MAC Address elements.
Destination	
Service	The Services match an Ethernet frame type. The Service cell accepts Ethernet Services elements.
Action	Command for the engine to carry out when a connection matches the rule.
Logging	The options for logging.
Comment	An optional free-form comment for this rule. You can also add separate comment rows in between rules.
Rule Name	Contains a rule tag and optionally a rule name. Name: (*Optional*) Name or description for the rule. Displayed alongside the rule tag. Tag: (*Not editable*) Automatically assigned unique identification for the rule. Works as a link between the log entries and the rule that has generated the log entries. The rule tag consists of two parts (for example, @20.1). The first part of the tag is permanent and belongs to only that rule. The second part changes when the rule is changed. The first part and the second part are separated by a period.

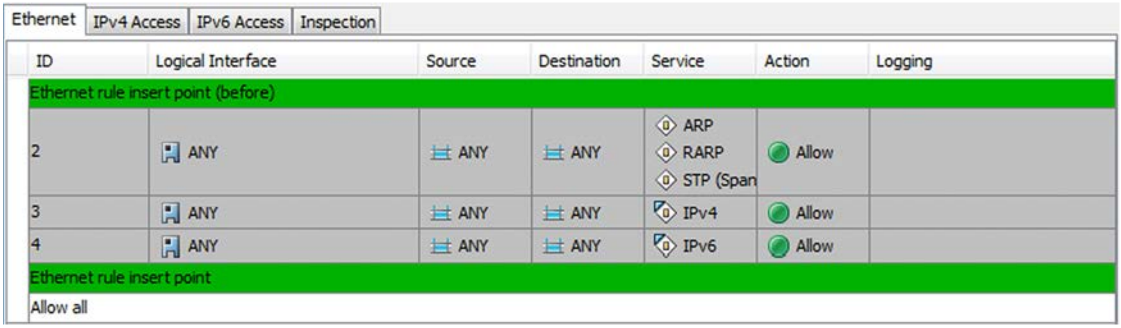
Considerations for Designing Ethernet Rules

Ethernet rules are read from the top down. More specific rules must be placed above more general rules that match the same traffic. The actions **Allow** and **Discard** stop the processing from continuing down the rule table for any packet that matches the rule. Rules with these actions must be placed so that the more limited the rule is in scope, the higher up in the rule table it is. If the traffic does not match any of the Ethernet rules by the end of the policy, it is allowed by default.

Default Elements

The IPS Template and the Layer 2 Firewall Template contain predefined Ethernet rules. Because the template policies are added and updated through dynamic update packages, the templates you currently have in your SMC may look slightly different from the one that is presented in this section. Newer versions of the templates work in the same way as described below. Any changes to the templates are documented in the Release Notes document for each dynamic update package. The predefined Ethernet rules allow the most common types of Ethernet traffic.

Illustration 13.3 IPS Template - Ethernet Rules



ID	Logical Interface	Source	Destination	Service	Action	Logging
Ethernet rule insert point (before)						
2	ANY	ANY	ANY	ARP RARP STP (Span	Allow	
3	ANY	ANY	ANY	IPv4	Allow	
4	ANY	ANY	ANY	IPv6	Allow	
Ethernet rule insert point						
Allow all						

In the illustration above, you can see a green insert point at the top of the rule table, three default rules below it, and then another insert point below them.

- The first rule contains common Ethernet protocols and allows the matching traffic to pass through.
- The second rule contains the IPv4 protocol and allows IPv4 traffic with further inspection against the IPv4 Access rules.
- The third rule contains the IPv6 protocol and allows IPv6 traffic with further inspection against the IPv6 Access rules.

The two insert points indicate where you can add Ethernet rules to a policy that uses the IPS Template or the Layer 2 Firewall Template. The first insert point above the default rules allows you to modify and make exceptions to how traffic that matches the three default rules is checked. For example, you could add a rule defining that no IPv4 or IPv6 traffic at all is allowed between certain MAC addresses.

The second insert point below the default rules allows you to define how traffic that matches other protocols is checked. The final rule allows all traffic.

Configuration Workflow

The following sections provide an overview to the configuration tasks. Detailed step-by-step instructions can be found in the Management Client *Online Help* and the *McAfee SMC Administrator's Guide*.

Task 1: Define the Source and Destination

The source and destination MAC addresses specified in a rule are compared to the MAC addresses in each packet's header. Based on these and other criteria, the rule is applied to matching packets. By default, these cells are set to NONE, and you must change the value in both cells to make the rule valid.

The Source and Destination cells accept MAC address elements. You can set these cells to ANY to make the rule match all possible source or destination MAC addresses. You can also add more than one element in each cell to make the rule match multiple MAC addresses.

Task 2: Define the Service

The Service cell defines which protocol(s) the rule you design applies to. By default, the Service is set to NONE, and you must change the value to make the rule valid. The Service cell accepts only Ethernet Services elements. You can set the Service to ANY to make the rule match all protocols.

Task 3: Select the Action

The Action cell defines what happens when a packet matches the rule. The following actions are available in the Ethernet rules:

- Allow: the traffic is let through the engine.
- Discard: the traffic is silently dropped if going through an Inline interface.

Note – When the Allow action is used for IPv4 or IPv6 traffic in the Ethernet rules, the traffic is then checked against the IPv4 Access rules or the IPv6 Access rules. The final action for the IPv4 and IPv6 traffic is determined according to traffic type by the IPv4 Access rules or the IPv6 Access rules.

Task 4: Select Logging Options

Rules can create a log or alert entry each time they match. By default, logging options set in a previous rule with Continue as its action are used. If no such rule exists, Layer 2 Firewalls and Virtual Layer 2 Firewalls log the connections by default. IPS engines and Virtual IPS engines do not log the connections by default. Each individual rule can be set to override the default values.

The log levels are explained in Table 13.2.

Table 13.2 Log Levels in Ethernet Rules

Log Level	Explanation
None	Does not create any log entry.
Transient	Creates a log entry that is displayed in the Current Events mode in the Logs view (if someone is viewing it at the moment) but is not stored.
Stored	Creates a log entry that is stored on the Log Server.
Essential	Creates a log entry that is shown in the Logs view and saved for further use.
Alert	Triggers an alert entry.

When the Log Server is unavailable, log entries are temporarily stored on the engine. When the engine is running out of space to store the log entries, it begins discarding log data in the order of importance. Monitoring data is discarded first, followed by log entries marked as Transient and Stored, and finally log entries marked as Essential. The Alert entries are the last log entries to be discarded.

The settings for storing the logs temporarily on the engine are defined in the engine's log spooling policy. For more information, see the *McAfee SMC Administrator's Guide*.

Note – A log entry is generated for each packet that matches an Ethernet rule. Use careful consideration when setting the logging options to avoid producing an excessive amount of log data.

Using Ethernet Rules

You can validate Ethernet rules and add comments to the rules just like for any other types of rules. See Using Policy Elements and Rules (page 123) for more information.

Examples of Ethernet Rules

The examples in this section illustrate some common modifications to the default Ethernet rules and general steps on how each scenario is configured.

Logging Ethernet Protocol Use

The administrators at Company A have installed an IPS engine in Transparent Access Control mode and they want to create some custom Ethernet rules. The administrators know that the majority of traffic uses the IPv4 protocol, but they do not have a clear idea of which other Ethernet protocols are being used in the company's network. They decide to temporarily log the use of Ethernet protocols, excluding IPv4.

To do this, the administrators:

1. Create a new IPS Policy based on the IPS Template.
2. Add a new rule in the Ethernet rules to exclude IPv4 traffic from logging:

Table 13.3 Ethernet Rule for Excluding IPv4 Traffic from Logging

Source	Destination	Service	Action	Options
ANY	ANY	IPv4	Allow	Logging: None

3. Add a rule to log the use of other Ethernet protocols:

Table 13.4 Ethernet Rule for Logging Ethernet Protocol Use

Source	Destination	Service	Action	Options
ANY	ANY	ANY	Allow	Logging: Stored

4. Save and install the policy on the IPS engine.
5. View the logs generated by the matches to the Ethernet rules in the Logs view.
6. Disable the logging Ethernet rule to prevent excess log data from being generated.

Restricting the Use of Ethernet Protocols

Now that the administrators at Company A from the previous example have a clear picture of which Ethernet protocols are being used in the company's network, they want to restrict which protocols are allowed. The administrators determine that ARP and Spanning Tree Protocol (STP) must be allowed. Since the majority of traffic will use these protocols, the administrators do not want to log matches to the rules that allow specific protocols.

They decide to block the Cisco Discovery Protocol (CDP) on the logical interface named Inline Interface because of the security problems it creates, and store log entries of detected CDP use.

To do this, the administrators:

1. Add a new rule in the Ethernet rules to allow ARP, Spanning Tree Protocol (STP), and IPv4 without producing any logs:

Table 13.5 Ethernet Rule for Allowing ARP and STP Use

Logical Interface	Source	Destination	Service	Action	Options
ANY	ANY	ANY	ARP STP IPv4	Allow	Logging: None

2. Add another rule to block the use of Cisco Discovery Protocol (CDP) on the Inline Interface, and produce logs that will be stored:

Table 13.6 Ethernet Rule for Blocking CDP Use

Logical Interface	Source	Destination	Service	Action	Options
Inline interface	ANY	ANY	CDP	Discard	Logging: Stored

3. Add a rule on the last line of the Ethernet rules to block the use of other Ethernet protocols without producing logs:

Table 13.7 Ethernet Rule for Blocking Other Ethernet Protocols

Logical Interface	Source	Destination	Service	Action	Options
ANY	ANY	ANY	ANY	Discard	Logging: None

4. Save and install the policy on the IPS Engine.

Chapter 14

Access Rules

Access rules are lists of matching criteria and actions that define which traffic is allowed or discarded, as well as which allowed traffic is inspected against the Inspection Policy.

The following sections are included:

Overview of Access Rules

The IPv4 and IPv6 Access rules define which traffic IPS engines and Layer 2 Firewalls with Inline interfaces stop, which traffic is inspected against the Inspection Policy and which traffic is passed through without inspection. The Access rules are stored in policy elements, which are discussed in IPS and Layer 2 Firewall Policies (page 113).

The traffic matching in Access rules is based on source IP address, destination IP address, and port information included in the packets. Additional matching criteria that is not based on information in the packets includes the day of the week and the time of day (allowing you to enforce rules during specific times, such as working hours) and the physical network interfaces the traffic is picked up from.

The Access rules provide several different ways to react when some traffic is found to match a rule. You can:

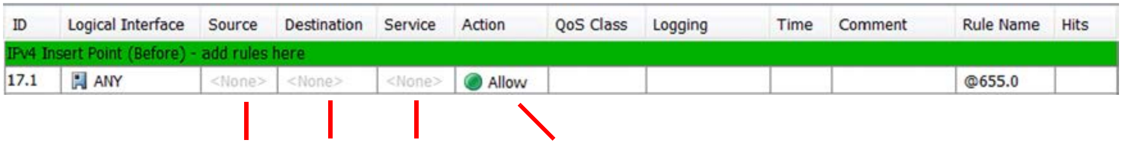
- Allow the traffic with inspection against the Inspection Policy.
- Allow the traffic without further inspection.
- Stop the traffic if the traffic passes through the inline interfaces of an IPS engine or Layer 2 Firewall with a license that allows this action.

Regardless of which of the above actions is taken, a matching rule can also create a log or alert entry.

Configuration of Access Rules

Access rules are configured on the **IPv4 Access** tab and the **IPv6 Access** tab inside IPS and Layer 2 Firewall Policy and Template Policy elements. You can also configure Access rules on the **IPv4 Access** tab and the **IPv6 Access** tab in IPS and Layer 2 Firewall Sub-Policy elements. You can create new IPv4 and IPv6 Access rules in the Policy Editing View, and also in the Logs view based on one or more selected log entries.

Illustration 14.1 Newly Inserted Access Rule - Main Cells



ID	Logical Interface	Source	Destination	Service	Action	QoS Class	Logging	Time	Comment	Rule Name	Hits
IPv4 Insert Point (Before) - add rules here											
17.1	ANY	<None>	<None>	<None>	● Allow					@655.0	

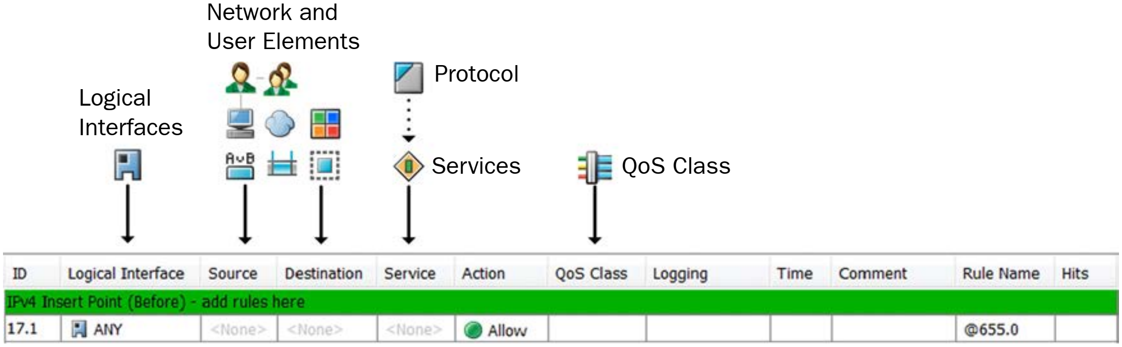
Mandatory cells for matching traffic

Engine applies Action when it finds a match

The illustration above displays an Access rule that has just been inserted into the policy. The matching cells are set to **<None>** to prevent the rule from having any effect on traffic in case a new rule is added to the policy accidentally. It is not necessary to modify all cells in each rule, but the mandatory cells for traffic matching (**Source**, **Destination**, and **Service**) must be set to some value other than **<None>** for the rule to be valid. The **Logical Interface** cell is also matched against traffic, but it is not mandatory to change its value if you want the rule to apply regardless of the interface. The other editable cells specify further conditions and options for handling connections that match the cells that are used for traffic matching.

Before starting to build policies, make sure you understand the network element types available and how you can use them efficiently to define the resources that you want to protect and control. The illustration below shows the types of elements that you can use in IPv4 and IPv6 Access rules.

Illustration 14.2 Elements in Access Rules



ID	Logical Interface	Source	Destination	Service	Action	QoS Class	Logging	Time	Comment	Rule Name	Hits
IPv4 Insert Point (Before) - add rules here											
17.1	ANY	<None>	<None>	<None>	Allow					@655.0	

The table below explains briefly what each Access rule cell does and which elements you can use in the rules. More information on each cell is included in the task flow later in this chapter. The cells are presented in the default order, but you can drag and drop them to your preferred order in your own Management Client.

Table 14.1 Access Rule Cells

Cell	Explanation
ID	(*Not editable*) Automatically assigned ID number that indicates the order of the rules in the policy. The rules are matched against traffic in the order of the ID numbers. For example, the rule 14.3 is the third rule added in this policy to the insert point that is the fourteenth rule in the upper-level template.
Logical Interface	Matches the rule based on which interface the traffic is picked up from. The same logical interface may be assigned to one or several interfaces as configured in the properties of the engine. This cell accepts only Logical Interface elements.
Source	A set of matching criteria that defines the IP addresses and interfaces that the rule matches. Both the Source and the Destination defined must match the source and destination of a packet for the packet to match the rule. The Source and Destination cells accept any elements in the Network Elements category, as well as User and User Group elements. Network Elements used in IPv4 Access Rules must contain IPv4 addresses, and Network Elements used in IPv6 Access Rules must contain IPv6 addresses.
Destination	
Service	The Services match a certain Protocol that defines the protocol for the traffic when it is further inspected against the Inspection Policy.
Action	Command for the engine to carry out when a connection matches the rule, and action-specific options for blacklisting, connection tracking, deep inspection (whether traffic is matched against the Inspection Policy), rate-based DoS protection, scan detection, and user responses.

Table 14.1 Access Rule Cells (Continued)

Cell	Explanation
QoS Class	The QoS Class that the engine assigns to connections that match this rule. Used in traffic prioritization and bandwidth management. The QoS Class has effect only if you set up QoS Policies.
Logging	The options for logging.
Time	The time period when the rule is applied. If this cell is left empty, the rule applies at all times.
Comment	Your optional free-form comment for this rule. If you add a rule from the Logs view, the Comment cell of the rule automatically includes information on the log entry which was used as the basis of the rule. Note that you can also add separate comment rows in between rules.
Rule Name	Contains a rule tag and optionally a rule name. Name: (*Optional*) Name or description for the rule. Displayed alongside the rule tag. Tag: (*Not editable*) Automatically assigned unique identification for the rule. Works as a link between the log entries and the rule that has generated the log entries. The rule tag consists of two parts (for example, @20.1). The first part of the tag is permanent and belongs to only that rule. The second part changes when the rule is changed. The first part and the second part are separated by a period.
Hits	*(Not editable)* Shows the number of connections that have matched the rule. This information is only shown if you have run a Rule Counter Analysis for the Policy. The cell shows “N/A” if no information is available.

Considerations for Designing Access Rules

One of the crucial issues in designing any policies is the order of the rules. The most important thing to keep in mind when editing the IPS Template Policies, IPS Policies, and IPS Sub-Policies is that the rules are read from top down. The actions **Allow**, **Refuse**, and **Discard** stop the processing from continuing down the Access rule table for any connection that matches the rule. Therefore, rules with any of these actions must be placed so that the more limited the rule is in scope, the higher up in the rule table it is. At its simplest, this principle means, for example, that an Access rule that allows connections to the IP address 192.168.10.200 must be put above an Access rule that stops all connections to the network 192.168.10.0/24. See Exempting Traffic from Inspection (page 149) for a more detailed example. If the traffic does not match any of the Access rules by the end of the policy, it is allowed by default.

Default Elements

The IPS Template and the Layer 2 Firewall Template have predefined Ethernet rules, IPv4 and IPv6 Access rules. Because the default policy elements are introduced when you import and activate a recent dynamic update package (for example, during the installation), the templates you currently have in your SMC may look slightly different from the one that is presented in this section. Newer versions of the templates work in the same way as described below. Any changes to the templates are documented in the Release Notes document for each dynamic update package.

There are several IPv4 and IPv6 Access rules with various Services defined with Continue as the action and a yellow insert point indicating the place where a Policy that uses the template can be edited.

The Access rules that you add at the insert points in custom policies based on the IPS Template or the Layer 2 Firewall Template are (in most cases) quite specific exceptions to the rules inherited from the template. For example, you could insert a rule there that allows a connection between two particular hosts to continue without any further inspection, or rules for inline IPS engines to always stop traffic between particular IP addresses and ports.

Configuration Workflow

The following sections provide an overview of the configuration tasks. Detailed step-by-step instructions can be found in the Management Client *Online Help* and the *McAfee SMC Administrator's Guide*.

Task 1: Define the Source and Destination

The Source and Destination cells specify the IP addresses that are compared to the IP addresses in each packet's header. Based on these and other criteria, the rule is applied to matching packets. By default, these cells are set to NONE, and you must change the value in both cells to make the rule valid.

You can add more than one element in each cell. You can optionally define detailed lists of matching criteria for each cell by combining Users (stored in an integrated Active Directory database), Network Elements, DNS Domain Names, and interface Zones. Each row of the list is combined with a logical AND: all items must match for the row to match. Groups, Aliases, Address Ranges, and Expressions are also useful for defining IP addresses in complex scenarios.

You can set the Source and Destination cells to ANY to make the rule match all possible source or destination IP addresses.

Task 2: Define the Service

The Service cell defines which protocol(s) the rule you design applies to, which also determines the protocol used in the Inspection Policy for matching traffic (the protocol that is detected and selected for traffic by an Access rule is a matching criteria in the Inspection Policy). By default, the Service is set to <None>, and you must change the value to make the rule valid.

The Service cell accepts only Service and Service Group elements. There are ready-made Services that cover most needs, but you may also use your own customized versions, for example, to define a non-standard port. The Services available for rule design are categorized according to protocols.

You can add more than one element in this cell to make the rule match several Services. You can optionally define detailed lists of matching criteria by combining URL Situations (for URL filtering), Applications, Services, and TLS matches. Each row of the list is combined with a logical AND: all items must match for the row to match.

Protocol Agent parameters are available for some Protocols in Service elements.

You can set the Service to ANY to make the rule match all protocols. A previous Continue rule may define a Protocol for traffic allowed by rules that use ANY as the Service (see Configuring Default Settings for Several Rules (page 146)). If there is no previous Continue rule that matches the same connection that would define a Protocol of the type Protocol Agent, a rule that allows ANY Service does not apply a Protocol of the type Protocol Agent to the traffic.

Note – For the most efficient inspection, select the correct Protocol of the type Protocol Agent when connection tracking is on (stateful inspection).

Task 3: Select the Action and Action Options

The Action cell defines what happens when a packet matches the rule. The available actions are explained in Table 14.2.

Table 14.2 Actions

Action	Explanation
Allow	The traffic is let through the engine.
Continue	Stores the contents of the Options cell and the Protocol option (inside the Service used) in memory and continues the inspection process. Used for setting options for subsequent rules as explained in Configuring Default Settings for Several Rules (page 146).
Discard	The traffic is silently dropped if going through an inline interface pair on IPS engine or Layer 2 Firewall with the appropriate license.
Refuse	The traffic is dropped if going through an inline interface pair on an IPS engine or Layer 2 Firewall with the appropriate license. An ICMP error message is sent in response to notify the packet’s sender.
Jump	Matching is continued in the specified sub-policy until a match is found. If there is no matching rule in the sub-policy, the process is resumed in the main policy.
Apply Blacklist	Checks the packet against the blacklist according to the options set for this rule. If the packet matches a blacklist entry and is going through an inline interface, the engine discards the connection.

Each IPv4 and IPv6 Access rule has action-specific options for blacklisting, connection tracking, deep inspection, rate-based DoS protection, scan detection, and user responses.

Table 14.3 Action Options

Action Option	Explanation
Blacklisting	Defines which blacklist entries are enforced for connections that match the rule based on the component that added the blacklist entry to the blacklist. A restriction based on the blacklist sender may be necessary, for example, if the same IP address exists in two different networks. The default setting is to enforce all blacklist entries regardless of the component that created the entry. Used by rules with the Apply Blacklist action.
Connection Tracking	Defines whether the IPS or Layer 2 Firewall engine keeps a record of the currently open connections (stateful inspection). See Connection Tracking vs. Connectionless Packet Inspection (page 124) for more information. Used by rules with the Allow, Continue, or Jump action.
Deep Inspection	Defines whether matching connections are also inspected against the Inspection rules. Used by rules with the Allow, Continue, or Jump action.
Rate-Based DoS Protection	Defines whether rate-based DoS protection is enabled for traffic that matches the rule. Used by rules with the Allow, Continue, or Jump action.

Table 14.3 Action Options (Continued)

Action Option	Explanation
Scan Detection	Defines whether scan detection is enabled for traffic that matches the rule. Used by all rules.
User Response	Defines which automatic response is shown to the end-user when an HTTP connection is discarded. Used by rules with the Discard action.

Task 4: Select Logging Options

Rules can create a log or alert entry each time they match. By default, logging options set in a previous rule with Continue as its action are used. If no such rule exists, Layer 2 Firewalls and Virtual Layer 2 Firewalls log the connections by default. IPS engines and Virtual IPS engines do not log the connections by default. Each individual rule can be set to override the default values.

The log levels are explained in Table 14.4.

Table 14.4 Log Levels in Access Rules

Log Level	Explanation
None	Does not create any log entry.
Transient	Creates a log entry that is displayed in the Current Events mode in the Logs view (if someone is viewing it at the moment) but is not stored.
Stored	Creates a log entry that is stored on the Log Server.
Essential	Creates a log entry that is shown in the Logs view and saved for further use.
Alert	Triggers an alert entry.

When the Log Server is unavailable, log entries are temporarily stored on the engine. When the engine is running out of space to store the log entries, it begins discarding log data in the order of importance. Monitoring data is discarded first, followed by log entries marked as Transient and Stored, and finally log entries marked as Essential. The Alert entries are the last log entries to be discarded.

The settings for storing the logs temporarily on the engine are defined in the engine's log spooling policy. For more information, see the *McAfee SMC Administrator's Guide*.

Task 5: Restrict the Time When the Rule Is Enforced

Optionally, you can set a specific time period when a rule is applied using the **Time** cell. The validity of the rule can be set by month, day of the week, and time of day. For example, you might have certain rules that allow access only during business hours on weekdays. If you leave the **Time** cell empty, the rule is always valid.

Note – The times are entered in Coordinated Universal Time (UTC), and you must adjust the times you enter to make them correspond to the engine's local time zone. Also consider that UTC time does not adjust to daylight saving time (summer time).

Using Access Rules

The general configuration of Access rules is explained above. The sections below provide further information on configuring Access rules:

- Allowing System Communications
- Configuring Default Settings for Several Rules (page 146)
- Rematching Tunneled Packets (page 148)
- Using Aliases in Access Rules (page 148)

For general information on using rules, see Using Policy Elements and Rules (page 123).

Allowing System Communications

If NAT is applied to system communications, you must create Location elements and add Contact Addresses for the elements to define which translated addresses are necessary for making contact. Only IPv4 addresses are used in system communications.

If you have inline IPS engines or Layer 2 Firewalls, be careful that you do not define rules that would prevent other SMC components from communicating with each other. The system communications are detailed in Default Communication Ports (page 217).

There are predefined Service elements for all system communications. You can use these in your Access rules as necessary.

Configuring Default Settings for Several Rules

You may want to set default values for some settings in rules to avoid defining the same settings for several rules individually. The **Continue** action is used to set such default values.

The options that can be set using Continue rules in Access rules includes:

- The Connection Tracking options:
    - The Idle Time-out option overrides the global defaults set in the IPS or Layer 2 Firewall element's properties.
    - The concurrent connection limits define the maximum number of connections allowed from a single source and/or destination IP address.
- The Protocol option inside the Service used.
- The Logging options.

When a connection matches a rule with Continue as the action, the rule's settings are written in memory but the matching continues until another rule that matches is found. This matching rule uses the defaults set in the Continue rule unless the rule specifically overrides the defaults with different settings. This way, you do not have to define the settings for each rule separately.

You can use Continue rules to set default settings for a general type of traffic and define settings for individual rules only when specifically required. A Continue rule can be overridden by some subsequent Continue rule that has an identical scope (Source, Destination, and Service), or partially overridden by a Continue rule that partially overlaps with the previous Continue rule. When you define Continue rules with different matching criteria, you can have several Continue rules one after another without them interfering with each other in any way at all.

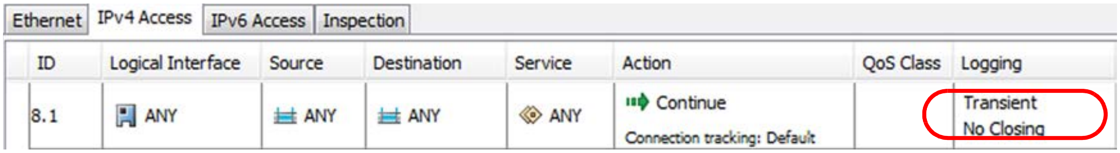
Continue rules are defined in the same way as other rules. However, you must keep in mind that many or even all rules below may be affected. Options in Continue rules are used by the rules below, provided that the Source, Destination, and Service match the same connection as the Continue rule. Continue rules are inherited from Template Policies into lower-level templates and policies like any other rules.

Sub-Policies may require special attention with Continue rules: the Sub-Policies may have different options when you insert them into different policies if the Sub-Policy rules do not override the options set by preceding Continue rules. Also, when a Sub-Policy contains a Continue rule, the options are then used for further matching in the higher-level policy (if the processing returns to the higher-level policy).

Using Continue Rules to Set Logging Options

One common use for the Continue action is to set the default log level for all subsequent rules. Instead of setting the log level for all rules individually, you can set a Continue rule in a template or in a policy to set the default log level. The log level for any subsequent matching rules can be left undefined. The rules trigger logging as defined in the Continue rule.

Illustration 14.3 Setting the Default Log Level



ID	Logical Interface	Source	Destination	Service	Action	QoS Class	Logging
8.1	ANY	ANY	ANY	ANY	Continue Connection tracking: Default		Transient No Closing

In Illustration 14.3, the default log level is set to Transient for any source, destination or service. All subsequent rules in this policy and any sub-policies log Transient by default. Individual rules can still override this option with specific log levels, such as Essential or Stored.

If logging is not defined for a rule and there is no prior Continue rule that sets logging options, the default log level is **Stored**.

Using Continue Rules to Set the Protocol

Default Protocols are set using the Continue action. This way, the correct Protocol is used also for traffic that is allowed by rules that match any Service (and therefore have no particular Service element that would set the correct protocol). The Protocol is needed to associate the traffic with the correct protocol for further inspection and to handle some types of traffic, such as FTP, correctly. The IPS Template and the Layer 2 Firewall Template include several Continue rules that associate all traffic with Protocols according to standard ports.

If you have TCP and UDP services set up in your network under non-standard ports, the traffic may not be associated to the correct protocol and be therefore inspected at a more general (TCP or UDP) level. In this case, you can create your own custom Situation for the traffic and add it in your policy to have the traffic inspected with the correct protocol information. Only some protocols and some of their parameters are supported in the services that are used in IPS policies.

You can also add your own rules for the opposite purpose: to have some traffic not inspected as a particular protocol, but more generally as TCP or UDP traffic. In this case, you add a rule in your policy that includes the general TCP or UDP Service element from the **IP-proto** branch of the Services tree.

Rematching Tunneled Packets

If an engine inspects traffic that is tunneled using IP-in-IP tunneling or Generic Routing Encapsulation (GRE), the traffic can be checked against IPv4 Access rules and/or IPv6 Access rules several times according to the number and type of layers in the tunnel. For example, when an IPv4 datagram contains an IPv6 datagram, the IPv4 datagram is first matched according to Access rules. If the tunneling Service in the Access rule specifies that the encapsulated IPv6 datagram should be matched again, the contents are then matched against the IPv6 Access rules.

To limit the number of encapsulating layers, the engine properties define the maximum rematch count. By default, the maximum rematch count is 1. If this count is exceeded, the packet is allowed or discarded according to the setting specified in the engine properties and a log or an alert is generated.

Using Aliases in Access Rules

Aliases are one of the most useful tools for reducing the complexity of a policy. In a sense, Aliases are like variables in a mathematical equation—their value changes depending on the component on which they are installed. Because Aliases are able to change their meaning to adapt to local contexts, they can be used to create a single rule that changes in meaning depending on where it is installed. Thanks to Aliases, you can use a single rule to replace multiple, near duplicate rules created separately for each engine.

To better understand this concept, let us consider an example company, which has its headquarters in Helsinki and branch offices in Atlanta, Munich, Tokyo, and Montreal. Each of these offices has its own web server. In this scenario, it seems we would require a separate rule or set of rules for each location's web server.

By using aliases, however, we can create a single rule or set of rules that is still valid in all parts when applied on different components.

The administrator of the example company can create a web server alias, **`$WebServers`**. In the Alias's properties, the administrator defines what **`$WebServers`** means for each component. For the IPS engine in Helsinki, the web server would be defined as 192.168.1.101, for the IPS engine in Tokyo as 192.168.2.101, and so on.

When the administrator installs a policy containing the web server rules with the Alias, the addresses are translated to the correct address on that component. Therefore, when the policy is installed on the Helsinki IPS engine, the Alias translates to an IP address of 192.168.1.101. The other addresses are not included in the policy that is transferred to that particular engine.

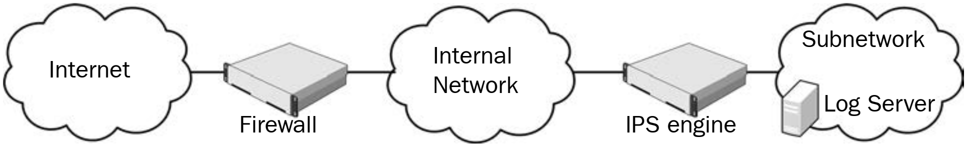
Examples of IPS Access Rules

The examples in this section illustrate some common uses for Access rules and general steps on how each scenario is configured.

Exempting Traffic from Inspection

At Company A, there is an IPS engine deployed between the general office network and a subnetwork.

Illustration 14.4 Company A's Networks



In the subnetwork, there are several servers that provide services to the general office network as well as the Management Server and Log Server. There is also a Firewall deployed between the internal and external networks. There is heavy traffic to the subnetwork where the internal servers are, so the administrators decide to exempt the log transmissions between the Firewall and the Log Server from being inspected against the Inspection Policy to reduce the IPS engine's workload. The administrators:

1. Create a new IPS policy based on the IPS Template to replace the Default IPS Policy that they have currently installed.
2. Add a new rule in the Access rules for their IPS engine:

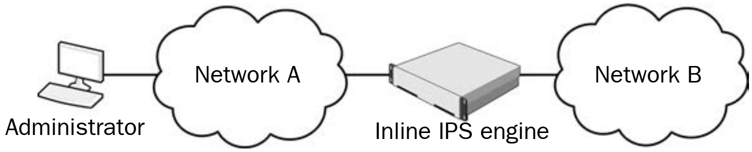
Table 14.5 Access Rule for Exempting Traffic from Inspection Against the Inspection Policy

Source	Destination	Service	Action	Options
Firewall	Log Server	SG Engine to Log	Allow	Deep inspection: Off

Filtering Traffic on an Inline IPS engine

Administrators at company B decide that they want more control over which hosts and ports can be used between two networks.

Illustration 14.5 Company B's Network



Hosts in the two networks must be able to communicate between each other using certain specific ports. Additionally, one of the administrators has a workstation connected to Network A. The administrator's workstation must have unrestricted access to Network B. The administrators decide that the inline IPS engine provides an acceptable level of security at this point between two internal networks.

The administrators:

1. Create elements for network A, network B, and administration host.
2. Add new Access rules for their inline IPS engine:

Table 14.6 Access Rules for Filtering Traffic

Source	Destination	Service	Action	Options
Administrator Network B	Administrator Network B	ANY	Allow	Logging: Undefined Deep inspection: On
Network A Network B	Network A Network B	Service elements for allowed services	Allow	Logging: Stored Deep inspection: On
ANY	ANY	ANY	Refuse	Logging: Stored Deep inspection: (irrelevant, because dropped traffic is never inspected further)

- Each of the first two rules allows traffic between the Source and the Destination in both directions. The order of the elements within the Source, Destination, and Service cells makes no difference to the outcome of the matching process.
- The order of the rules is important. The rules above proceed correctly from most specific to the least specific. The two first rules must be in this order, because the administrators want all connections from the Administrator host (which is in Network A) to always match the first rule and never the second one, since the rules have different logging options.
- The last of the added rules stops all traffic that is not allowed in the rules above to prevent unauthorized traffic from passing.

Note – If the inline interfaces are on a fail-open network card, traffic passes freely whenever the IPS engine is offline regardless of what the Access rules state.

Chapter 15

Inspection Policies

Inspection Policies define how the engines look for patterns in traffic allowed through the Access rules and what happens when a certain type of pattern is found.

The following sections are included:

Overview of Inspection Policies

Inspection Policies define how the main traffic analysis is done for traffic that has been allowed and selected for inspection in the Access rules. The Inspection Policies are selected in IPS and Layer 2 Firewall Policy elements, which are discussed in IPS and Layer 2 Firewall Policies (page 113).

Inspection Policies examine the entire contents of the packets throughout whole connections to see if the data being transferred contains a pattern of interest. Dynamic update packages are the main source of these patterns, but you can also define new patterns as Situation elements, which are discussed in Situations (page 105).

Virtual Security Engines select traffic for inspection, but they do not directly inspect traffic. One shared inspection process running on the Master Engine handles the inspection and correlation of traffic from all Virtual Security Engines associated with the Master Engine. Master Engines also inspect their own communications.

There are three general types of cases for using Inspection Policies:

- You can detect attempts to exploit known vulnerabilities in your systems and prevent such attempts from succeeding if the system is not patched against it.
- You can monitor traffic that does not cause alarm on the surface, but when examined for certain patterns, may turn out to conceal actual threats. For example, you can detect if a series of occasional service requests are actually someone secretly scanning the network structure or if a spike in traffic is a denial-of-service attack under way.
- You can also detect other sequences in traffic, such as the use of certain applications or even access to a particular file.

Based on the detection results, the Inspection Policy provides several different ways to react when some traffic is found to match a pattern of interest:

- Stop the traffic if it is going through an engine with inline interfaces.
- Reset the connection.
- Blacklist the connection on one or more Security Engines.
- Allow the traffic.

Regardless of which action is taken, a match can also create:

- A log entry with or without recording some of the detected traffic.
- An alert with or without recording some of the detected traffic.

Configuration of Inspection Policies

IPS engines and Layer 2 Firewalls inspect traffic based on Situation elements, which contain the information about traffic patterns. Patterns may trigger immediate responses or just be recorded. Detected events can be matched against Correlation Situations, which combine and further analyze the traffic-based findings to detect additional threats and produce an easy-to-read event stream.

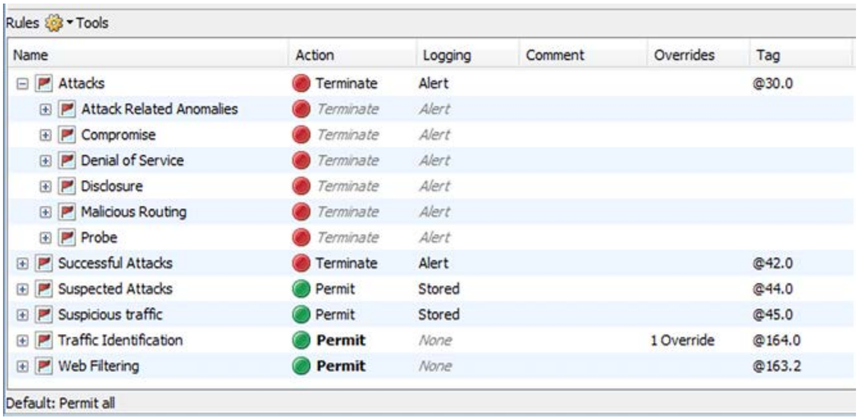
Inspection Policies are selected on the **Inspection** tab inside IPS Policy, IPS Template Policy, Layer 2 Firewall Policy, and Layer 2 Firewall Template Policy elements. Sub-Policies cannot contain Inspection Policies. You can add new rules to the Inspection Policy in the Policy Editing View and also in the Logs view based on log entries.

The Inspection Policy has two parts:

- The **Inspection** tab contains the main rules for finding traffic patterns. The Rules tree is applied to all traffic that is not handled as Exceptions.
- The **Exceptions** tab contains rules that match specific portions of the traffic based on Logical Interface, IP addresses, and Ports. Exceptions have some additional options, and can also set some of those options for the main Rules through the use of *Continue* rules.

The main Rules tree on the Inspection tab contains a tree of Situations, which are organized under Situation Types. This tree allows you to control which inspection checks trigger a reaction and which checks are ignored. The Rules tree defines general checks that are applied to all patterns that are not handled by a more specific definition. It is not possible to limit the scope of the checks by IP addresses or Logical Interfaces in the Rules tree.

Illustration 15.1 Inspection Tab - Rules Tree



Rules Tools

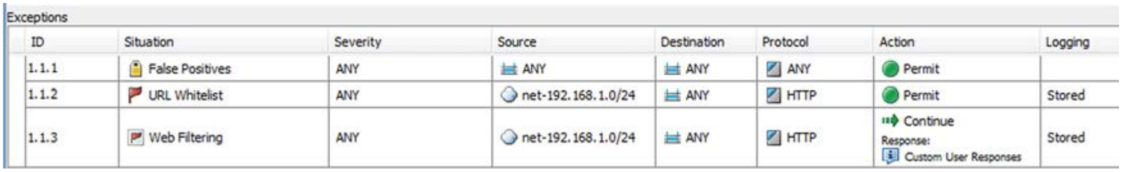
Name	Action	Logging	Comment	Overrides	Tag
Attacks	Terminate	Alert			@30.0
Attack Related Anomalies	*Terminate*	*Alert*			
Compromise	*Terminate*	*Alert*			
Denial of Service	*Terminate*	*Alert*			
Disclosure	*Terminate*	*Alert*			
Malicious Routing	*Terminate*	*Alert*			
Probe	*Terminate*	*Alert*			
Successful Attacks	Terminate	Alert			@42.0
Suspected Attacks	Permit	Stored			@44.0
Suspicious traffic	Permit	Stored			@45.0
Traffic Identification	**Permit**	*None*		1 Override	@164.0
Web Filtering	**Permit**	*None*			@163.2

Default: Permit all

The Exceptions are matched before the main rules. The most frequent use of Exceptions is to eliminate false positives, which typically require permitting a pattern for some part of the traffic while it still triggers a reaction when encountered in any other traffic.

The illustration below shows the Exceptions panel with some rules.

Illustration 15.2 Exceptions Panel



Exceptions

ID	Situation	Severity	Source	Destination	Protocol	Action	Logging
1.1.1	False Positives	ANY	ANY	ANY	ANY	Permit	
1.1.2	URL Whitelist	ANY	net-192.168.1.0/24	ANY	HTTP	Permit	Stored
1.1.3	Web Filtering	ANY	net-192.168.1.0/24	ANY	HTTP	Continue Response: Custom User Responses	Stored

The main matching cell is the Situation that contains the actual patterns. The other matching cells are Logical Interface, Source, Destination, Protocol, and Time. The role of the other matching cells is to limit the scope of the rule to some specific traffic, for example, to take different action based on which host is the sender or receiver of traffic identified as malicious.

The cells are explained in more detail in Exception Rule Cells (page 156).

Verifying and Tuning Inspection

The most common way to start using IPS is to start with a default policy. A general policy that is meant to work in all environments is not perfectly suited to your particular network scenario. A tuning period is needed to activate and deactivate inspection checks based on the findings and your particular needs. Tuning the policy is important, since with a small tuning effort, you can save a lot of time due to the increased relevancy and accuracy of the findings that the engine generates.

To assist in policy tuning, you can use *passive termination*. When passive termination is used, the engine creates a log entry that notes that a certain connection was selected for termination, but the engine does not actually terminate the connection. This allows you to check the logs and adjust your policy without the risk of cutting important business communications. There are two levels of activating this feature:

- Passive termination can be activated globally in the IPS or Layer 2 Firewall element properties for the initial policy tuning.
- Later on, you can test newly added Situations by setting individual Exception rules to passive termination mode.

For cautious introduction of new Situations introduced in dynamic update packages, you can utilize the Tags for the five most recent updates (**Situations→By Tag→By Situation Tag→Recent Updates)**.

Considerations for Designing Inspection Policies

The basic design principle is the same as in other rules: the rules are read from top down, and more specific rules must be placed above more general rules that match the same traffic. The detailed rules specific to some IP addresses and Protocols is defined as Exceptions. The general rules that are applied to remaining traffic are defined in the Rules tree on the Inspection tab.

The traffic matching in Inspection rules and exceptions is different from other types of rules, because it is done based on the traffic pattern definitions in Situation elements. The engines monitor the network for all patterns included in the policy. When a pattern is found, the

Inspection rules and exceptions are matched based on the Situation element that contained the detected pattern. Inspection rules and exceptions match certain patterns only. Non-matching traffic is passed through without any reaction.

The Situation-based configuration logic means that the behavior of the Inspection rules and exceptions can change without anyone editing the policy directly. Just creating a new Situation element may include the Situation in the policy if the Situation is associated with a Situation Tag or Situation Type grouping included in the policy. For more information about Situation elements, see Situations (page 183).

Actual rules may look quite different even if they refer to the exact same Situation, since Situations have grouping mechanisms. However, it makes no difference in matching a pattern whether you add the Situation as a single element or together with other Situations through a Situation Tag or Situation Type.

The Permit action allows the traffic pattern and the Terminate action stops the traffic that matches the pattern. A Permit action does not unconditionally allow the traffic, because processing still continues to look for other patterns, but a Permit match does prevent the exact same Situation from matching again if it appears at any point further down in the policy.

Example **Situation A matches a Permit rule with logging level set to “None”. A second rule that contains Situation A exists below the first rule in the policy with Terminate as the action and logging level set to “Stored”. The logs do not show any matches to Situation A and the traffic that matches the pattern continues uninterrupted.**

Similarly, the Terminate action prevents the same Situation from matching again as the policy is processed to the end, but does not prevent other Situations from matching simultaneously.

Each Situation element is considered as a unique pattern (with the exception that is discussed below). Avoid defining the exact same pattern in different Situation elements, because such duplicates in the policy can create unintended results and makes the policies difficult to manage.

Example **A Continue rule sets a User Response for Situation A, which matches the URL www.example.com. A different rule specifies Termination for Situation B, which also matches www.example.com. When the users access the URL, their connections are terminated without a User Response, because the User Response is set for Situation A and the traffic is terminated by Situation B. The configuration handles these as two separate patterns.**

The exception where one Situation is specifically used in the configuration to prevent a different Situation from matching is URL filtering. When you whitelist URLs, the special URL filtering Situations stop further URL-based matching.

Example **A URL filtering category defined in Situation A prevents users from accessing www.example.com (among other sites). The administrators add www.example.com to a custom Situation B that is permitted higher up in the policy. Users can now access www.example.com. With other types of Situations, matching connections would continue to be terminated if two different Situations were used.**

Exception Rule Cells

The table below explains briefly what each Exception rule cell does.

Table 15.1 Exception Rule Cells

Cell	Explanation
ID	*(Not editable)* Automatically assigned ID number that indicates the order of the rules in the policy. The rules are matched against traffic in the order of the ID numbers. For example, rule 1.3 is the third rule added in this policy to the insert point that is the first Inspection rule in the upper-level template.
Situation	Defines the patterns of traffic that the rule matches. In addition to individual Situation elements, this cell may contain Situation Type and Tag elements, which are shown as branches in the Situations tree and allow adding the whole branch of Situations at once to a rule.
Logical Interface	Limits the rule based on which interface the traffic is picked up from. The same logical interface may be assigned to one or several interfaces as configured in the properties of the IPS engine. This cell accepts only Logical Interface elements.
Severity	Limits the rule to those matching Situations that have a severity value within a range you define. This is most useful with rules that include Situation Tags in the Situation cell.
Source	Limit the IP addresses that the rule matches, for example, to create different responses to the same pattern depending on the communicating hosts. The Source and Destination cells accept any elements in the Network Elements branch.
Destination	
Protocol	Limits the Protocols that the rule matches. The protocol is set for traffic in the Access rules in the Service cell of the rule that allows the traffic. The Protocol cell allows you to limit the scope of an Inspection rule based on the protocol that an Access rule has assigned.
Action	Command for the engine to carry out when a connection matches the rule. Action-specific options for blacklisting, connection termination, and user response. The Continue action can be used to set action-specific Action Options for Exceptions and (depending on the option) for the Rules tree as explained in Setting Default Options for Several Inspection Exceptions (page 160).
Logging	Options for logging.
Time	Limits the time period when the rule is applied. If the cell is empty, the rule applies at all times.
Comment	Your free-form comment for this rule. If you add a rule from the Logs view, the Comment cell of the rule automatically includes information on the log entry which was used as the basis of the rule. Note that you can also section the rules under comment rows.

Table 15.1 Exception Rule Cells (Continued)

Cell	Explanation
Rule Name	Contains a rule tag and optionally a rule name. Name: (*Optional*) Name or description for the rule. Displayed alongside the rule tag. Tag: (*Not editable*) Automatically assigned unique identification for the rule. Works as a link between the log entries and the rule that has generated the log entries. The rule tag consists of two parts (for example, @20.1). The first part of the tag is permanent and belongs to only that rule. The second part changes when the rule is changed. The first part and the second part are separated by a period.

Default Elements

There are several Inspection Policy templates, which are introduced when you import and activate a dynamic update package. The rules in the Inspection Policy templates may change when you activate new update packages. The table below lists the default Inspection Policy template elements.

Table 15.2 Default Inspection Policy Template Elements

Template	Description
Empty Inspection Policy	Inspection Policy with a set of Inspection rules that do not enforce inspection.
Loose Inspection Policy	Inspection Policy with a set of Inspection rules for detecting common threats. Loose Inspection Policy logs Situations categorized as Suspected Attacks but allows the traffic to pass.
Strict Inspection Policy	Inspection Policy with a set of Inspection rules for detecting common threats. Strict Inspection Policy terminates Suspected Attacks with an alert. Strict Inspection Policy is suitable as the initial policy in most environments. The Strict Inspection Policy terminates a connection if the engine cannot see the whole connection. It is recommended that you use the Strict Inspection Policy as a starting point for your Inspection Policies.
Customized Strict Inspection Policy	Inspection Policy that is based on the Strict Inspection Policy and contains a set of customized Inspection rules. Customized Strict Inspection Policy was used when the IPS was tested at ICSA Labs and NSS Labs. It provides an example of a customized Inspection Policy.

Configuration Workflow

The following sections provide an overview of the configuration tasks. Detailed step-by-step instructions can be found in the Management Client *Online Help* and the *McAfee SMC Administrator's Guide*.

Note – Keeping your system up-to-date with latest dynamic updates is an essential part of maintaining your Inspection Policies. See the Management Client *Online Help* for information on dynamic updates and instructions for enabling automatic update download and activation.

Task 1: Create an Inspection Policy

To customize inspection, you must have a custom Inspection Policy element. The pre-defined templates are a good starting point for your own customizations. Policy elements are discussed in detail in IPS and Layer 2 Firewall Policies (page 113).

Task 2: Activate Deep Inspection in IPS and Layer 2 Firewall Policies

Typically, you introduce deep inspection after creating and testing initial Access rules. You must specifically activate deep inspection for the portion of traffic that you want to deep inspect. This is done in the Access rules. See Access Rules (page 99) for more information. You also select which Inspection Policy is used for deep inspection on the Inspection tab of the IPS or Layer 2 Firewall Policy.

Task 3: Activate the Relevant Inspection Checks

Traffic patterns of interest are defined in Situations, so the inspection checks are based on selecting the desired reaction to the Situations when the pattern is found in network traffic. It is not mandatory to create any additional Situations to activate inspection checks, since there are many default Situation elements and they are continuously updated through dynamic update packages.

The Rules tree on the Inspection tab is the main tool that allows you to select which traffic patterns are permitted and stopped, whether a log entry or an alert is triggered, and whether matching traffic is recorded. All Rules in the Rules tree can be edited, including overrides that have been set in a higher-level template. The Rules tree can contain a maximum of one instance of each Situation to prevent the definitions within the Rules tree from overlapping.

Task 4: Define the Exceptions

The Exceptions tab allows you to create detailed rules, which are processed before the Rules tree definitions on the Inspection tab. The Exceptions have additional features compared to the Rules tree:

- You can make exceptions to the general Rules tree definitions based on Source, Destination, and Protocol information.
- You can set options for connection termination (including User Responses) in addition to the options that are available in the Rules tree. The Response options define an automatic client notification for any HTTP connection that is terminated.
- You can create Continue rules to set Action Options and general rule Options for other Exceptions and the Rules tree. The Rules tree contains specific definitions for logging, so the logging options set with Continue rules do not affect traffic that matches the Rules tree.

- You can create rules in Inspection Policy templates that cannot be changed in the inheriting policies.
- You can create rules that are applied only on certain days and/or times of day.

In addition to individual Situation elements, the Situation cell may contain Tag and Situation Type elements, which are shown as branches in the Situations tree and allow adding the whole branch of Situations at once to a rule. Most of the Situations you add to the Exceptions are those that you consider false positives in your environment (for example, Situations for exploit attempts against an operating system that is not used in your organization).

In the Exceptions, it is highly unusual to set the Situation cell to ANY. This is not useful in most cases because the patterns that Situations define range widely from Situations that detect something as benign as the use of particular applications to something as malicious as successful attacks on a servers. This also creates unnecessary load on the engines, as a high number of Situations is checked in each matching connection.

Task 5: Eliminate False Positives

As the Inspection rules and exceptions are matched to traffic, there are always some occurrences of false positives (matches that are incorrect or irrelevant in your environment). By tuning the Inspection Policy to the actual traffic and applications in your network environment, you can increase the relevance of inspection results greatly. To eliminate a false positive, you adjust either the Inspection Rules tree or the Exceptions depending on whether the change should be applied globally or to traffic between specific hosts. An easy way to create new Exceptions is to use an existing log entry as the basis: you can create Exceptions through the right-click menu of log entries. See the Eliminating a False Positive (page 161) example for a practical overview of one approach to eliminating a false positive.

Task 6: Add Custom Inspection Checks

If you want to detect some pattern in traffic that is not defined in the predefined Situations (for example, a particular internal file server in your network being accessed) or if you want to create a modified version of some existing Situation, you can create a new Situation element. This is explained in Configuration of Situations (page 106). You can add your custom Situations to the Rules tree by selecting a Situation Type for them.

Using Inspection Policies

For general information on using rules, see Using Policy Elements and Rules (page 123).

Setting Default Options for Several Inspection Exceptions

You may want to set default settings for some Exception rules to avoid defining the same settings for several rules individually. The Continue action in Exception rules is used to set such default options in the same general way as in the Access rules. See Configuring Default Settings for Several Rules (page 146). In Exception rules, all settings in the Action Options and the Logging cell can be set using Continue rules. However, the Rules tree on the Inspection tab ignores any logging options set with Continue rules. In the Rules tree, the rules either inherit the logging settings from a higher level in the tree or define a specific logging option as an override.

Importing Snort Rules Libraries

You can import rule definitions from an existing Snort rules library (`.rules`) files. Importing a Snort rules library creates a new Inspection Policy. Each Snort rule is converted into a Situation element and an Exception rule in the Inspection Policy:

- The Action and Source/Destination Network parameters in the Snort rules are used to define the Exception rule.
- The Snort rule options are used to define the Situation element. The Situation element is used in the corresponding Exception rule.

The original Snort rule is included as a comment in the Context of the Situation. For more information about Situation elements, see Situations (page 105).

Example of Inspection Policies

The example in this section illustrates a common modification to the Inspection Policy and general steps on how the scenario is configured.

Eliminating a False Positive

The administrators in this example have just started using Inspection rules. They have installed a policy that includes only the rules defined in the Loose Inspection Policy. When they install the IPS Policy, they soon start receiving alerts.

After some investigation, the administrators realize that the alert is caused by a custom-built application, which communicates in a way that happens to match the pattern of how a certain exploit would be carried out by an attacker. The custom-built application is only used by a specific server and a few clients in the internal network, so they quickly modify the Inspection Policies to exclude those particular hosts for the Situation in question. The administrators:

1. Create Host elements to represent the server and the clients.
2. Create a Group element that includes the client's Host elements.
   - The administrators name the Group so that it is immediately clear from the name that the Group contains those hosts that must contact the server running their custom-built application. This makes the new rule easier to read than if they included the hosts directly in the rule.
3. Add the following rule on the Exceptions tab in their Inspection Policy:

Table 15.3 Rule for Eliminating a False Positive

Situation	Source	Destination	Action	Logging
The Situation element that is mentioned in the alerts in the Logs view.	The Group defining the clients.	The Host for the internal server.	Permit	None

   - If the Situation matches traffic between any other hosts than those included in the Group, the IP address does not match those defined in the new rule, so the processing will continue to the next rule, which terminates the traffic and triggers an alert.
   - The logging would not have to be set to None, because it is the default option, but the administrators want to do so anyway to make sure any rules they add in the future cannot accidentally set logging on for this rule.
4. Refresh the policy on the IPS engines.

Chapter 16

Protocol Agents

Protocols of the *Protocol Agent* type are special modules for some protocols and services that require advanced processing. Protocol Agents can enforce policies on the application layer.

The following sections are included:

Overview of Protocol Agents

Protocol Agents are software modules for advanced processing of some protocols that require special handling on the Layer 2 Firewall or the IPS engine due to their complexity, address information in the data payload, related connections, etc. Protocol elements also associate the traffic with a certain protocol for inspection against the Inspection Policy.

Protocol Agents on Layer 2 Firewalls and IPS engines can:

- Validate application-level protocol use (for example, FTP command syntax).
- Open related connections when required (for example, FTP data connections).

Some protocols always require the use of the correct Protocol Agent to pass inspection by the Layer 2 Firewall or the IPS.

Connection Handling

When related new connections are opened based on information exchanged in an initial connection, Protocol Agents may be needed. Protocol Agents are provided to handle the following protocols:

- FTP with related active and passive data connections.
- Microsoft RPC (MSRPC) for Microsoft Exchange and Outlook communications.
- NetBIOS for the Windows NetBIOS datagram services.
- Oracle TNS protocol communications.
- Remote Shell protocol communications.
- SunRPC portmapper communications.
- TFTP file transfers.

Example **File Transfer Protocol (FTP) uses two related connections: a control connection and a separately established data connection. If the control connection is allowed without the Protocol Agent, the IPS engine does not recognize that the data connection is part of an existing connection and handles it as a new connection (which usually leads to failed data transfer).**

Protocol Validation

Protocol Agents can be used to validate communications against standards of specific protocols. Exactly how this works depends on the protocol in question.

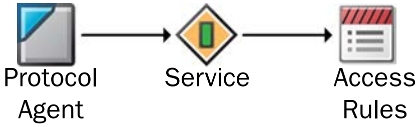
A few examples:

- The FTP Protocol Agent can be set to strictly limit the allowed commands within the control connection to standard commands as listed in the FTP specifications. If other commands are sent in the control connection, the connection is dropped.
- The Oracle Protocol Agent can control the size of the Oracle TNS packets, or the location of the Listener service with respect to the database services.
- The SSH Protocol Agent can ensure that the SSH handshake is performed at the beginning of an SSH connection.

Configuration of Protocol Agents

Protocol Agents are represented in the Management Client by Protocol elements that have *Protocol Agent* as their type. Other Protocol elements are of the type *Protocol Tag*.

Illustration 16.1 Using Protocol Agents



Protocol Agents are not included directly in IPS policies or Layer 2 Firewall Policies. They are used inside custom Service elements that you create. The custom Service elements are used in Access rules. Whenever traffic matches a rule that contains a Service element with an associated Protocol Agent, the Protocol Agent is automatically activated.

All Protocol Agents are default elements, and you cannot change them or add any new ones. There are also default Service elements for most supported protocols that you can use to activate the Protocol Agents. However, some Protocol Agents have parameters and options you can set by creating customized Services as explained below.

Configuration Workflow

The following sections provide an overview of the configuration tasks. Detailed step-by-step instructions can be found in the Management Client *Online Help* or the *McAfee SMC Administrator's Guide* PDF.

Task 1: Create a Custom Service with a Protocol Agent

There are default Service elements that refer to Protocol Agents. These default Services can be used without additional configuration in the Access rules. However, the default Services do not allow you to change the default parameters of Protocol Agents that have configurable parameters. If you want to modify the way a Protocol Agent behaves, you must create a new custom Service of your own and attach the correct Protocol Agent to that Service. The Service element contains the identifying information, such as a port number, that determines which traffic the Service matches. In most cases, this alone ensures that the Protocol Agent is not applied to the wrong type of traffic.

Task 2: Set Parameters for the Protocol Agent

If you create your own custom Service that uses a Protocol Agent that has configurable parameters, you can specify parameters for the Protocol Agent in the properties of the Service. The Protocol Agents are listed in Using Protocol Agents (page 167). See the Management Client *Online Help* or the *McAfee SMC Administrator's Guide* PDF for information on the parameters for the Protocol Agents.

Task 3: Insert the Service in Access Rules

Whether you create a custom Service or use one of the predefined Services that have a Protocol Agent attached to them, you must define the traffic the Protocol Agent handles in the Access rules in your Layer 2 Firewall Policies or IPS Policies.

A Protocol Agent can be set either on a rule-by-rule basis, or you can create a rule with Continue as the rule's Action. When there is a Continue rule, rules further down in the rule table that match the same traffic (same source and destination) use the Protocol Agent defined in the Continue rule. With Protocol Agents, the Continue rule affects only rules where the Service cell is set to ANY. More specific Service definitions override the Continue rule, as all Service elements specify that either some particular Protocol Agent or no Protocol Agent is used. Some protocols may require a Protocol Agent if the Connection Tracking option is enabled for the rule. Those protocols may not be allowed by a rule that has ANY as its Service unless a Protocol Agent is configured using a previous matching Continue rule.

Since Protocol Agents validate traffic against the specifics of a particular protocol, you must ensure that a Service with a Protocol Agent is not applied to traffic that does not use that protocol. Also, Protocol Agents are designed for particular types of uses, so they may not always be appropriate even if the protocol matches. See below for details of what each Protocol Agent does.

Using Protocol Agents

There are Protocol Agents for many different protocols. This section describes the available Protocol Agents and lists the configurable parameters that they add to Services that use them. When the description below states "There are no configurable parameters for this Protocol Agent", the Protocol Agent does not have any options, but may still have a control for turning the Protocol Agent on/off in the Service.

FTP Agent

One of the most common ways to transfer files across networks is using FTP. An FTP session starts with a control connection (by default, TCP port 21), and the data connection continues using a dynamically allocated port. The Protocol Agent keeps track of the actual ports used so that ports can be opened only as needed for specific connections. This way, the whole range of possible dynamic ports does not need to be allowed in the policy. The FTP Protocol is platform-independent.

This agent has parameters you can set in the Service properties.

GRE Agent

The Generic Routing Encapsulation (GRE) protocol is a tunneling protocol that allows the encapsulation of network layer packets inside IP tunneling packets. The GRE agent provides protocol inspection for tunneled GRE traffic. This agent specifies rematching parameters for GRE-encapsulated packets, and defines which traffic is tunneled. This agent has parameters you can set in the Service properties.

GTP Agent

The GPRS Tunneling Protocol (GTP) is used to carry GPRS (general packet radio service) packets in GSM, UMTS, and LTE networks. The GTP agent provides protocol inspection for GTP traffic. There are no configurable parameters for this Protocol Agent.

H.323 Agent

H.323 defines a set of protocols as well as the components and procedures for real-time multimedia communication. H.323 consists of a series of different types of standards related to video and audio services, real-time transport, control channels, security, etc.

There are no configurable parameters for this Protocol Agent.

HTTP Agent

The HTTP agent can be used to log the URLs from HTTP requests. This agent has parameters you can set in the Service properties.

HTTPS Agent

The HTTPS agent can be used for identifying encrypted HTTPS traffic for decryption and inspection in the Access rules, and for identifying encrypted HTTPS traffic for inspection in the Inspection policy. This agent has parameters you can set in the Service properties.

IPv4 Encapsulation Agent

The IPv4 Encapsulation Agent provides protocol inspection for tunneled IPv4 traffic. This Protocol Agent specifies rematching parameters for IPv4 packets encapsulated in IPv6 packets. This agent has parameters you can set in the Service properties.

IPv6 Encapsulation Agent

The IPv6 Encapsulation Agent provides protocol inspection for tunneled IPv6 traffic. This Protocol Agent specifies rematching parameters for IPv6 packets encapsulated in IPv4 packets. This agent has parameters you can set in the Service properties.

MGCP Agent

The MGCP (Media Gateway Control Protocol) agent provides support for related RTP (Real-time Transport Protocol) connections in VoIP (Voice over IP) traffic. There are no configurable parameters for this Protocol Agent.

MSRPC Agent

The MSRPC Protocol Agent is primarily meant for communications between Microsoft Outlook clients and a Microsoft Exchange server.

The supported end-point mapper (EPM) connection method between the Outlook client and the Exchange server is TCP. By default, the Microsoft RPC/EPM service is available at port 135/TCP and the communications continue using a dynamically allocated port. This Protocol Agent keeps track of the actual ports used, so that the range of dynamic ports does not need to be allowed in the policy.

There are no configurable parameters for this Protocol Agent.

NetBIOS Agent

This Protocol Agent provides deep inspection for Windows NetBIOS Datagram Service connections. This agent is also used to allow Windows NetBIOS Datagram Service connections through the Layer 2 Firewall. There are no configurable parameters for this Protocol Agent.

Oracle Agent

This Protocol Agent handles Oracle Transparent Network Substrate (TNS) protocol-based SQL*Net, Net7, and Net8 connections. It is meant for cases where TCP port 1521 is used only for negotiating the port number for Oracle database connections, and the port number for the actual connection is assigned dynamically.

This Protocol Agent is needed only if the database is located on a different computer than the Oracle listener. The Oracle Protocol Agent does not modify payload data because the database service connections may go through a different route than the listener connection. You can create custom Oracle agents with different settings when required.

RTSP Agent

The RTSP (Real Time Streaming Protocol) network control protocol is used for establishing and controlling media sessions between clients and media servers. The RTSP Protocol Agent allows RTP (Real-time Transport Protocol) and RTCP (Real-time Control Protocol) media streaming connections initiated with RTSP through the engine. This agent has parameters you can set in the Service properties.

SCCP Agent

The SCCP (Skinny Call Control Protocol) provides support for related RTP (Real-time Transport Protocol) connections in VoIP (Voice over IP) traffic. There are no configurable parameters for this Protocol Agent.

Services in Firewall Agent

This Protocol Agent is used with services running on Firewalls managed by the same Management Server as the Layer 2 Firewall or IPS engine. It is only intended for the system's internal use. There are no configurable parameters for this Protocol Agent.

Shell Agent

Remote Shell is a widely used remote management protocol. This Protocol Agent manages Remote Shell connections and RExec connections. RExec is a remote protocol with which it is possible to run commands on another computer. This agent has parameters you can set in the Service properties.

SMTP Agent

The Simple Mail Transfer Protocol (SMTP) Protocol Agent provides protocol validation and deep inspection. There are no configurable parameters for this Protocol Agent.

SSH Agent

Secure Shell (SSH) is an encrypted remote use protocol. This Protocol Agent validates the communications to make sure the protocol used really is SSH. The SSH Agent validates SSHv1 only. There are no configurable parameters for this Protocol Agent.

SunRPC Agent

The Sun Remote Procedure Call (RPC) Protocol Agent only assists the Layer 2 Firewall or IPS engine in *Portmapper* connections. It makes the handling of RPC program numbers used in the Access rules faster. Only Portmapper connections going through the Layer 2 Firewall or IPS engine are assigned this Protocol Agent. This Protocol Agent is not intended for other communications.

The SunRPC Protocol Agent collects information about RPC services by interpreting the GET PORT and DUMP PORTS requests and their respective answers. All information it collects is stored in the Portmapper cache.

When the packet filter needs to evaluate RPC matches, it consults the Portmapper cache to check if the destination of the packet has the appropriate service defined in the rule. If the cache does not have the requested information available, the packet under evaluation is not let through and a query is sent to the destination host for RPC information. The information received is stored in cache.

There are no configurable parameters for this Protocol Agent.

TCP Proxy Agent

The TCP Proxy Agent is used for TCP connections that need to be closed after a certain amount of idle time. Certain TCP-based applications do not properly handle the closing of connections, and leave them open for a long period of time, unnecessarily consuming resources. For such situations, the TCP Proxy Agent can be used to actively close the connections after a certain idle time. In addition, the TCP Proxy Agent may abort a connection if the closing of the connection does not complete in a specified period of time.

There are no configurable parameters for this Protocol Agent.

TFTP Agent

The Trivial File Transfer Protocol (TFTP) Agent performs data transfer from a server to a client using dynamically selected ports. There are no specific limits to the port range in the TFTP protocol (RFC 1350).

A TFTP Agent is attached to a UDP connection established between the client and the server. The client opens the control connection from a dynamically selected source port to the fixed destination port 69/UDP on the server. A separate UDP data connection is established between the client and the server after the client has sent a read or write command to the server. The server opens a data connection from a dynamic source port to the client’s destination port, which is the same as the one used as the source port of the control connection.

There are no configurable parameters for this Protocol Agent on the IPS.

Examples of Protocol Agent Use

The examples in this section illustrate some common uses for Protocol Agents and the general steps on how each scenario is configured.

Preventing Active Mode FTP

Company A has an FTP server that allows access from the Internet. According to company policy, the IPS engine must restrict users to passive mode FTP connections.

The administrators:

1. Create a new Service element for passive FTP.
2. Attach the FTP Protocol Agent to the Service.
3. Change active mode FTP setting to **No** in the Service properties.
4. Create an Access rule that allows users to connect to the FTP server using their custom-made Service element.
5. Refresh the policy on the IPS engine.

Logging URLs Accessed by Internal Users

Company B has decided to keep track of which web pages the employees visit. In addition to logging the connections, the administrators also want to log URLs. The access is currently controlled by an Access rule that allows all outbound connections from the internal networks to the Internet regardless of the service, so the administrators decide to add the HTTP Protocol Agent in a Continue rule.

The administrators:

1. Add the Continue rule above the existing Access rule:

Source	Destination	Service	Action
Internal Networks	Expression “NOT Local Protected Sites”	“HTTP (with URL Logging)” Service	Continue
Internal Networks	Expression “NOT Local Protected Sites”	ANY	Allow

   - Using the “NOT Local Protected Sites” expression requires the Alias “Local Protected Sites” to be configured with a translation value for the IPS engine in question.

2. Refresh the policy on the IPS engine.

Chapter 17

TLS Inspection

The TLS Inspection feature decrypts TLS connections so that they can be inspected for malicious traffic, and then re-encrypts the traffic before sending it to its destination.

The following sections are included:

Overview of TLS Inspection

HTTPS is used to secure HTTP connections. When a web browser connects to a server that uses HTTPS, the browser and the server negotiate an encryption algorithm, which is used to create the encrypted connection. The server sends a certificate that is signed by a certificate authority to authenticate its identity to the web browser.

However, the encrypted HTTPS connection can also be used to obscure web-based attacks. TLS Inspection allows you to decrypt HTTPS traffic so that it can be inspected.

Strict TCP inspection mode is automatically applied to TCP connections when TLS inspection is used. See TCP Modes for Deep Inspection (page 60) for more information.

The TLS Inspection feature consists of server protection, which inspects incoming connections to servers in the protected network, and client protection, which inspects HTTPS outgoing connections initiated by clients in the protected network.

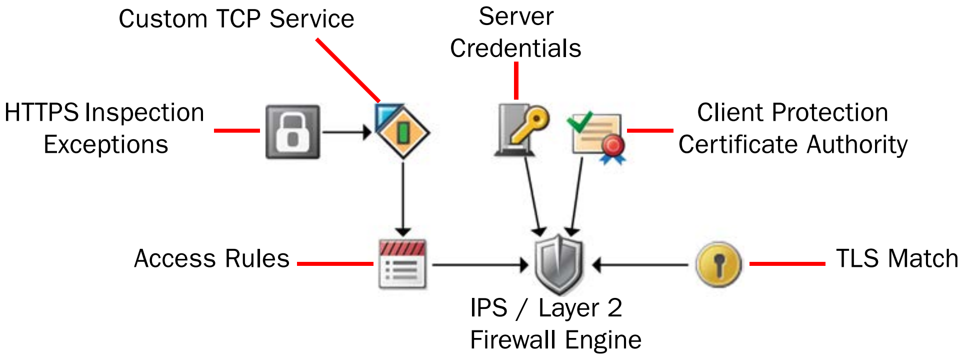
When a TLS server in the internal network is the destination of an incoming connection, the engine uses the server's credentials to decrypt and re-encrypt the traffic. When a client in the internal network initiates a connection to an external TLS server, the engine checks whether the server's certificate was signed by a certificate authority that is considered trusted. If the certificate was signed by a trusted certificate authority, the engine makes a new certificate that matches the server's certificate. From the point of view of a user in the internal network, the process is invisible: the connection is established in the same way as a connection made directly to a TLS server.

When a server's certificate is self-signed or has not been signed by a trusted certificate authority, the engine cannot trust the server certificate. In this case the engine makes a new self-signed certificate. This certificate is presented to the user in the internal network, and the user's browser shows the same warning it would show if it received a self-signed certificate directly from a TLS server. In this case, the user must decide whether or not to accept the certificate.

In both cases, the engine adds a Netscape Certificate Comment to the Extensions in the certificate to indicate that the certificate is a dynamically created certificate for SSL/TLS deep inspection. Substituting the original server certificate allows the engine to decrypt and re-encrypt the traffic.

After decrypting the traffic, normal HTTP inspection is applied, and if the traffic is allowed to continue, it is re-encrypted before forwarding it.

Illustration 17.1 Elements in TLS Inspection


Custom TCP Service
Server Credentials
HTTPS Inspection Exceptions
Client Protection Certificate Authority
Access Rules
TLS Match
IPS / Layer 2 Firewall Engine

The Server Credentials and the Client Protection Certificate Authority are specified in the properties of the engine that provides TLS Inspection. The engine uses the private key and certificate stored in the Server Credentials to decrypt traffic to and from HTTPS servers in the protected network for inspection.

The Client Protection Certificate Authority contains a private key and a certificate. The engine uses the private key stored in the Client Protection Certificate Authority to sign the certificates presented to the end-user, and the certificate to negotiate encrypted connections with TLS servers.

TLS Match elements define matching criteria for the use of the TLS protocol in traffic, and allow you to prevent specified traffic from being decrypted. TLS Matches that deny decrypting are applied globally, even if the TLS Match elements are not used in the policy.

The HTTPS Inspection Exceptions element is a list of domains that are excluded from decryption and inspection. The HTTPS Inspection Exceptions can be specified in the Protocol Parameters of a custom HTTPS Service, which is used in the Access rules to select HTTPS traffic for inspection.

The Access rules define which traffic is decrypted and inspected. You can select specific traffic for decryption and inspection, or you can enable the decryption and inspection of all TLS traffic.

Once a certificate for client and/or server protection has been uploaded to the engine, it is possible to unintentionally enable TLS decryption for all traffic in one of the following ways:

- Adding an Application that allows or requires the use of TLS to an Access rule
- Enabling the logging of Application information in the Access rules
- Enabling Deep Inspection in an Access rule with the Service cell of the rule set to ANY.

Default Elements

The **Default HTTPS Inspection Exceptions** element is an HTTPS Inspection Exceptions element that excludes domains used by the Security Management Center and engines from decryption and inspection. You cannot edit the Default HTTPS Inspection Exceptions element. If you need to make changes, you can duplicate the Default HTTPS Inspection Exceptions element and edit the copy.

The default **HTTPS (with decryption)** Service element enables the decryption of HTTPS traffic that uses the default port 443, excluding the domains that are specified in the Default HTTPS Inspection Exceptions. You cannot edit the default HTTPS (with decryption) Service element. If you need to make changes, you can duplicate the HTTPS (with decryption) Service element and edit the copy.

There are predefined Trusted Certificate Authority elements that represent the signing certificates of major certificate authorities. Default Trusted Certificate Authority elements are automatically added from dynamic update packages and cannot be edited or deleted. When client protection is used, the engine checks whether the certificate of an external server was signed by one of the Trusted Certificate Authorities. You can also create your own Trusted Certificate Authority elements to represent other certificate authorities that the engine should consider trusted.

Configuration Workflow

The following sections provide an overview to the configuration tasks. Detailed step-by-step instructions can be found in the Management Client *Online Help* and the *McAfee SMC Administrator's Guide*.

Task 1: Create Server Credentials Elements

If you want to inspect TLS traffic for which an internal server is the destination, you must create a Server Credentials element to store the private key and certificate of the server. The private key and certificate allow the engine to decrypt TLS traffic for which the internal server is the destination so that it can be inspected.

Task 2: Create Client Protection Certificate Authority Elements

If you want to inspect TLS traffic between a client in the internal network and an external server, you must create a Client Protection Certificate Authority element that contains the credentials the engine uses to sign the certificate it generates. You can import an existing private key and certificate, or generate a new private key and certificate.

You must configure users' web browsers to trust certificates signed using the credentials in the Client Protection Certificate Authority element to avoid excessive warnings or error messages about invalid certificates.

Task 3: Exclude Traffic From Decryption and Inspection

Traffic to and from some servers that use TLS may contain users' personal information that is protected by laws related to the privacy of communications. Decrypting and inspecting this traffic may be illegal in some jurisdictions. You can optionally exclude traffic from decryption and inspection in two ways: globally with a TLS Match element, or for specific matching traffic with an HTTPS Inspection Exception element.

TLS Matches define matching criteria for the use of the TLS protocol in traffic, and allow you to prevent specified traffic from being decrypted. TLS Matches that deny decrypting are applied globally, even if the TLS Match elements are not used in the policy. However, TLS Match elements that are used in specific Access rules can override globally-applied TLS matches.

In most cases, TLS Matches are the recommended way to prevent traffic from being decrypted and inspected. Globally excluding domains from decryption may also prevent some Applications from being detected in encrypted connections. In this case, you can use HTTP Inspection Exceptions exclude the domain from TLS inspection.

HTTPS Inspection Exceptions are used in a custom HTTPS service to define a list of domains for which HTTPS traffic is not decrypted. The custom HTTPS service must be used in a rule, and only traffic that matches the rule is excluded from decryption and inspection. HTTPS Inspection Exceptions are primarily intended for backwards compatibility.

Task 4: Activate TLS Inspection

In the engine properties, you specify the Client Protection Certificate Authority (if you want to inspect traffic between internal clients and external servers), and the Server Credentials (if you want to inspect traffic for which an internal server is the destination). Depending on the options you specify, you can configure only client protection, only server protection, or both client and server protection.

Note – Once a certificate for client and/or server protection has been uploaded to the engine, it is possible to unintentionally enable TLS decryption for all traffic by adding an Application that allows or requires the use of TLS to an Access rule, enabling the logging of Application information in the Access rules, or enabling Deep Inspection in an Access rule with the Service cell of the rule set to ANY.

If the default HTTPS (with decryption) Service element meets your needs, you can use the default HTTPS (with decryption) Service element in the Access rules without modification. You must create a custom HTTPS Service in the following cases:

- You want to enable decryption for HTTPS traffic that uses a different port.
- You want to select a different HTTPS Inspection Exceptions element.
- You want to log the URLs in matching traffic.
- You want to modify any of the other settings in the Service Properties.

The Access rules define which traffic is decrypted and inspected. To select specific traffic for decryption and inspection, you create Access rules that enable Deep Inspection and use a custom HTTPS Service or the default HTTPS (with decryption) Service element. To enable the decryption and inspection of all TLS traffic, you enable Deep Inspection in an Access rule with the Service cell of the rule set to ANY. Traffic that matches the Access rule is decrypted and inspected in the same way as unencrypted HTTP traffic according to the Inspection rules. See Access Rules (page 137) for more information about the Access Rules.

Using TLS Inspection

The general configuration of TLS Inspection is explained above. This section provides further information on configuring TLS Inspection.

Security Considerations

Because the HTTPS communications mediated by the engine are decrypted for inspection, and because the private keys of the servers are stored in the Server Credentials elements on the Management Server, you must carefully consider security precautions when using TLS Inspection. The following recommendations are general guidelines for ensuring the security of the engine and the SMC:

- Run the Management Server on a hardened operating system.
- Disable SSH access to the engine's command line if it is not needed regularly.
- Ensure that the engine's Control interface is in a controlled network.
- Save Management Server backups as encrypted files.

Engine Deployment

TLS Inspection requires two separate secure connections: one from the client to the engine, and one from the engine to the server. For this reason, engines must be deployed in inline mode to use TLS Inspection. TLS Inspection cannot be done for traffic picked up through Capture interfaces.

TLS inspection cannot be used on redundant single inline engines deployed alongside a Firewall cluster using dispatch clustering. In dispatch clustering, traffic is received by one node in the Firewall cluster. The node forwards the traffic to the other Firewall nodes. This can result in a situation where one of the single inline engines only receives one direction of the traffic and the other single inline engine receives both directions of the traffic. If one engine has created substitute certificates, and traffic is dispatched through a different engine without passing through the engine that created the substitute certificates, the connection fails.

For more information about engine deployment, see NGFW Deployment in IPS and Layer 2 Firewall Roles (page 29).

URL Filtering Decrypted TLS Traffic

Once TLS traffic has been decrypted, URL filtering (license-controlled feature) can be done in the same way as for regular traffic. Any traffic that is allowed to continue after URL filtering is re-encrypted and sent to its destination. For more information about how URL filtering works, see URL Filtering (page 181).

Examples of TLS Inspection

The examples in this section illustrate some common uses for TLS Inspection and general steps on how each scenario is configured.

Server Protection

Company A's web server offers HTTPS services to their customers. The administrators want to be able to detect and block attacks targeting the HTTPS server that are encrypted inside an SSL tunnel. They decide to configure TLS Inspection to decrypt and inspect traffic to and from the HTTPS server.

The administrators do the following:

1. Create a Server Credentials element and import the private key and certificate of the HTTPS server.
2. Select the Server Credentials in the engine properties.
3. Create Access rules with the default HTTPS (with decryption) Service as the Service.
4. Use the Inspection rules from the IPS Template to look for attacks in HTTP traffic.
5. Save and install the policy.

Client Protection

The administrators also want to detect and block web-based attacks targeting the web browsers of users in Company A's network to protect the workstations and internal networks. However, employees at Company A often use online banking services that are secured with HTTPS, and these connections should not be inspected. The administrators decide to configure TLS Inspection to detect and block -based attacks that are encrypted inside an SSL tunnel, and use a TLS Match element to globally exclude the online banking domains from decryption and inspection.

The administrators do the following:

1. Create Client Protection Certificate Authority elements and generate a new certificate and private key. In their network environment, the administrators add the Client Protection Certificate Authority they created to the list of trusted certificate authorities in the users' web browsers.
2. Select the Client Protection Certificate Authority in the engine properties.
3. Create a TLS Match element that prevents decryption when certificate validation succeeds for the domain names for the online banking sites that are excluded from decryption. Because the TLS Match is applied globally, the administrators do not have to use it in any specific rules.
4. Create Access rules with the default HTTPS (with decryption) Service as the Service.
5. Use the Inspection rules from the IPS Template to look for attacks in HTTP traffic.
6. Save and install the policy.

CHAPTER 18

URL FILTERING

URL filtering compares the URLs (uniform resource locators) that end-users attempt to open to a list of URLs, which can be defined manually or through pre-analyzed and categorized addresses. When a match is found, you can configure the system to respond in the various ways.

The following sections are included:

Overview of URL Filtering

URL filtering can prevent end-users from intentionally or accidentally accessing most web sites that are objectionable (based on the content they contain) or potentially harmful (for example, phishing and malware sites). This type of content filtering can increase network security and enforce an organization's policy on acceptable use of resources.

In URL filtering, the engines compare the URLs in web browser page requests against a list of forbidden URLs. There are two ways to define the forbidden URLs:

- You can define a small number of blacklisted URLs manually according to your own criteria.
- You can filter access according to a supplied URL categorization scheme (for example, filter out 'adult content').

Both methods can be used together. You can also define whitelisted URLs manually if a useful site happens to be included in a category of URLs that you otherwise want to block.

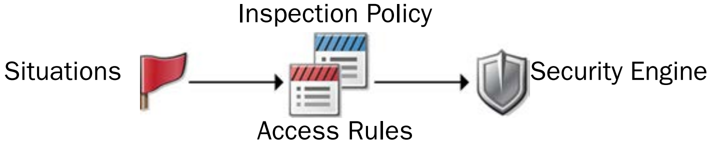
The URL categorizations are provided by the external BrightCloud service. BrightCloud provides categories for malicious sites, as well as several categories for different types of non-malicious content you may want to filter or log. Category-based filtering with BrightCloud is a license-controlled feature.

The categories allow you to configure policies based on the types of sites to block instead of manually typing in URLs. The individual URLs included in the categories are updated continuously. The engines query the actual URLs from the external URL categorization service to access up-to-date URL listings. The individual URLs are not viewable in the Management Client except when a match is found in traffic and the match is logged.

Different responses can be taken when a URL match is found: for example, you can log the matches or block the traffic. If you decide to block traffic, the engine can additionally notify the end-user with a custom message that the end-users see in their browsers instead of the page they tried to open.

When URL filtering is used, Strict TCP inspection is automatically applied to TCP connections. See TCP Modes for Deep Inspection (page 60).

Illustration 18.1 Elements in the Configuration



The URL filtering feature is configured through McAfee-supplied URL Filtering Situations and/or manual URL lists. The Access rules and the Inspection Policy define how URL Filtering Situations are matched to traffic and what kind of reaction a match triggers. URL Filtering Situations can be configured to directly override other Situations to whitelist some URLs manually (as explained further in this chapter).

Since the URLs that are included in category-based filtering are defined dynamically by an external service, it is not possible for you to manually add new categories or edit the existing ones. The URL category names are updated through dynamic update packages.

Default Elements

There are default elements for the categories you can use in URL filtering. These are represented by a specific type of Situation elements, which can be found under **Situations→By Type→URL Filtering** in the element tree and in the corresponding branch of the Rules tree in the Inspection rules.

The Context for manually defining lists of URLs is **HTTP URL Filter** (under **Protocols→Application Protocols→HTTP** when selecting a Context for a Situation).

The Situations that represent URL filtering categories have a distinctive blue color so that you can easily spot them in the rules. URL lists that you create yourself carry the standard red Situation icon.

Configuration Workflow

The following sections provide an overview of the configuration tasks. Detailed step-by-step instructions can be found in the Management Client *Online Help* and the *McAfee SMC Administrator's Guide*.

Task 1: Prepare the Engine

Category-based URL filtering requires that the engine is licensed to use the BrightCloud categorization service. You must also define DNS server addresses in the engine element so that the engines can contact the BrightCloud servers.

Task 2: Create User Response Messages

Optionally, you can define customized User Responses for URL filtering matches, such as a custom HTML page that is displayed in the end-user's browser when a connection is blocked.

Task 3: Blacklist/Whitelist Individual URLs

The HTTP URL Filter Situation Context allows you to create Situations that blacklist or whitelist URLs that you manually define. There is only one type of list for both uses. Whether a particular list is a blacklist or a whitelist depends on the action you configure for it in the Inspection Policy.

Task 4: Configure URL Filtering Rules

The Access rules and the Inspection Policy define how URL Filtering Situations are matched to traffic and what kind of reaction a match triggers.

Category-based URL filtering can be configured in the IPv4 or IPv6 Access rules, or in the Inspection rules. In the Access rules, category-based URL filtering is configured as part of the matching criteria in the Service definition. URL filtering based on URL lists can be configured in the Inspection Policy.

Different URL filtering features require you to adjust either the main Inspection Rules tree or the Exceptions. The URL Filtering branch in the Rules tree contains all category-based filters by default, making it easy to activate filtering for content categories and subcategories. Whitelists must be configured as Exceptions. Blacklists can be configured as parts of the Rules tree or as Exceptions depending on your needs. User Responses are configured in Exceptions. You can use the Continue action to set User Response options for other Exceptions and the Rules tree. See Inspection Policies (page 117) for more information on Inspection rule configuration.

The available categories may change when you activate a new dynamic update package, and be automatically enforced after the next policy upload (depending on the Rules tree settings).

Examples of URL Filtering

Allowing a Blocked URL

The company is using category-based URL filtering. Among other categories, the administrators have blocked end-users from viewing web sites categorized as “Questionable” in the Rules tree. However, now one of the network security administrators notices that they are blocked from accessing a hacker-oriented site that they have occasionally browsed to research new security threats. To make an exception for their own use, the administrators:

1. Create a new Situation called “URL Filtering Whitelist” with the Context “HTTP URL Filter” and type in the URL of the hacker site they want to access.
2. Add the following type of new Exception Rule.

Table 18.1 New Rule for Allowing a URL Above the Previously Added Category-Based Rule

Situation	Source	Destination	Action
Custom “URL Filtering Whitelist” Situation	Administrator’s workstations	ANY	Permit

Chapter 19

Applications

Application elements collect together combinations of identified characteristics and detected events in traffic to dynamically identify traffic related to the use of a particular application.

The following sections are included:

Overview of Applications

Applications are elements that provide a way to dynamically identify traffic patterns related to the use of a particular application. Applications allow you to more flexibly identify traffic beyond specifying a network protocol and ports for TCP and UDP traffic with a Service element. Matching is done based on the payload in the packets, making it possible to identify the protocol even when non-standard ports are used. Applications first identify the protocol, and then a protocol-specific pattern matching context is applied to identify the applications.

Configuration of Applications

No configuration is required to be able to use Applications in Access rules. There are several predefined Application elements available that define the criteria for matching commonly-used applications. Creating new Applications or duplicating existing elements is not recommended. If you need to override the settings of a predefined Application, you can edit the Service Definition of the rule in which you use the Application.

Default Elements

Application Type elements define general categories of applications. One Application Type can be associated with each Application. Application Types are predefined, and you cannot create new Application Types.

Tags help you to create simpler policies with less effort. Tag elements represent all Applications that are associated with that Tag. For example, the Media Tag includes several web-based image, music, and video applications. Several Tags can be associated with each Application.

TLS Match elements define matching criteria for the use of the TLS (transport layer security) protocol in traffic. When a connection that uses the TLS protocol is detected, the server certificate for the connection is compared to the TLS Match in the Application definition. TLS connections are allowed only to sites that have trusted certificates that meet the following criteria:

- The certificate domain name must match the domain name in the TLS Match element.
- The certificate must be signed by a valid certificate authority.
- The certificate must be valid (not expired or revoked).

The predefined elements are imported and updated from dynamic update packages. This means that the set of elements available in your SMC changes whenever you update your system with new definitions. The Release Notes of each dynamic update package list the new elements that the update introduces to your SMC.

Configuration Workflow

The following sections provide an overview to the configuration tasks. Detailed step-by-step instructions can be found in the Management Client *Online Help* and the *McAfee SMC Administrator's Guide*.

Task 1: Define TLS Matches

In addition to the predefined TLS Matches used in predefined Applications, you can optionally define your own TLS Matches.

TLS Matches can match traffic based on the following criteria:

- Whether certificate validation succeeded, failed, or was not performed.
- The server domain name in a valid certificate.
- Specific reasons a certificate is considered invalid if certificate validation failed.

TLS Matches also specify whether to decrypt TLS traffic to particular Internet domains for inspection. TLS Matches that deny decryption are applied globally. Even if the TLS Match element is not used in the properties of any Applications or in the Access rules, matching connections are never decrypted. Denying decryption in a TLS Match prevents Applications from being detected in encrypted connections to the specified domain(s). If the server certificate provides sufficient information to identify the Application without decrypting the client communications, you can alternatively specify that decryption is not necessary for application identification in the Application Properties.

An Application matches a TLS connection only if a TLS Match element in the Application also matches. However, TLS Matches used in Service Definitions override the TLS Match of an Application. In this case, the rule matches when the TLS Matches specified in the rule match.

Task 2: Create Access Rules

To detect application use, you must create Access rules and use an Application in the Service cell. You can either use Applications directly in the Service cell, or as part of the Service Definition. Any other criteria in the Service Definition override the Application properties. For example, the predefined Google Application matches only TCP ports 80 and 443, but using the Any TCP Service allows the Application to match any traffic pattern that resembles the use of Google regardless of the port.

Alternatively, you can use Application Types and Tags directly in the Service cell to match any of the Applications that belong to the Application Type or Tag group.

Some Applications can open several related connections. If a related connection is identified by an Access rule that detects Application use, the related connection is matched against the Access rules again. If the rule that detected the Application use has Deep Inspection enabled and the related connection matches a rule that has Deep Inspection enabled, the related connection is matched against the Inspection Policy.

Examples of Applications

The example in this section illustrates a common use for Applications and the general steps on how the scenario is configured.

Blocking Application Use

The administrators at Company A want to allow the use of HTTP in general, but block the use of social media applications from its corporate network. When social media use is detected, the administrators want to redirect users to the corporate security policy page on the company intranet.

The administrators:

1. Create a User Response to redirect dropped connections to the corporate security policy intranet page.
2. Add the following Access rules:

Source	Destination	Service	Action
Internal networks	Not internal networks expression	Social Media Application Tag	Discard Response: User Response to redirect connections to the intranet page
Internal networks	Not internal networks expression	HTTP	Allow

3. Refresh the IPS engine’s policy.

CHAPTER 20

BLACKLISTING

Blacklisting is a way to temporarily block unwanted network traffic either manually or automatically with blacklist requests from a Security Engine or Log Server. Blacklisted connections are blocked for the duration of blacklist entries, after which the connections are again allowed.

The following sections are included:

Overview of Blacklisting

Blacklisting makes it possible to block unwanted network traffic for a specified time. Engines can add entries to their own blacklists based on events in the traffic they inspect. Security Engines and Log Servers can also send blacklist requests to other Security Engines. You can also blacklist IP addresses manually.

Risks of Blacklisting

Blacklisting can have unintended consequences that could disrupt business-critical traffic. Use blacklisting with careful consideration. The following two categories represent the typical risks associated with blacklisting:

Table 20.1 Risks of Blacklisting

Risk	Explanation
Blacklisting legitimate connections (false positive)	If the defined pattern for detecting malicious traffic is inaccurate, legitimate traffic may sometimes be blacklisted. This causes service downtime for hosts that are incorrectly identified as a source of malicious traffic.
Causing self-inflicted denial-of-service (DoS)	When an attacker uses spoofed IP addresses, a different (legitimate) IP address may be blacklisted instead of the attacker’s IP address. This may cause a self-inflicted denial-of-service on legitimate traffic.

These risks can be minimized with good planning. The threats must be identified and evaluated carefully, and blacklisting must be defined only with good reasons.

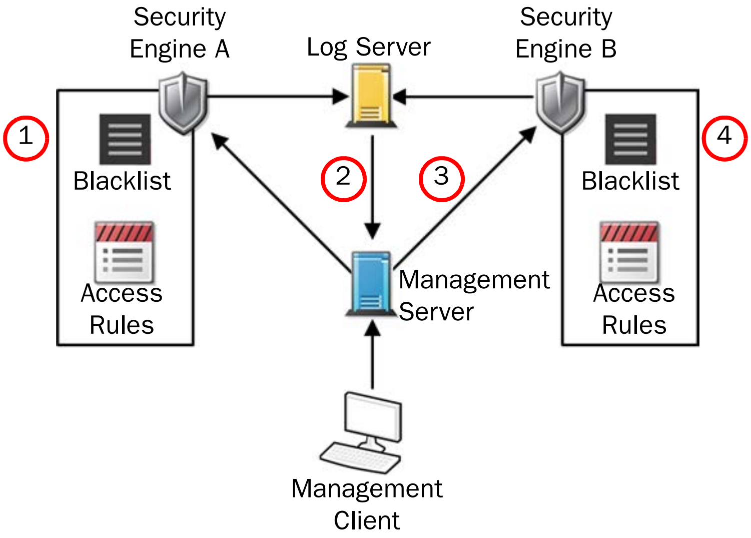
Limitations of Blacklisting

- Layer 2 Firewalls can only blacklist IPv4 traffic.
- There is no direct communication between different Virtual Security Engines or between Virtual Security Engines and the Management Server. This means that Virtual Security Engines cannot send blacklisting requests to other Virtual Security Engines, or to Security Engines.

Configuration of Blacklisting

Blacklisting is executed as defined in the Access rules of the Firewall Policy, the Layer 2 Firewall Policy, or the IPS Policy. Automatic blacklisting requests are sent as defined in the Inspection Policy.

Illustration 20.1 Blacklisting Process



1. Engines add entries to their own blacklists for traffic they inspect.
   - There is one blacklist for each Firewall, Layer 2 Firewall, IPS engine, or Virtual Security Engine.
   - In engine clusters, there is one blacklist for each cluster. The nodes in the cluster exchange blacklist information in their synchronization communications.
2. Log Servers send blacklisting requests as a response to correlation of detected events. When one Security Engine sends a blacklisting request to another Security Engine, the Log Server relays the blacklisting request to the Management Server.
3. Management Servers relay manual blacklisting commands from administrators, and blacklisting requests sent by Log Servers to the Security Engines.
4. Engines enforce the entries on their blacklists according to their Access rules.
   - Each blacklist entry exists only for a defined time period after which the entry is cleared and matching connections are again allowed. The duration of the blocking is defined when the blacklist entry is created.
   - Access rules check connections against the blacklist. If the IP addresses and ports in one of the blacklist entries match, the connection is discarded.
   - If the connection does not match a blacklisting Access rule or its related blacklist entries, the next Access rule in the policy is checked as usual.

Configuration Workflow

The following sections provide an overview of the configuration tasks. Detailed step-by-step instructions can be found in the Management Client *Online Help* and the *McAfee SMC Administrator's Guide*.

Task 1: Define Blacklisting in Access Rules

Access rules define which components are allowed to add entries to an engine's blacklist, and which connections are checked against the blacklist. Blacklisting is applied with Access rules that contain the Apply Blacklist action. Only connections that match the blacklisting Access rules are blacklisted.

No further configuration is needed if you want to blacklist connections manually. Task 2 explains the additional configuration needed for automatic blacklisting with the Inspection Policy.

Task 2: Define Exceptions in the Inspection Policy

Blacklist scope options in the Exceptions of the Inspection Policy trigger automatic blacklisting for the detected events. You can define Blacklisting scope options for any type of Exception, including rules that use Correlation Situations. Automatic blacklist entries are created using the detected event's IP source and destination addresses, and optionally the TCP or UDP ports. If the event does not contain this information, a blacklist entry cannot be created. Netmasks can optionally be used to blacklist the detected event's network.

Using Blacklisting

Manual Blacklisting

You can blacklist connections manually through the Management Client. There are three ways to create new blacklist entries manually. You can blacklist a connection found in the log data, define a new blacklist entry for a Security Engine element, or create new blacklist entries in the Blacklist view. Blacklist entries can be removed and added manually.

Monitoring Blacklisting

The currently active blacklisting entries on the engine can be monitored in the Blacklist view. Blacklist monitoring does not show you which connections are actually dropped. Blacklist monitoring only shows you the IP addresses that are currently on the blacklist. The Logs view can show which connections are actually dropped, depending on the logging options you have set. The blacklist can be sorted and filtered in the same way as log entries.

Whitelisting

Whitelisting means defining a list of IP addresses that must never be blacklisted. Whitelisting is implemented by following general Access rule design principles. Blacklisting applies only at the position of the blacklisting Access rule(s) in the policy. Connections that have already been allowed or discarded before the blacklisting rules is not affected by blacklisting. If an Access rule in the policy allows a connection, an Access rule that refers to the blacklist further down in the policy cannot blacklist the connection.

APPENDICES

In this section:

Appendix A

Command Line Tools

This appendix describes the command line tools for McAfee Security Management Center and the NGFW engines.

Note – Using the Management Client is the recommended configuration method, as most of the same tasks can be done through it.

The following sections are included:

Security Management Center Commands

Security Management Center commands include commands for the Management Server, Log Server, Web Portal Server, and Authentication Server. Most of the commands are found in the `<installation directory>/bin/` directory. In Windows, the command line tools are `*.bat` script files. In Linux, the files are `*.sh` scripts.

> **Note – If you installed the Management Server in the** `C:\Program Files\McAfee\Security Management Center` **directory in Windows, some of the program data is stored in the** `C:\ProgramData\McAfee\Security Management Center` **directory. Command line tools may be found in the** `C:\Program Files\McAfee\Security Management Center\bin` **and/or the** `C:\ProgramData\McAfee\Security Management Center\bin` **directory.**

Commands that require parameters must be run through the command line (cmd.exe in Windows). Commands that do not require parameters can alternatively be run through a graphical user interface, and may be added as shortcuts during installation.

> **Caution –** `login` **and** `password` **parameters are optional. Giving them as Command Line parameters may pose a security vulnerability. Do not enter login and password information unless explicitly prompted to do so by a Command Line tool.**

Table A.1 Security Management Center Command Line Tools

Command	Description
sgArchiveExport [**host=**<*Management Server Address* [\Domain]>] [**login**=<*login name*>] [**pass**=<*password*>] [**format**=**<*exporter format: CSV or*** *XML*>] **i=**<*input files and/or directories*> [**o=**<*output file name*>] [**f=**<*filter file name*>] [**e=**<*filter expression*>] [**-h** | **-help** | **-?**] [**-v**]	Displays or exports logs from archive. This command is only available on the Log Server. The operation checks privileges for the supplied administrator account from the Management Server to prevent unauthorized access to the logs. Enclose details in double quotes if they contain spaces.

Table A.1 Security Management Center Command Line Tools (Continued)

Command	Description
`sgArchiveExport` (*continued*)	**`Host`** specifies the address of the Management Server. If the parameter is not defined, the loopback address (localhost) is used. **`login`** defines the username for the account that is used for this operation. If this parameter is not defined, the username `root` is used. **`pass`** defines the password for the user account. **`format`** defines the file format for the output file. If this parameter is not defined, the XML format is used. **`i`** defines the source from which the logs will be exported. Can be a folder or a file. The processing recurses into subfolders. **`o`** defines the destination file where the logs will be exported. If this parameter is not defined, the output is displayed on screen. **`f`** defines a file that contains the filtering criteria you want to use for filtering the log data. You can export log filters individually in the Management Client through **Tools→Save for Command Line Tools** in the filter's right-click menu. **`e`** allows you to type in a filter expression manually (using the same syntax as exported filter files). **`-h`**, **`-help`**, or **`-?`** displays information on using the script. **`-v`** displays verbose output on the command execution. **Example** (exports logs from one full day to a file using a filter): `sgArchiveExport login=admin pass=abc123 i=c:/mcafee/security_management_center/data/archive/firewall/year2011/month12/./sgB.day01/ f=c:/mcafee/security_management_center/export/MyExportedFilter.flp format=CSV o=MyExportedLogs.csv`

`sgBackupAuthSrv`
`[`**`pwd=`**`<`*`password`*`>]`
`[`**`path=`**`<`*`destpath`*`>]`
`[`**`nodiskcheck`**`]`
`[`**`comment=`**`<`*`comment`*`>]`
`[`**`-h`**` | `**`--help`**`]`
	Creates a backup of Authentication Server user information. The backup file is stored in the `<`*`installation directory`*`>/backups/` directory. Backing up the Authentication only backs up Users, not the configuration of the Authentication Server. The Authentication Server configuration is included in the Management Server backup.
	`pwd` enables encryption.
	`path` defines the destination path.
	`nodiskcheck` ignores free disk check before creating the backup.
	`comment` allows you to enter a comment for the backup. The maximum length of a comment is 60 characters.
	`-h` or **`--help`** displays information on using the script.
	Also see **`sgRestoreAuthBackup`**.

Table A.1 Security Management Center Command Line Tools (Continued)

Command	Description
sgBackupLogSrv [**pwd=**<*password*>] [**path=**<*destpath*>] [**nodiskcheck**] [**comment=**<*comment*>] [**nofsstorage**] [**-h** | **--help**]	Creates a backup of Log Server configuration data. The backup file is stored in the <*installation directory*>/backups/ directory. Twice the size of log database is required on the destination drive. Otherwise, the operation fails. **pwd** entering a password enables encryption. **path** defines the destination path. **nodiskcheck** ignores free disk check before creating the backup. **comment** allows you to enter a comment for the backup. The maximum length of a comment is 60 characters. **nofsstorage** creates a backup only of the log server configuration without the log data. **-h** or **--help** displays information on using the script. Also see **sgRestoreLogBackup**.
sgBackupMgtSrv [**pwd=**<*password*>] [**path=**<*destpath*>] [**nodiskcheck**] [**comment=**<*comment*>] [**-h** | **--help**]	Creates a complete backup of the Management Server (including both the local configuration and the stored information in the configuration database). The backup file is stored in the <*installation directory*>/backups/ directory. Twice the size of the Management Server database is required on the destination drive. Otherwise, the operation fails. **pwd** entering a password enables encryption. **path** defines the destination path. **nodiskcheck** ignores free disk check before creating the backup. **comment** allows you to enter a comment for the backup. The maximum length of a comment is 60 characters. **-h** or **--help** displays information on using the script. Also see **sgRestoreMgtBackup** and **sgRecoverMgtDatabase**.
sgCertifyAuthSrv	Contacts the Management Server and creates a new certificate for the Authentication Server to allow secure communications with other SMC components. Renewing an existing certificate does not require changing the configuration of any other SMC components.

Table A.1 Security Management Center Command Line Tools (Continued)

Command	Description
sgCertifyLogSrv [**host=**<*Management Server Address* [\Domain]>]	Contacts the Management Server and creates a new certificate for the Log Server to allow secure communications with other SMC components. Renewing an existing certificate does not require changing the configuration of any other SMC components. **host** specifies the address of the Management Server. If the parameter is not defined, the loopback address (localhost) is used. **Domain** specifies the administrative Domain the Log Server belongs to if the system is divided into administrative Domains. If the Domain is not specified, the Shared Domain is used. The Log Server needs to be shut down before running this command. Restart the server after running this command.
sgCertifyMgtSrv	Creates a new certificate for the Management Server to allow secure communications between the SMC components. Renewing an existing certificate does not require changes on any other SMC components. The Management Server needs to be shut down before running this command. Restart the server after running this command.
sgCertifyWebPortalSrv [**host=**<*Management Server Address* [\Domain]>]	Contacts the Management Server and creates a new certificate for the Web Portal Server to allow secure communications with other SMC components. Renewing an existing certificate does not require changing the configuration of any other SMC components. **host** specifies the address of the Management Server. If the parameter is not defined, the loopback address (localhost) is used. **Domain** specifies the administrative Domain the Web Portal Server belongs to if the system is divided into administrative Domains. If the Domain is not specified, the Shared Domain is used. The Web Portal Server needs to be shut down before running this command. Restart the server after running this command.
sgChangeMgtIPOnAuthSrv <*IP address*>	Changes the Management Server's IP address in the Authentication Server's local configuration to the IP address you give as a parameter. Use this command if you change the Management Server's IP address. Restart the Authentication Server after running this command.
sgChangeMgtIPOnLogSrv <*IP address*>	Changes the Management Server's IP address in the Log Server's local configuration to the IP address you give as a parameter. Use this command if you change the Management Server's IP address. Restart the Log Server service after running this command.

Table A.1 Security Management Center Command Line Tools (Continued)

Command	Description
sgChangeMgtIPOnMgtSrv *<IP address>*	Changes the Management Server's IP address in the local configuration to the IP address you give as a parameter. Use this command if you change the Management Server's IP address. Restart the Management Server service after running this command.
sgClient	Starts a locally installed Management Client.
sgCreateAdmin	Creates an unrestricted (superuser) administrator account. The Management Server needs to be stopped before running this command.
sgExport [**host=***<Management Server Address* [\*Domain*]>] [**login**=<*login name*>] [**pass**=<*password*>] **file=***<file path and name>* [**type**=<*all|nw|ips|sv|rb|al*>] [**name**= <*element name 1, element name 2, ...*>] [**recursion**] [**-system**] [**-h** | **-help** | -?]	Exports elements stored on the Management Server to an XML file. Enclose details in double quotes if they contain spaces. **host** specifies the address of the Management Server. If the parameter is not defined, the loopback address (localhost) is used. **Domain** specifies the administrative Domain for this operation if the system is divided into administrative Domains. If the Domain is not specified, the Shared Domain is used. **login** defines the username for the account that is used for this operation. If this parameter is not defined, the username root is used. **pass** defines the password for the user account. **file** defines the name and location of the export ZIP file. **type** specifies which types of elements are included in the export file: all for all exportable elements nw for network elements ips for IPS elements sv for services rb for security policies al for alerts vpn for VPN elements. name allows you to specify by name the element(s) that you want to export. **recursion** includes referenced elements in the export, for example, the network elements used in a policy that you export. **-system** includes any system elements that are referenced by the other elements in the export. **-h**, **-help**, or **-?** displays information on using the script.

Table A.1 Security Management Center Command Line Tools (Continued)

Command	Description
sgHA [**host=**<*Management Server Address* [\Domain]>] [**login**=<*login name*>] [**pass**=<*password*>] [**master**=<*Management Server used as master server for the operation*>] [**-set-active**] [**-set-standby**] [**-check**] [**-retry**] [**-force**] [**-restart**] [**-h**|**-help**|**-?**]	Controls active and standby Management Servers. If you want to perform a full database synchronization, use the sgOnlineReplication command. **host** specifies the address of the Management Server. If the parameter is not defined, the loopback address (localhost) is used. **Domain** specifies the administrative Domain for this operation if the system is divided into administrative Domains. If the Domain is not specified, the Shared Domain is used. **login** defines the username for the account that is used for this operation. If this parameter is not defined, the username `root` is used. **pass** defines the password for the user account. **master** defines the Management Server used as a master Management Server for the operation. **-set-active** activates and locks all administrative Domains. **-set-standby** deactivates and unlocks all administrative Domains. **-check** checks that the Management Server's database is in sync with the master Management Server. **-retry** retries replication if this has been stopped due to a recoverable error. **-force** enforces the operation even if all Management Servers are not in sync. Note that using this option may cause instability if used carelessly. **-restart** restarts the specified Management Server. **-h**, **-help**, or **-?** displays information on using the script.
sgImport [**host=**<*Management Server Address* [\Domain]>] [**login**=<*login name*>] [**pass**=<*password*>] **file=**<*file path and name*> [**-replace_all**] [**-h**|**-help**|-?]	Imports Management Server database elements from an XML file. When importing, existing (non-default) elements are overwritten if both the name and type match. **host** specifies the address of the Management Server. If the parameter is not defined, the loopback address (localhost) is used. **Domain** specifies the administrative Domain for this operation if the system is divided into administrative Domains. If the Domain is not specified, the Shared Domain is used. **login** defines the username for the account that is used for this operation. If this parameter is not defined, the username `root` is used. **pass** defines the password for the user account. **file** defines the ZIP file whose contents you want to import. **-replace_all** ignores all conflicts by replacing all existing elements with new ones. **-h**, **-help**, or **-?** displays information on using the script.

Table A.1 Security Management Center Command Line Tools (Continued)

Command	Description
sgImportExportUser **[host=**<*Management Server Address* [\Domain]>] [**login**=<*login name*>] [**pass**=<*password*>] **action=**<*import*|*export*> **file=**<*file path and name*> [**-h**|**-help**|**-?**]	Imports and exports a list of Users and User Groups in an LDIF file from/to a Management Server's internal LDAP database. To import User Groups, all User Groups in the LDIF file must be directly under the **stonegate** top-level group (dc=stonegate). **The user information in the export file is stored as plaintext. Handle the file securely.** **host** specifies the address of the Management Server. If the parameter is not defined, the loopback address (localhost) is used. **Domain** specifies the administrative Domain for this operation if the system is divided into administrative Domains. If the Domain is not specified, the Shared Domain is used. **login** defines the username for the account that is used for this operation. If this parameter is not defined, the username `root` is used. **pass** defines the password for the user account. **action** defines whether users are imported or exported. **file** defines the file that is used for the operation. **Example**: `sgImportExportUser login=admin pass=abc123 action=export file=c:\temp\exportedusers.ldif` **-h**, **-help**, or **-?** displays information on using the script.
sgInfo **SG_ROOT_DIR** **FILENAME** [**fast**] [**-nolog**] [**-client**] [**-h**|**-help**|-?]	Creates a ZIP file that contains copies of configuration files and the system trace files. The resulting ZIP file is stored in the logged in user's home directory. The file location is displayed on the last line of screen output. Provide the generated file to support for troubleshooting purposes. **SG_ROOT_DIR** Security Management Center installation directory. **FILENAME** name of output file. **-nolog** extended log server information is NOT collected. **-client** collects traces only from the Management Client. **-h**, **-help**, or **-?** displays information on using the script.

Table A.1 Security Management Center Command Line Tools (Continued)

Command	Description
sgOnlineReplication [**login**=<*login name*>] [**pass**=<*password*>] [**active-server=**<*name of active Management Server*>] [**standby-server=**<*name of additional Management Server*>] [**standby-server-address=**<*IP address of additional Management Server*>] [**-nodisplay**] [**-h**|**-help**|**-?**]	Replicates the Management Server's database from the active Management Server to an additional Management Server. The Management Server to which the database is replicated must be shut down before running this command. Restart the Management Server after running this command. **Note!** Use this script to replicate the database only if the additional Management Server's configuration has been corrupted, the additional Management Server's certificate has expired, or in new SMC installations if the automatic database replication between the Management Servers has not succeeded. Otherwise, synchronize the database through the Management Client. See the *McAfee SMC Administrator's Guide* for more information. **login** defines the username for the account that is used for this operation. If this parameter is not defined, the username root is used. **pass** defines the password for the user account. **active-server** option specifies the IP address of the active Management Server from which the Management database is replicated. **standby-server** option specifies the name of the additional Management Server to which the Management database is replicated. **standby-server-address** option specifies the IP address of the additional Management Server to which the Management database is replicated. **-nodisplay** sets a text only console. **-h**, **-help**, or **-?** displays information on using the script.
sgReinitializeLogServer	**Note!** This script is located in <*installation directory*>/bin/install. Creates a new Log Server configuration if the configuration file has been lost.
sgRestoreArchive <ARCHIVE_DIR>	Restores logs from archive files to the Log Server. This command is available only on the Log Server. **ARCHIVE_DIR** is the number of the archive directory (0 – 31) from where the logs will be restored. By default, only archive directory 0 is defined. The archive directories can be defined in the <*installation directory*>/data/LogServerConfiguration.txt file: ARCHIVE_DIR_*xx*=*PATH*.

Table A.1 Security Management Center Command Line Tools (Continued)

Command	Description
sgRestoreAuthBackup [**-pwd=**<*password*>] [**-backup=**<*backup file name*>] [**-nodiskcheck**] [**-h**|**-help**]	Restores the Authentication Server user information from a backup file in the <*installation directory*>/backups/ directory. Apply the Authentication Server's configuration after this command. **-pwd** defines a password for encrypted backup. **-backup** defines a name for the backup file. **-nodiskcheck** ignores free disk check before backup restoration. **-h** or **-help** displays information on using the script.
sgRestoreLogBackup [**-pwd=**<*password*>] [**-backup=**<*backup file name*>] [**-nodiskcheck**] [**-overwrite-syslog-template**] [**-h**|**-help**]	Restores the Log Server (logs and/or configuration files) from a backup file in the <*installation directory*>/backups/ directory. Apply the Authentication Server's configuration after this command. **-pwd** defines a password for encrypted backup. **-backup** defines a name for the backup file. **-nodiskcheck** ignores free disk check before backup restoration. **-overwrite-syslog-template** overwrites a syslog template file if found in the backup. **-h** or **-help** displays information on using the script.
sgRestoreMgtBackup [**-pwd**=<*password*>] [**-backup**=<*backup file name*>] [**-nodiskcheck**] [**-h**|**-help**]	Restores the Management Server (database and/or configuration files) from a backup file in the <*installation directory*>/backups/ directory. **-pwd** defines a password for encrypted backup. **-backup** defines a name for the backup file. **-nodiskcheck** ignores free disk check before backup restoration. **-h** or **-help** displays information on using the script.
sgRevert	**Note!** This script is located in <*installation directory*>/bin/uninstall. Reverts to the previous installation saved during the upgrade process. The previous installation can be restored at any time, even after a successful upgrade.
sgShowFingerPrint	Displays the CA certificate's fingerprint on the Management Server.
sgStartAuthSrv	Starts the Authentication Server.
sgStartLogSrv	Starts the Log Server and its database.
sgStartMgtDatabase	Starts the Management Server's database. There is usually no need to use this script.

Table A.1 Security Management Center Command Line Tools (Continued)

Command	Description
sgStartMgtSrv	Starts the Management Server and its database.
sgStartWebPortalSrv	Starts the Web Portal Server.
sgStopLogSrv	Stops the Log Server.
sgStopMgtSrv	Stops the Management Server and its database.
sgStopMgtDatabase	Stops the Management Server's database. There is usually no need to use this script.
sgStopWebPortalSrv	Stops the Web Portal Server.
sgStopRemoteMgtSrv [**host=<***Management Server Host Name***>**] [**login=<***login name*>] [**pass=<***password*>] [**-h**|**-help**|**-?**]	Stops the Management Server service when run without arguments. To stop a remote Management Server service, provide the arguments to connect to the Management Server. **host** is the Management Server's host name if not localhost. **login** is an SMC administrator account for the login. **pass** is the password for the administrator account. **-h**, **-help**, or **-?** displays information on using the script.

Table A.1 Security Management Center Command Line Tools (Continued)

Command	Description
sgTextBrowser [**host=**<*Management Server address* [\Domain]>] [**login**=<*login name*>] [**pass**=<*password*>] [**format**=<*CSV*|*XML*>] [**o=**<*output file*>] [**f=**<*filter file*>] [**e=**<*filter expression*>] [**m=**<*current*|*stored*>] [**limit=**<*maximum number of unique records to fetch*>] [**-h**|**-help**|**-?**]	Displays or exports current or stored logs. This command is available on the Log Server. Enclose the file and filter names in double quotes if they contain spaces. **host** defines the address of the Management Server used for checking the login information. If this parameter is not defined, Management Server is expected to be on the same host where the script is run. If Domains are in use, you can specify the Domain the Log Server belongs to. If `domain` is not specified, the Shared Domain is used. **login** defines the username for the account that is used for this export. If this parameter is not defined, the username `root` is used. **pass** defines the password for the user account used for this operation. **format** defines the file format for the output file. If this parameter is not defined, the XML format is used. **o** defines the destination output file where the logs will be exported. If this parameter is not defined, the output is displayed on screen. **f** defines the exported filter file that you want to use for filtering the log data. **e** defines the filter that you want to use for filtering the log data. Type the name as shown in the Management Client. **m** defines whether you want to view or export logs as they arrive on the Log Server (current) or logs stored in the active storage directory (stored). If this option is not defined, the current logs are used. **limit** defines the maximum number of unique records to be fetched. The default value is unlimited. **-h**, **-help**, or **-?** displays information on using the script.

NGFW Engine Commands

The commands in the following two tables can be run on the command line on Firewall, Layer 2 Firewall, IPS engines and/or Master Engines.

Note – All command line tools that are available for single Security Engines are also available for Virtual Security Engines that have the same role. However, there is no direct access to the command line of Virtual Security Engines. Commands to Virtual Security Engines must be sent from the command line of the Master Engine using the `se-virtual-engine` command.

Table A.2 NGFW Engine Command Line Tools

Command	Engine Role	Description
`avdbfetch` [**`--dbzip`** `=`*`<path to zip file>`*] [**`--proxy`**`=`*`<proxy address>`*] [**`--proxy-pass`** `=`*`<proxy password>`*] [**`--proxy-user`** `=`*`<proxy user>`*] [**`--url`**`=`*`<url path>`*]	Firewall	If the separately-licensed anti-virus feature is enabled on a Firewall, use this command to manually update the anti-virus database. **`--dbzip`** defines the location of the locally-stored database zip file. This option can be used when there is not an internet connection and you have manually copied the database to a folder on the engine. This parameter does not need to be defined if the zip file is stored in `/var/tmp`. **`--proxy`** defines the address of an HTTP proxy if one is required to connect to the database mirror. **`--proxy-pass`** defines the password (if required) for the HTTP proxy. **`--proxy-user`** defines the username (if required) for the HTTP proxy. **`--url`** defines the address of the database mirror. If not specified, the default address is http://update.nai.com/Products/CommonUpdater.

Table A.2 NGFW Engine Command Line Tools (Continued)

Command	Engine Role	Description
sg-blacklist **show** [-v] [-f *FILENAME*] | **add [** [**-i** *FILENAME*] | [**src** *IP_ADDRESS/MASK*] [**src6** *IPv6_ADDRESS/PREFIX*] [**dst** *IP_ADDRESS/MASK*] [**dst6** *IPv6_ADDRESS/PREFIX*] [**proto** *{tcp|udp|icmp|NUM}*] [**srcport** *PORT{-PORT}*] [**dstport** *PORT{-PORT}*] [**duration** *NUM*] **]** | **del [** [**-i** *FILENAME*] | [**src** *IP_ADDRESS/MASK*] [**src6** *IPv6_ADDRESS/PREFIX*] [**dst** *IP_ADDRESS/MASK*] [**dst6** *IPv6_ADDRESS/PREFIX*] [**proto** *{tcp|udp|icmp|NUM}*] [**srcport** *PORT{-PORT}*] [**dstport** *PORT{-PORT}*] [**duration** *NUM*] **]** | **iddel** *NODE_ID ID* | **flush**	Firewall, Layer 2 Firewall, IPS	Used to view, add, or delete active blacklist entries. The blacklist is applied as defined in Access Rules. **Commands:** **show** displays the current active blacklist entries in format: engine node ID | blacklist entry ID | (internal) | entry creation time | (internal) | address and port match | originally set duration | (internal) | (internal). Use the -f option to specify a storage file to view (/data/blacklist/db_<number>). The **-v** option adds operation's details to the output. **add** creates a new blacklist entry. Enter the parameters (see below) or use the **-i** option to import parameters from a file. **del** deletes the first matching blacklist entry. Enter the parameters (see below) or use the **-i** option to import parameters from a file. **iddel** *NODE_ID ID* removes one specific blacklist entry on one specific engine. NODE_ID is the engine's ID, ID is the blacklist entry's ID (as shown by the show command). **flush** deletes all blacklist entries.

Table A.2 NGFW Engine Command Line Tools (Continued)

Command	Engine Role	Description

sg-blacklist
(*continued*)
	Firewall, Layer 2 Firewall, IPS	**Add/Del Parameters**:
		Enter at least one parameter. The default value is used for the parameters that you omit. You can also save parameters in a text file; each line in the file is read as one blacklist entry.
		src *IP_ADDRESS/MASK* defines the source IP address and netmask to match. Matches any IP address by default.
		src6 *IPv6_ADDRESS/PREFIX* defines the source IPv6 and prefix length to match. Matches any IPv6 address by default.
		dst *IP_ADDRESS/MASK* defines the destination IP address and netmask to match. Matches any IP address by default.
		dst6 *IPv6_ADDRESS/PREFIX* defines the destination IPv6 address and prefix length to match. Matches any IPv6 address by default.
		proto *{tcp|udp|icmp|NUM}* defines the protocol to match by name or protocol number. Matches all IP traffic by default.
		srcport *PORT[-PORT*] defines the TCP/UDP source port or range to match. Matches any port by default.
		dstport *PORT[-PORT]* defines the TCP/UDP destination port or range to match. Matches any port by default.
		duration *NUM* defines in seconds how long the entry is kept. Default is 0, which cuts current connections, but is not kept.
		Examples:
		`sg-blacklist add src 192.168.0.2/32 proto tcp dstport 80 duration 60`
		`sg-blacklist add -i myblacklist.txt`
		`sg-blacklist del dst 192.168.1.0/24 proto 47`

sg-bootconfig [**--primary-console** =*tty0|ttyS PORT,SPEED*] [**--secondary-console** = [*tty0|ttyS PORT,SPEED*]] [**--flavor**=*up|smp*] [**--initrd**=*yes|no*] [**--crashdump**=*yes|no|Y@X*] [**--append**=*kernel options*] [**--help**] **apply**	Firewall, Layer 2 Firewall, IPS	Used to edit boot command parameters for future bootups. **--primary-console**=*tty0|ttyS PORT,SPEED* parameter defines the terminal settings for the primary console. **--secondary-console**= [*tty0|ttyS PORT,SPEED*] parameter defines the terminal settings for the secondary console. **--flavor**=*up|smp* [*-kdb*] parameter defines whether the kernel is uniprocessor or multiprocessor. **--initrd**=*yes|no* parameter defines whether Ramdisk is enabled or disabled. **--crashdump**=*yes|no|Y@X* parameter defines whether kernel crashdump is enabled or disabled, and how much memory is allocated to the crash dump kernel (Y). The default is 24M. X must always be 16M. **--append**=*kernel options* parameter defines any other boot options to add to the configuration. **--help** parameter displays usage information. **apply** command applies the specified configuration options.

Table A.2 NGFW Engine Command Line Tools (Continued)

Command	Engine Role	Description
sg-clear-all	Firewall, Layer 2 Firewall, IPS	**Note! Use this only if you want to clear all configuration information from the engine.** This command resets all configuration information from the engine. It does not remove the engine software. After using this command, you must reconfigure the engine using the sg-reconfigure command.
sg-cluster **[-v *<virtual engine ID>*]** **[status [-c *SECONDS*]]** **[versions]** **[online]** **[lock-online]** **[offline]** **[lock-offline]** **[standby]** **[safe-offline]** **[force-offline]**	Firewall, Layer 2 Firewall, IPS	Used to display or change the status of the node. **-v *<virtual engine ID>*** (*Master Engine only*) option specifies the ID of the Virtual Security Engine on which to execute the command. **status** [**-c** *SECONDS*] command displays cluster status. When **-c** *SECONDS* is used, status is shown continuously with the specified number of seconds between updates. **version** command displays the engine software versions of the nodes in the cluster. **online** command sends the node online. **lock-online** command sends the node online and keeps it online even if another process tries to change its state. **offline** command sends the node offline. **lock-offline** command sends the node offline and keeps it offline even if another process tries to change its state. **standby** command sets an active node to standby. **safe-offline** command sets the node to offline only if there is another online node. **force-offline** command sets the node online regardless of state or any limitations. Also sets all other nodes offline.
sg-contact-mgmt	Firewall, Layer 2 Firewall, IPS	Used for establishing a trust relationship with the Management Server as part of engine installation or reconfiguration (see sg-reconfigure below). The engine contacts the Management Server using the one-time password created when the engine's initial configuration is saved.

Table A.2 NGFW Engine Command Line Tools (Continued)

Command	Engine Role	Description
sg-dynamic-routing **[start]** **[stop]** **[restart]** **[force-reload]** **[backup** *<file>*] **[restore** *<file>*] **[sample-config]** **[route-table]** **[info]**	Firewall	`start` starts the Quagga routing suite. `stop` stops the Quagga routing suite and flushes all routes made by zebra. `restart` restarts the Quagga routing suite. `force-reload` forces reload of the saved configuration. `backup` *<file>* backs up the current configuration to a compressed file. `restore` *<file>* restores the configuration from the specified file. `sample-config` creates a basic configuration for Quagga. `route-table` prints the current routing table. `info` displays the help information for the sg-dynamic-routing command, and detailed information about Quagga suite configuration with vtysh.
sg-ipsec -d [**-u** *<username[@domain]>* | **-si** *<session id>* | **-ck** *<ike cookie>* | **-tri** *<transform id>* **-ri** *<remote ip>* | **-ci** *<connection id>*]	Firewall	Deletes VPN-related information (use `vpninfo` command to view the information). Option **-d** (for delete) is mandatory. **-u** deletes the VPN session of the named VPN client user. You can enter the user account in the form <username@domain> if there are several user storage locations (LDAP domains). **-si** deletes the VPN session of a VPN client user based on session identifier. **-ck** deletes the IKE SA (Phase one security association) based on IKE cookie. **-tri** deletes the IPSEC SAs (Phase two security associations) for both communication directions based on transform identifier. **-ri** deletes all SAs related to a remote IP address in gateway-to-gateway VPNs. **-ci** deletes all SAs related to a connection identifier in gateway-to-gateway VPNs.
sg-logger **-f** *FACILITY_NUMBER* **-t** *TYPE_NUMBER* [**-e** *EVENT_NUMBER*] [**-i** *"INFO_STRING"*] [**-s**] [**-h**]	Firewall, Layer 2 Firewall, IPS	Used in scripts to create log messages with the specified properties. **-f** *FACILITY_NUMBER* parameter defines the facility for the log message. **-t** *TYPE_NUMBER* parameter defines the type for the log message. **-e** *EVENT_NUMBER* parameter defines the log event for the log message. The default is 0 (H2A_LOG_EVENT_UNDEFINED). **-i** *"INFO_STRING"* parameter defines the information string for the log message. **-s** parameter dumps information on option numbers to stdout **-h** parameter displays usage information.

Table A.2 NGFW Engine Command Line Tools (Continued)

Command	Engine Role	Description
sg-raid [**-status**] [**-add**] [**-re-add**] [**-force**] [**-help**]	Firewall, Layer 2 Firewall, IPS	Configures a new hard drive. This command is only for McAfee NGFW appliances that support RAID (Redundant Array of Independent Disks) and have two hard drives. **-status** option displays the status of the hard drive. **-add** options adds a new empty hard drive. Use **-add -force** if you want to add a hard drive that already contains data and you want to overwrite it. **-re-add** adds a hard drive that is already partitioned. This command prompts for the drive and partition for each degraded array. Use **-re-add -force** if you want to check all the arrays. **-help** option option displays usage information.
sg-reconfigure [**--boot**] [**--maybe-contact**] [**--no-shutdown**]	Firewall, Layer 2 Firewall, IPS	Used for reconfiguring the node manually. **--boot** option applies bootup behavior. Do not use this option unless you have a specific need to do so. **--maybe-contact** option contacts the Management Server if requested. This option is only available on firewall engines. **--no-shutdown** option allows you to make limited configuration changes on the node without shutting it down. Some changes may not be applied until the node is rebooted.
sg-selftest [**-d**] [**-h**]	Firewall	Runs cryptography tests on the engine. **-d** option runs the tests in debug mode. **-h** option displays usage information.
sg-status [**-l**] [**-h**]	Firewall, Layer 2 Firewall, IPS	Displays information on the engine's status. **-l** option displays all available information on engine status. **-h** option displays usage information.

Table A.2 NGFW Engine Command Line Tools (Continued)

Command	Engine Role	Description
sg-toggle-active *SHA1 SIZE* | **--force** [**--debug**]	Firewall, Layer 2 Firewall, IPS	Switches the engine between the active and the inactive partition. This change takes effect when you reboot the engine. You can use this command, for example, if you have upgraded an engine and want to switch back to the earlier engine version. When you upgrade the engine, the active partition is switched. The earlier configuration remains on the inactive partition. To see the currently active (and inactive) partition, see the directory listing of /var/run/stonegate (**ls -l /var/run/stonegate)**. The *SHA1 SIZE* option is used to verify the signature of the inactive partition before changing it to active. If you downgrade the engine, check the checksum and the size of the earlier upgrade package by extracting the signature and size files from the sg_engine_[version.build]_i386.zip file. **--debug** option reboots the engine with the debug kernel. **--force** option switches the active configuration without first verifying the signature of the inactive partition.
sg-upgrade	Firewall	Upgrades the node by rebooting from the installation DVD. Alternatively, the node can be upgraded remotely using the Management Client.
sg-version	Firewall, Layer 2 Firewall, IPS	Displays the software version and build number for the node.
se-virtual-engine **-l | --list** **-v** <*virtual engine ID*> **-e | --enter** **-E** "<*command [options]*>" **-h | --help**	Firewall (*Master Engine only)*	Used to send commands to Virtual Firewalls from the command line of the Master Engine. All commands that can be used for the Firewall role can also be used for Virtual Firewalls. **-l** or **--list** list the active Virtual Security Engines. **- v <virtual engine ID>** specifies the ID of the Virtual Security Engine on which to execute the command. **-e** or **--enter** enters the command shell for the Virtual Security Engine specified with the -v option. To exit the command shell, type exit. **-E "<***command [options]***>"** executes the specified command on the Virtual Security Engine specified with the -v option. **-h** or **--help** shows the help message for the se-virtual-engine command.

Table A.2 NGFW Engine Command Line Tools (Continued)

Command	Engine Role	Description
sginfo [**-f**] [**-d**] [**-s**] [**-p**] [**--**] [**--help**]	Firewall, Layer 2 Firewall, IPS	Gathers system information you can send to McAfee support if you are having problems. Use this command only when instructed to do so by McAfee support. **-f** option forces sgInfo even if the configuration is encrypted. **-d** option includes core dumps in the sgInfo file. **-s** option includes slapcat output in the sgInfo file. **-p** option includes passwords in the sgInfo file (by default passwords are erased from the output). **--** option creates the sgInfo file without displaying the progress **--help** option displays usage information.

The table below lists some general Linux operating system commands that may be useful in running your engines. Some commands can be stopped by pressing `Ctrl+c`.

Table A.3 General Command Line Tools on Engines

Command	Description
dmesg	Shows system logs and other information. Use the -h option to see usage.
halt	Shuts down the system.
ip	Displays IP address information. Type the command without options to see usage. **Example:** type **ip addr** for basic information on all interfaces.
ping	Tests connectivity with ICMP echo requests. Type the command without options to see usage.
ps	Reports the status of running processes.
reboot	Reboots the system.
scp	Secure copy. Type the command without options to see usage.
sftp	Secure FTP. Type the command without options to see usage.
ssh	SSH client (for opening a terminal connection to other hosts). Type the command without options to see usage.
tcpdump	Gives information on network traffic. Use the **-h** option to see usage. You can also analyze network traffic by creating tcpdump files from the Management Client with the Traffic Capture feature. See the *McAfee SMC Administrator's Guide* for more information.
top	Displays the top CPU processes taking most processor time. Use the **-h** option to see usage.
traceroute	Traces the route packets take to the specified destination. Type the command without options to see usage.
vpninfo	Displays VPN information and allows you to issue some basic commands. Type the command without options to see usage.

Server Pool Monitoring Agent Commands

You can test and monitor the Server Pool Monitoring Agents on the command line with the commands described in the table below.

Table A.4 Server Pool Monitoring Agent Commands

Command	Description
agent [-v *level*] [-c *path*] [test [*files*]] [syntax [*files*]]	(*Windows only*) Allows you to test different configurations before activating them. -v *level* Set the verbosity level. The default level is 5. Levels 6-8 are for debugging where available. -c *path* Use the specified path as the first search directory for the configuration. test [*files*] Run in the test mode - status queries do not receive a response. If you specify the files, they are used for reading the configuration instead of the default files. The output is directed to syslog or eventlog instead of the console where the command was run unless you use the -d option. syntax [*files*] Check the syntax in the configuration file. If no files are specified, the default configuration files are checked.
sgagentd [-d] [-v *level*] [-c *path*] [test [*files*]] [syntax [*files*]]	(*Linux only*) Allows you to test different configurations before activating them. -d Don't Fork as a daemon. All log messages are printed to stdout or stderr only. -v *level* Set the verbosity level. The default level is 5. Levels 6-8 are for debugging where available. -c *path* Use the specified path as the first search directory for the configuration. test [*files*] Run in the test mode - status queries do not receive a response. If you specify the files, they are used for reading the configuration instead of the default files. The output is directed to syslog or eventlog instead of the console where the command was run unless you use the -d option. syntax [*files*] Check the syntax in the configuration file. If no files are specified, the default configuration files are checked. The output is directed to syslog or eventlog instead of the console where the command was run unless you use the -d option.

Table A.4 Server Pool Monitoring Agent Commands (Continued)

Command	Description
sgmon [*status*|*info*|*proto*] [-p *port*] [-t *timeout*] [-a *id*] *host*	Sends a UDP query to the specified host and waits for a response until received, or until the timeout limit is reached. The request type can be defined as a parameter. If no parameter is given, `status` is requested. The commands are: `status` - query the status. `info` - query the agent version. `proto` - query the highest supported protocol version. `-p` *`port`* Connect to the specified port instead of the default port. `-t` *`timeout`* Set the timeout (in seconds) to wait for a response. `-a` *`id`* Acknowledge the received log messages up to the specified id. Each response message has an id, and you may acknowledge more than one message at a given time by using the id parameter. Note that messages acknowledged by sgmon will no longer appear in the firewall logs. *`host`* The IP address of the host to connect to. To get the status locally, you may give `localhost` as the host argument. This parameter is mandatory.

APPENDIX B

DEFAULT COMMUNICATION PORTS

This chapter lists the default ports used in connections between SMC components and the default ports SMC components use with external components.

The following sections are included:

Security Management Center Ports

The illustrations below present an overview to the most important default ports used in communications between the Security Management Center (SMC) components and from the SMC to external services. See the table below for a complete list of default ports.

Illustration B.1 Destination Ports for Basic Communications Within SMC

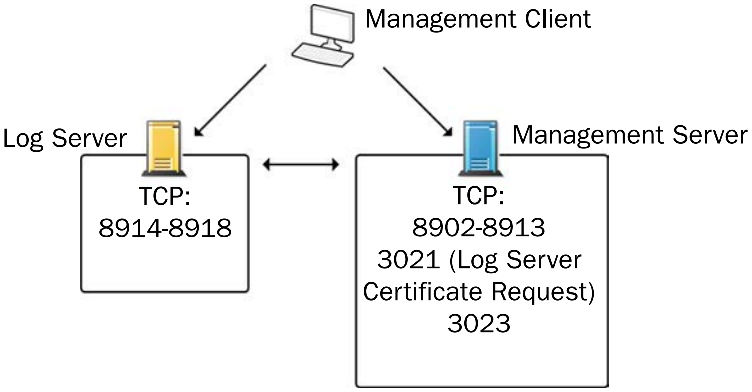
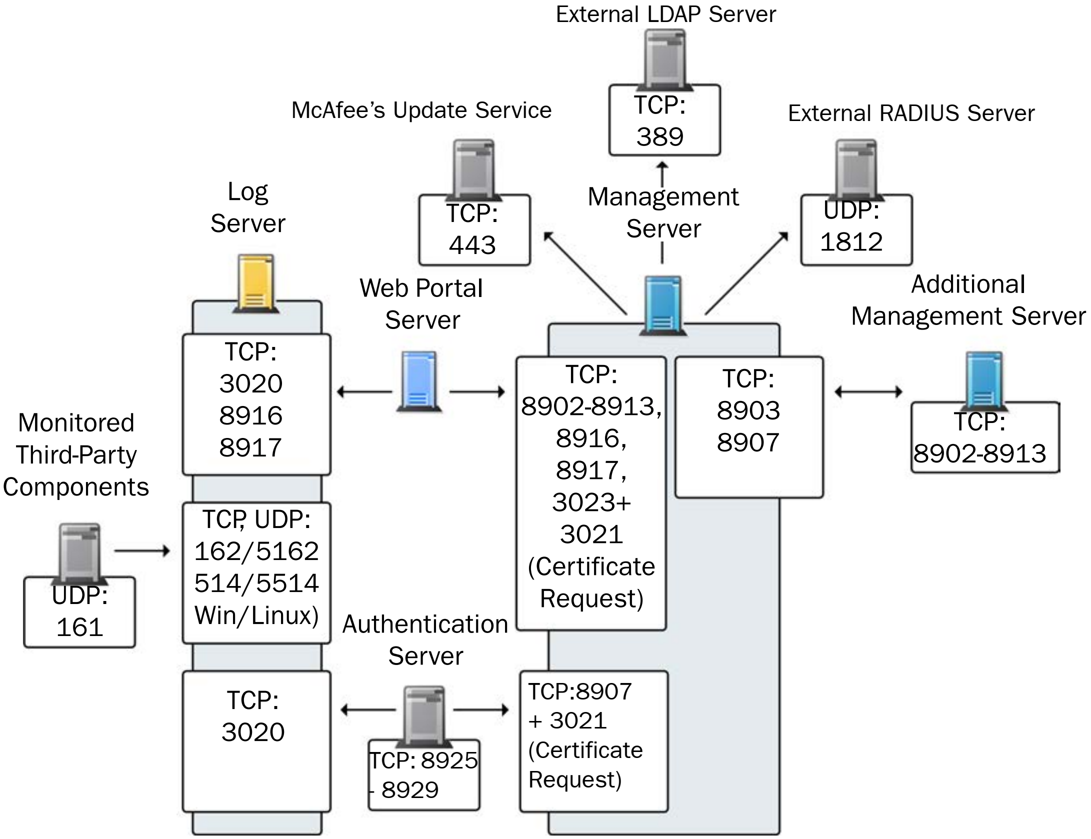


Illustration B.2 Default Destination Ports for Optional SMC Components and Features



The table below lists all default ports SMC uses internally and with external components. Many of these ports can be changed. The name of corresponding default Service elements are also included for your reference. For information on communications between SMC components and the engines, see the separate listings.

Table B.1 Security Management Center Default Ports

Listening Host	Port/ Protocol	Contacting Hosts	Service Description	Service Element Name
Additional Management Servers	8902-8913/TCP	Management Server	Database replication (push) to the additional Management Server.	SG Control
Authentication Server	8925-8929/TCP	Management Server	Security Management Server commands to Authentication Server.	SG Authentication Commands
Authentication Server node	8988-8989/TCP	Authentication Server node	Data synchronization between Authentication Server nodes.	SG Authentication Sync
DNS server	53/UDP, 53/TCP	Management Client, Management Server, Log Server	DNS queries.	DNS (UDP)
LDAP server	389/TCP	Management Server	External LDAP queries for display/ editing in the Management Client.	LDAP (TCP)
Log Server	162/UDP, 5162/UDP	Monitored third-party components	SNMPv1 trap reception from third-party components. Port 162 is used if installed on Windows, port 5162 if installed on Linux.	SNMP (UDP)
Log Server	514/TCP, 514/UDP, 5514/TCP, 5514/UDP	Monitored third-party components	Syslog reception from third-party components. Port 514 is used if installed on Windows, port 5514 if installed on Linux.	Syslog (UDP) [*Partial match*]
Log Server	2055/UDP	Monitored third-party components	NetFlow or IPFIX reception from third-party components. Port 2055 is used in both Windows and Linux.	NetFlow (UDP)
Log Server	3020/TCP	Authentication Server, Log Server, Web Portal Server, Security Engines	Alert sending from the Authentication Server, Log Server, and Web Portal Server. Log and alert messages; monitoring of blacklists, connections, status, and statistics from Security Engines.	SG Log
Log Server	8914-8918/TCP	Management Client	Log browsing.	SG Data Browsing
Log Server	8916-8917/TCP	Web Portal Server	Log browsing.	SG Data Browsing (Web Portal Server)

Table B.1 Security Management Center Default Ports (Continued)

Listening Host	Port/ Protocol	Contacting Hosts	Service Description	Service Element Name
Management Server	3021/TCP	Log Server, Web Portal Server	System communications certificate request/renewal.	SG Log Initial Contact
Management Server	8902-8913/TCP	Management Client, Log Server, Web Portal Server	Monitoring and control connections.	SG Control
Management Server	3023/TCP	Additional Management Servers, Log Server, Web Portal Server	Log Server and Web Portal Server status monitoring. Status information from an additional Management Server to the active Management Server.	SG Status Monitoring
Management Server	8903, 8907/TCP	Additional Management Servers	Database replication (pull) to the additional Management Server.	SG Control
Management Server	8907/TCP	Authentication Server	Status monitoring.	SG Control
Monitored third-party components	161/UDP	Log Server	SNMP status probing to external IP addresses.	SNMP (UDP)
RADIUS server	1812/UDP	Management Server	RADIUS authentication requests for administrator logins. The default ports can be modified in the properties of the RADIUS Server element.	RADIUS (Authentication)
SMC servers	443/TCP	Management Server	Update packages, engine upgrades, and licenses from update-pool.stonesoft.com and smc-pool.stonesoft.com.	HTTPS
Syslog server	514/UDP, 5514/UDP	Log Server	Log data forwarding to syslog servers. The default ports can be modified in the LogServerConfiguration.txt file.	Syslog (UDP) [*Partial match*]
Third-party components	2055/UDP	Log Server	NetFlow or IPFIX forwarding to third-party components. Port 2055 is used in both Windows and Linux.	NetFlow (UDP)

Security Engine Ports

The illustrations below present an overview to the most important default ports used in communications between Security Engines and the SMC and between clustered Security Engine nodes. See the table below for a complete list of default ports for the engines.

Note – Master Engines use the same default ports as clustered Security Engines. Virtual Security Engines do not communicate directly with other system components.

Illustration B.3 Destination Ports for Basic Security Engine Communications

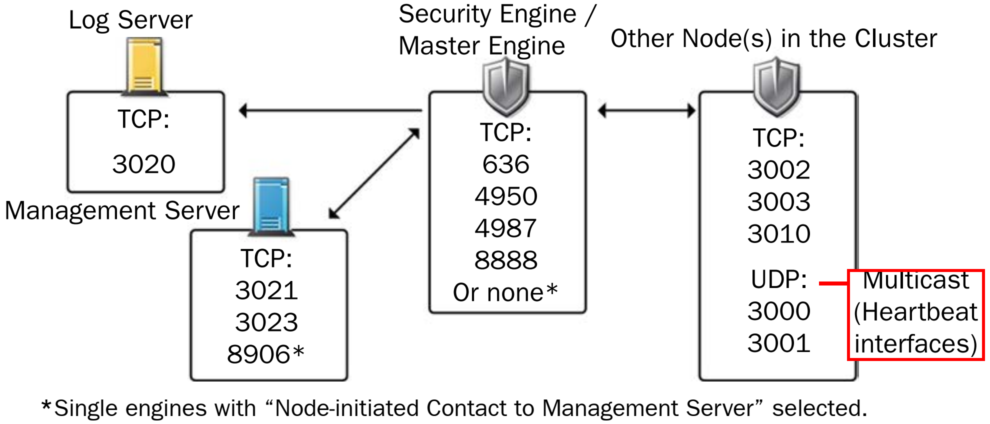
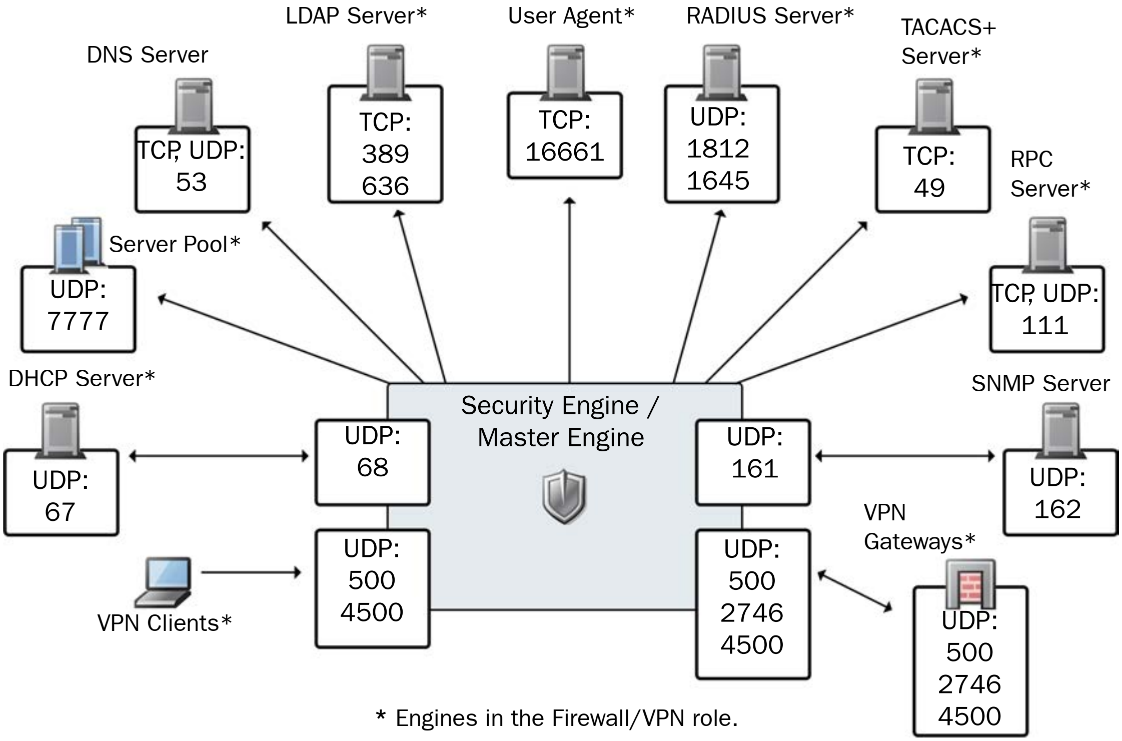


Illustration B.4 Default Destination Ports for Security Engine Service Communications



The table below lists all default ports the Security Engines use internally and with external components. Many of these ports can be changed. The name of corresponding default Service elements are also included for your reference.

Table B.2 Security Engine and Master Engine Default Ports

Listening Host	Port/Protocol	Contacting Hosts	Service Description	Service Element Name
Anti-virus signature server	80/TCP	Firewall	Anti-virus signature update service.	HTTP
Authentication Server	8925-8929/ TCP	Firewall, Master Engine	User directory and authentication services.	LDAP (TCP), RADIUS (Authentication)
BrightCloud Server	2316/TCP	Firewall, Layer 2 Firewall, IPS, Master Engine	BrightCloud URL filtering update service.	BrightCloud update
DHCP server	67/UDP	Firewall	Relayed DHCP requests and requests from a firewall that uses dynamic IP address.	BOOTPS (UDP)
DNS server	53/UDP, 53/TCP	Firewall, Master Engine	Dynamic DNS updates.	DNS (TCP)
Firewall	67/UDP	Any	DHCP relay on firewall engine.	BOOTPS (UDP)
Firewall	68/UDP	DHCP server	Replies to DHCP requests.	BOOTPC (UDP)
Firewall, Master Engine	500/UDP	VPN clients, VPN gateways	VPN negotiations, VPN traffic.	ISAKMP (UDP)
Firewall, Master Engine	636/TCP	Management Server	Internal user database replication.	LDAPS (TCP)
Firewall, Master Engine	2543/TCP	Any	User authentication (Telnet) for Access rules.	SG User Authentication
Firewall	2746/UDP	McAfee VPN gateways	UDP encapsulated VPN traffic (engine versions 5.1 and lower).	SG UDP Encapsulation
Firewall, Master Engine	4500/UDP	VPN client, VPN gateways	VPN traffic using NAT-traversal.	NAT-T
Firewall Cluster Node, Master Engine cluster node	3000-3001/ UDP 3002-3003, 3010/TCP	Firewall Cluster Node, Master Engine cluster node	Heartbeat and state synchronization between clustered Firewalls.	SG State Sync (Multicast), SG State Sync (Unicast), SG Data Sync
Firewall, Layer 2 Firewall, IPS, Master Engine	4950/TCP	Management Server	Remote upgrade.	SG Remote Upgrade

Table B.2 Security Engine and Master Engine Default Ports (Continued)

Listening Host	Port/Protocol	Contacting Hosts	Service Description	Service Element Name
Firewall, Layer 2 Firewall, IPS, Master Engine	4987/TCP	Management Server	Management Server commands and policy upload.	SG Commands
Firewall, Layer 2 Firewall, IPS	8888/TCP	Management Server	Connection monitoring for engine versions 5.1 and lower.	SG Legacy Monitoring
Firewall, Layer 2 Firewall, IPS, Master Engine	15000/TCP	Management Server, Log Server	Blacklist entries.	SG Blacklisting
Firewall, Layer 2 Firewall, IPS, Master Engine	161/UDP	SNMP server	SNMP monitoring.	SNMP (UDP)
IPS Cluster Node	3000-3001/ UDP 3002-3003, 3010/TCP	IPS Cluster Node	Heartbeat and state synchronization between clustered IPS engines.	SG State Sync (Multicast), SG State Sync (Unicast), SG Data Sync
LDAP server	389/TCP	Firewall, Master Engine	External LDAP queries, including StartTLS connections.	LDAP (TCP)
Layer 2 Firewall Cluster Node	3000-3001/ UDP 3002-3003, 3010/TCP	Layer 2 Firewall Cluster Node	Heartbeat and state synchronization between clustered Layer 2 Firewalls.	SG State Sync (Multicast), SG State Sync (Unicast), SG Data Sync
Log Server	3020/TCP	Firewall, Layer 2 Firewall, IPS, Master Engine	Log and alert messages; monitoring of blacklists, connections, status, and statistics.	SG Log
Management Server	3021/TCP	Firewall, Layer 2 Firewall, IPS, Master Engine	System communications certificate request/renewal (initial contact).	SG Initial Contact
Management Server	3023/TCP	Firewall, Layer 2 Firewall, IPS, Master Engine	Monitoring (status) connection.	SG Status Monitoring
Management Server	8906/TCP	Firewall, Layer 2 Firewall, IPS	Management connection for single engines with "Node-Initiated Contact to Management Server" selected.	SG Dynamic Control
RADIUS server	1812, 1645/ UDP	Firewall, Master Engine	RADIUS authentication requests.	RADIUS (Authentication), RADIUS (Old)

Table B.2 Security Engine and Master Engine Default Ports (Continued)

Listening Host	Port/Protocol	Contacting Hosts	Service Description	Service Element Name
RPC server	111/UDP, 111/TCP	Firewall, Master Engine	RPC number resolve.	SUNRPC (UDP), Sun RPC (TCP)
Server Pool Monitoring Agents	7777/UDP	Firewall, Master Engine	Polls to the servers' Server Pool Monitoring Agents for availability and load information.	SG Server Pool Monitoring
SNMP server	162/UDP	Firewall, Layer 2 Firewall, IPS, Master Engine	SNMP traps from the engine.	SNMP Trap (UDP)
TACACS+ server	49/TCP	Firewall, Master Engine	TACACS+ authentication requests.	TACACS (TCP)
User Agent	16661/TCP	Firewall, Master Engine	Queries for matching Users and User Groups with IP addresses.	SG Engine to User Agent
VPN gateways	500/UDP, 2746/UDP (McAfee gateways only), or 4500 UDP.	Firewall, Master Engine	VPN traffic. Ports 2746 and 4500 may be used depending on encapsulation options.	ISAKMP (UDP)

APPENDIX C

PREDEFINED ALIASES

This appendix lists the predefined Aliases in the SMC. The predefined Aliases are used in the default policies. Some of them may be useful when you create your own rules.

The following sections are included:

Predefined User Aliases

User Aliases are usually created by administrators, but there are also some predefined user aliases in the SMC. User Aliases are preceded with one $ character. The table below lists all the editable automatically created user Aliases. These Aliases are used in the firewalls' default DHCP Relay Sub-Policy.

Table C.1 System-defined User Aliases

Predefined User Alias	Description
$ DHCP address pools	Addresses that can be allocated by DHCP server(s).
$ DHCP address pools for IPsec VPN Clients	Address pools for assigning virtual IP addresses to VPN clients.
$ DHCP servers	All DHCP servers defined for the Firewall.
$ DHCP servers for IPsec VPN Clients	The DHCP servers defined for assigning virtual IP addresses to VPN clients.

System Aliases

System Aliases are automatically created non-editable Aliases. The System Aliases are preceded with two $$ characters. The table below lists the definitions of all the System Aliases. These Aliases are used in the Firewall's default policies.

Table C.2 System Aliases

System Alias	Description
$$ DHCP Enabled Interface Addresses	IP addresses (of CVIs on clusters) which have DHCP relay enabled.
$$ DHCP Enabled interface addresses for IPsec VPN clients	IP addresses (of NDIs on clusters) which have DHCP relay enabled for VPN Clients.
$$ DHCP Interface X.dns	IP address of the DHCP-assigned DNS server for interface number X.
$$ DHCP Interface X.gateways	IP address of the DHCP-assigned default router for interface number X.
$$ DHCP Interface X.ip	DHCP-assigned IP address for interface number X.
$$ DHCP Interface X.net	Network behind the dynamic IP interface number X.
$$ Interface ID X.ip	First IP address (CVI) of Physical Interface ID X.
$$ Interface ID X.net	Directly connected networks behind Physical Interface ID X.
$$ Local Cluster	All addresses of the cluster.
$$ Local Cluster (CVI addresses only)	All CVI addresses of the cluster.

Table C.2 System Aliases (Continued)

System Alias	Description
$$ Local Cluster (DHCP Interface Addresses)	All DHCP-assigned IP addresses of the engine.
$$ Local Cluster (NDI addresses only)	All NDI addresses of all nodes in the cluster.
$$ Local Cluster (NDI for heartbeat addresses only)	Heartbeat NDI addresses of all nodes in the cluster.
$$ Local Cluster (NDI for management addresses only)	Management NDI address(es) of all nodes in the cluster.
$$ Log Servers	IP addresses of all Log Servers.
$$ Management Servers	IP addresses of all Management Server.
$$ Valid DHCP Address Pools for IPsec VPN clients	Address pools defined for assigning virtual IP addresses to VPN clients.
$$ Valid DHCP Servers	All DHCP servers defined for the Firewall.
$$ Valid DHCP Servers for IPsec VPN clients	The DHCP servers defined for assigning virtual IP addresses to VPN clients.

APPENDIX D

SITUATION CONTEXT PARAMETERS

This appendix describes the parameters you can define for Situation Contexts.

Note – The details related to the Contexts in your system may be different from what is described here, because the Contexts may have been updated through dynamic update packages after this guide was published. Read the release notes of each update package you import to see which elements are affected.

The following sections are included:

Event Compress

Event Compress combines repeated similar events into the same log entry, reducing clutter in the Logs view.

Table D.1 Event Compress Parameters

Field	Option (if any)	Explanation
Correlated Situations		Situation(s) you want to compress.
Time Window		All the matches to the Situation(s) selected are combined to a common log entry when they are triggered within the defined time from each other.
Log Fields Enabled	Select	Events triggered by the selected Situations are considered the same when the values those entries have in the Log Fields you place in Lognames are identical.
	Ignore	Events triggered by the selected Situations are considered the same, except when the values those entries have in the Log Fields you place in Lognames are identical.
Lognames		The selected log fields are used by the matching option you selected in the previous step.
Location	Very Early	The execution order of the Compress operation in relation to other operations. Compress operations that share the same Location are executed in parallel; each compress operation receives the same events as the other compress operations in the same Location. “Very Early” and “Early” locations may effect the operation of other Correlations.
	Early	
	Late	
	Very Late	
Compress Filter		Filters in data for the compression.

Event Count

Event Count finds recurring patterns in traffic by counting the times certain Situations occur within the defined period, so that action can be taken if the threshold values you set are exceeded.

Table D.2 Event Count Parameters

Field	Option (if any)	Explanation
Correlated Situations		Situation(s) you want to count.
Time Window		The period of time within which the matches to the Situation must occur the specified number of times.
Alarm Threshold		The number of times that the selected Situation(s) must occur for the Correlation Situation to match.
Log Fields Enabled	Select	Events triggered by the selected Situations are considered the same when the values those entries have in the Log Fields you place in Lognames are identical.
	Ignore	Events triggered by the selected Situations are considered the same, except when the values those entries have in the Log Fields you place in Lognames are identical.
Lognames		The selected log fields are used by the matching option you selected in the previous step.

Event Group

Event Group finds event patterns in traffic by following if all events in the defined set of Situations match at least once in any order within the defined time period.

Table D.3 Event Group Parameters

Field	Option (if any)	Explanation
Member (column)	Event Match	Filter for grouping.
	Needed Number	How many occurrences of the Event selected for this Member are required for them to be included in the grouping.
	Binding	Log field used for the grouping.
Correlated Situations		Situation(s) you want to group.
Keep and Forward Events	Yes	Makes the Correlation Situation examine the events and trigger the desired response defined in the Inspection Policy but does not actually group the matching events into one. All the individual events are still available for further inspection, event though they have already triggered a response.
	No	Makes the Correlation Situation group the matching events together, so that only the response defined in the Inspection Policy is triggered, and no further processing is done on the individual events.
Time Window Size		The period of time within which the Situation must occur for them to be grouped.
Continuous Responses	Yes	Makes the Security Engine or Log Server respond as defined in the Inspection Policy to each occurrence of the defined event within the selected Time Window.
	No	Makes the Security Engine or Log Server respond only to the first occurrence of the defined event within the selected Time Window.

Event Match

Event Match allows filtering event data produced by specific Situations using Filter expressions.

Table D.4 Event Match Parameters

Field	Explanation
Correlated Situations	Situation(s) you want the Correlation Situation to match.
Filter	Filter for finding a pattern in the event data.

Event Sequence

Event Sequence finds event patterns in traffic by following if all events in the defined set of Situations match in a specific order within the defined time period.

Table D.5 Event Sequence Parameters

Field	Option (if any)	Explanation
Entry to/Exit from (columns)	Event Match	Filter for selecting data for the sequencing.
	Binding	Log field that the Correlation Situation traces to find a sequence.
Correlated Situations		Situation(s) from which you want to find sequences.
Keep and Forward Events	Yes	Makes the Correlation Situation examine the events and trigger the desired response defined in the Inspection Policy but does not actually group the matching events into one. All the individual events are still available for further inspection, event though they have already triggered a response.
	No	Makes the Correlation Situation group the matching events together, so that only the response defined in the Inspection Policy is triggered, and no further processing is done on the individual events.
Time Window Size		The period of time within which the Situation must occur for them to be considered a sequence.

Regular Expression Parameter

See Regular Expression Syntax (page 235).

Other Context Parameters

See the properties dialog of the Context in question (the Contexts are shown as branches/sub-branches in the **Other Elements→Situations→By Context** tree in the Security Engine Configuration view).

APPENDIX E

REGULAR EXPRESSION SYNTAX

This section introduces SMC regular expression syntax. Regular expressions are used in Situations for matching network traffic. Situations are used in the Inspection rules on Security Engines.

The following sections are included:

SMC Regular Expression Syntax

A regular expression is a sequence of characters that defines a matching pattern. These patterns are used for matching byte sequences in network traffic. The expression matching always starts from the beginning of the traffic stream, defined by the associated Situation Context. Depending on the context, this may mean the beginning of a TCP stream, the beginning of a UDP packet, or a protocol-specific field or header, such as the beginning of an HTTP request header or the beginning of an HTTP Request URI.

A regular expression consists of one or more branches that are separated by a logical OR symbol “|”. A Situation match occurs if any of the branches matches the traffic stream.

Illustration E.1 Example: Regular Expression Matching

```
# This regular expression matches if any of the following patterns are seen
# at the beginning of the traffic stream: "aaa", "bbb", "ccc"
aaa|bbb|ccc
```

The basic sequences that can be used in an SMC regular expression are listed in the table below:

Table E.1 SMC Regular Expression Syntax

Sequence	Description	Example
`<char>`	Matches only the defined characters.	“2”, “A”, “foo” match exactly to the defined characters: “2”, “A”, and “foo” respectively.
`.` (dot)	matches any character, including the null character `\x00` and a missing character. Matches also other than printable characters, such as the linefeed. A missing character is a special character used by the engine to represent characters missing from a TCP connection. For example, in capture mode, the engine may not see all the traffic of a TCP connection.	“.” matches any single character or byte.
`\x<hex>`	Matches the hexadecimal byte value ranging from `\x00` to \xFF.	“\x4d” matches hexadecimal value “4d” which represents the decimal value 77 and the ASCII character “M”.
`[<char>]`	Matches any single character in the list.	“[15aB]“ matches when any of the characters “1”, “5”, “a”, or “B” in the matching location of the inspected string.
`[^<char>]`	Matches any single character that is not on the list.	“[^aBc]“ matches if none of the characters “a”, “B”, or “c” is present in the matching location of the inspected string.

Table E.1 SMC Regular Expression Syntax (Continued)

Sequence	Description	Example
[<*char1*>-<*char2*>]	Matches all the characters ranging from <*char1*> to <*char2*>, these two characters included.	"[a-f]" matches any character within the range from "a" to "f", with "a" and "f" included.
\\<*char*>	Used for escaping special metacharacters to be interpreted as normal characters. The metacharacters are: \\\|)(][^-*+?.#	"\\[" matches the "[" character instead of interpreting it as the regular expression class metacharacter.
#<*text*>	Anything starting with "# " up to the linefeed (\x0a) or the carriage return (\x0d) character is considered as a comment and not used in the matching process.	"# my comment." is not used in the matching process.
(<*expr1*>|<*expr2*>)	Matches if either the expression <*expr1*> or <*expr2*> matches.	"a(bc|de)" matches "abc" and "ade".

Illustration E.2 Example Regular Expression

```
# This regular expression matches any of the following strings:
# "login.php", "login1.php", "login2.php", "login_internal.php"
# Note: to match the "." character, the character must be escaped in the
# regular expression by prefixing the character with "\"
login\.php|login[12]\.php|login_internal\.php

# Alternatively, the branches of the above regular expression can be
# combined into one single branch as shown below
login([123]|_internal)?\.php
```

It is also possible to indicate repeated, consecutive characters or regular expressions using quantifiers. The quantifiers available in SMC regular expression syntax are listed in the table below:

Table E.2 SMC Regular Expression Quantifiers

Quantifier	Description	Example
<*expr*>*	Matches if there are zero or more consecutive <*expr*> strings.	"a*" matches "<empty>", "a", "aa" and so on.
<*expr*>+	Matches if there are one or more consecutive <*expr*> strings.	"a+" matches "a", "aa", "aaa" and so on, but not the empty string.
<*expr*>?	Matches if there is zero or one <*expr*> string.	"a?" matches "<empty>" and "a".
<*expr*>{*n*,*m*}	{*num*} matches exactly *num* times the expression. {*num*,} matches *num* or more times the expression. {*num*,*max*} matches at least *num* and no more than *max* times the expression.	"a{5,}" matches five or more consecutive "a". "a{5,7}" matches five, six, or seven consecutive "a".

The quantifiers always apply only to the single previous character (or special character sequence, see Table E.3), unless otherwise indicated by parentheses. For example, the regular expression “`login*`” matches “logi”, “login” or “loginnnn”, whereas the regular expression “`(login)*`“ matches the empty string “”, “login” or “loginloginlogin”.

As the matching of a regular expression is always started from the beginning of the traffic stream, “`.*`” (any character zero or more times) is often needed when writing SMC regular expressions. For example, the regular expression “`.*/etc/passwd`” searches for the string “`/etc/passwd`” anywhere in the traffic stream.

Note – Use the wildcard characters `'*'` and `'+'`, as well as `'<expr>{n,m}'` (where m has a large value) with care. If used in the middle of a regular expression, they may result in an expression that has a very large number of matching states and that is too complex for efficient use. It is recommended to use these wildcards only in the beginning of a branch.

Special Character Sequences

Printable characters, such as “a” or “b”, are defined by simply typing them into a regular expression. In addition, there are some shorthands for common non-printable characters and character classes. Special character sequences are listed in the table below:

Table E.3 Special Character Sequences

Sequence	Description
\a	Bell (BEL) = `\x07`
\t	Horizontal tab (HT) = `\x09`
\n	Linefeed (LF) = `\x0A`
\f	Formfeed (FF) = `\x0C`
\r	Carriage return (CR) = `\x0D`
\e	Escape (ESC) = `\x1B`
\000	Octal code *000* of the character.
\x*HH*	Hexadecimal code *HH* of the character. Case-insensitive. For example, “\xaa” is considered to be the same as “\xAA”.
\c<*char*>	Control character that corresponds to Ctrl+<*char*>, where <`char`> is an uppercase letter.
\w	“word” class character = `[A-Za-z0-9_]`
\W	Non-”word” class character = `[^A-Za-z0-9_]`
\s	Whitespace character = `[ \t\r\n\f]`
\S	Non-whitespace character = `[^ \t\r\n\f]`
\d	Digit character = `[0-9]`

Table E.3 Special Character Sequences (Continued)

Sequence	Description
\D	Non-digit character = `[^0-9]`
\b	Backspace (BS) = `\x08` Note: allowed only in bracket expressions.
\Q *<expr>* **\E**	Quotes all metacharacters between the `\Q` and `\E`. Backslashes are considered as normal characters. For example, “`\QC:\file.exe\E`” matches the “`C:\file.exe`” string, not the “`C:\x0Cile.exe`” string where `\x0C` is the formfeed “`\f`”.

Illustration E.3 Example of Using Special Character Sequences

```
# This fingerprint matches HTTP content for which the length is >= 10000
# The situation context for this regular expression could be either "HTTP
# Request Header Line" or "HTTP Reply Header Line"
Content-Length: \d\d\d\d\d

# The regular expression could be also written as shown below
Content-Length: \d{5}
```

Pattern-Matching Modifiers

The regular expression syntax has Perl-like extensions. The pattern-matching modifiers are extensions that can be used to control the matching process in more detail. The modifiers are enabled with **(?***<modifiers>***)** and disabled with a minus **(?-***<modifiers>***)**, where *<modifiers>* is a list of one or more modifiers.

Illustration E.4 Example of Pattern Matching Modifiers

```
# This fingerprint is identical to the one in Illustration E.3, except for
# the (?i) modifier.
# HTTP Header names are case-insensitive. For this reason, case-
# insensitivity is enabled in this fingerprint.
(?i)Content-Length: \d\d\d\d\d
```

The modifiers `(?C)`, `(?L)`, and `(?s)`are enabled by default. The pattern-matching modifiers are listed in the table below.

Table E.4 Pattern-Matching Modifiers

Sequence	Description
(?i)	“Case insensitive mode” When enabled, case insensitive matching is used for the uppercase and lowercase letters. Thus, a letter matches regardless of its capitalization. When disabled, the letters are matched case-sensitively so that capitalization is taken into account in the matching process.

Table E.4 Pattern-Matching Modifiers (Continued)

Sequence	Description
(?s)	"Single line mode" When enabled, the dot character "." matches also the null character \x00 and a missing character in addition to matching any character (including linefeed and other non-printable characters). When disabled, the linefeed or a missing character are not matched. This modifier is enabled by default. Use **(?-s)** to disable it.
(?x)	"Extended readability mode" When enabled, equals to enabling (?C), (?L), and (?S). Comments, linefeeds and spaces are not used in the matching process, allowing to use them for readability of the expression. When disabled, equals to disabling (?C), (?L), and (?S). Comments, linefeeds and spaces are used in the matching process.
(?C)	"Allow comments mode" When enabled, anything after the hash character "# " is considered as a comment and not included in the matching process. When disabled, the hash character "# " and anything following are used in the matching process. This modifier is enabled by default. Use **(?-C)** to disable it.
(?L)	"Ignore linefeeds mode" When enabled, the linefeed and carriage return characters are not included in the matching process unless specifically defined (\x0A or \n for linefeed and \x0D or \r for carriage return). When disabled, the linefeeds and carriage returns are used in the matching process. This modifier is enabled by default. Use **(?-L)** to disable it.
(?S)	"Ignore spaces mode" When enabled, the space and horizontal tab characters are not used in the matching process unless specifically defined (\x20 for space and \x09 or \t for horizontal tab). When disabled, the space and horizontal tab characters are used in the matching process.
(?*<modifiers>***:***<expr>***)**	Applies the *<modifiers>* modifiers only to the expression *<expr>*. These modifiers are not used in other parts of the regular expression.

Variable Expression Evaluation

Variable expression evaluation is an extension to regular expression syntax that provides the ability to use variables, parse values from the traffic stream and perform arithmetic operations.

Table E.5 Variable Expression Syntax

Sequence	Description
(?[<expression>])	<expression> is one or more comma-separated expressions

Illustration E.5 Example of Setting a Variable in a Variable Expression

```
# This regular expression searches for "aaa" anywhere in the traffic stream,
# and then sets the value of "parameter1" to 1

.*aaa(?[parameter1=1])
```

The default variable size is one bit. Variable size can be changed by appending "`@<`*`size`*`>`" to the variable name. For example "`parameter1@8`" is an 8-bit variable. Possible variable sizes, in addition to 1, are 8, 16, 32 and 64 bits. By default variables are visible within a situation context. For example a variable used in a situation with context "HTTP Request URI" is visible to all other situations in that context. Prefixing the variable name with a dollar sign "`$`" makes it a connection variable. A connection variable is visible in all the situations contexts for a single TCP connection, for example in both client and server stream contexts.

By default no situation match is created when the end of a variable expression in reached. To create a match when a variable expression is used, the "`sid()`" function must be called.

Table E.6 Variable Expression Syntax

Syntax	Description
<varexpr_a> -> <varexpr_b>	varexpr_b is executed only if varexpr_a is true

The following example shows a typical case where we want to search one string followed by another, for example "aaa" followed by "bbb". An expression such as "`.*aaa.*bbb`" breaks the guideline of not using "`.*`" in the middle of a regular expression. Illustration E.6 shows how to circumvent this issue using variable expressions.

Illustration E.6 Example of Setting and Checking a Variable Value in a Variable Expression

```
# This regular expression searches for "aaa" anywhere in the traffic stream,
# and then sets the value of 'my_var' to 1.
# It also searches for "bbb", and checks whether "aaa" has already been
# seen earlier (i.e. the value of 'my_var' is one). If "aaa" has been seen
# already, a match is created using the "sid()" function.

# The following traffic matches this regular expression: "aaabbb",
# "xxaaaxxxxxxbbbxx", "aaaxbbb"
# The following traffic does not match this regular expression: "bbbaaa",
# "aabbbxxaaa"
(?x)
.*aaa(?[my_var=1]) |
.*bbb(?[my_var==1 -> sid()])
```

Illustration E.7 Example of Setting and Checking a Variable

```
# This regular expression matches when "login.php" is seen in the traffic
# stream before "user=admin"
# Situation Context e.g. "HTTP Request URI"
(?x)
.*login\.php(?[login_page_seen=1]) |
.*user=admin(?[login_page_seen==1 -> sid()])
```

All of the arithmetic operations that are available in SMC regular expressions are listed in the table below. Operator precedence is the same as in the C programming language, except that '->' is the lowest in precedence. Statements inside parentheses '()' are always evaluated first, so the order of operations can be overridden with parentheses.

Table E.7 Operations on Expression Results

Sequence	Description
`false`	Always evaluates to a false.
`true`	Always evaluates to a true.
`<number>`	A literal number in decimal, octal and hexadecimal format, for example "32" or "0x20".
`<var> = <expr>`	Sets a value returned by expression <expr> to a variable <var>. See variable syntax below.
`<var> += <expr>`	Adds the value of variable <var> with the value returned by expression <expr> and sets the result to variable <var>.
`<var> -= <expr>`	Subtracts the value from variable <var> by the value returned by expression <expr> and sets the result to variable <var>.

Table E.7 Operations on Expression Results (Continued)

Sequence	Description
<var> *= <expr>	Multiplies the value of <var> by the value returned by expression <expr> and sets the result to variable <var>.
<var> /= <expr>	Divides the value of <var> with the value returned by expression <expr> and sets the result to variable <var>.
<var> %= <expr>	Divides the value of <var> with the value returned by expression <expr> and sets the modulo of result to variable <var>.
<var> <<= <expr>	Shifts the value of <var> to left by number of steps returned by expression <expr> and sets the result to variable <var>.
<var> >>= <expr>	Shifts the value of <var> to right by number of steps returned by expression <expr> and sets the result to variable <var>.
<var> &= <expr>	Performs bitwise AND with the value of variable <var> and the value returned by expression <expr> and sets the result to variable <var>.
<var> |= <expr>	Performs bitwise OR with the value of variable <var> and the value returned by expression <expr> and sets the result to variable <var>.
<var> ^= <expr>	Performs bitwise XOR with the value of variable <var> and the value returned by expression <expr> and sets the result to variable <var>.
<expr_a> -> <expr_b>	Expression <expr_b> is evaluated only if <expr_a> is true.
<expr_a> ? <expr_b> : <expr_c>	Expression <expr_b> is evaluated only if <expr_b> is true and expression <expr_c> is evaluated if <expr_a> is false.
<expr_a> == <expr_b>	Test if expressions <expr_a> and <expr_b> return an equal value.
<expr_a> != <expr_b>	Test if expressions <expr_a> and <expr_b> do not return an equal value.
<expr_a> < <expr_b>	Test if expression <expr_b> returns higher value than expression <expr_a>.
<expr_a> <= <expr_b>	Test if expression <expr_b> returns higher or equal value than expression <expr_a>.
<expr_a> > <expr_b>	Test if expression <expr_a> returns higher value than expression <expr_b>.
<expr_a> >= <expr_b>	Test if expression <expr_a> returns higher or equal value than expression <expr_b>.
<expr_a> & <expr_b>	Performs bitwise AND with expressions <expr_a> and <expr_b> and returns the result.
<expr_a> | <expr_b>	Performs bitwise OR with expressions <expr_a> and <expr_b> and returns the result.
<expr_a> ^ <expr_b>	Performs bitwise XOR with expressions <expr_a> and <expr_b> and returns the result.
<expr_a> && <expr_b>	Performs AND with expressions <expr_a> and <expr_b> and returns the result.
<expr_a> || <expr_b>	Performs OR with if expressions <expr_a> and <expr_b> and returns the result.

Table E.7 Operations on Expression Results (Continued)

Sequence	Description
`<var>++, ++<var>`	Increase value of variable <var> by one.
`<var>--, --<var>`	Decrease value of variable <var> by one.
`-<expr>`	Negate the result of the expression <expr>.
`~<expr>`	Bitwise invert the result of the expression <expr>.
`!<expr>`	Perform NOT operation with the expression <expr>.

Note – In a regular expression such as ".*aaa(?[var1=1])", the starting of the variable expression "(?[var1=1])" is the most time-consuming operation, whereas setting or checking a variable value is a relatively fast operation. For example the regular expression ".*/(?[parameter1=1])" in an HTTP context would cause the starting of a variable expression after every "/" character in the traffic stream. As this character is very common in HTTP protocol, the regular expression might degrade the system performance.

Stream Operations

Stream operations can be used to read data from the traffic stream. The value returned by stream operations can either be written to a variable or used directly in an arithmetic operation. The stream operations are listed in the tables below:

Table E.8 ASCII Data Variable Expressions

Sequence	Description
`parse_dec(<length>)`	Parse ASCII decimal value. <length> is the maximum number of the characters to parse. The actual number of parsed digits is available in the variable $parse_length@32. If no characters could be parsed, then the variable is set to zero.
`parse_hex(<length>)`	Parse ASCII hexadecimal value. <length> is the maximum number of the characters to parse. The actual number of parsed digits is available in the variable $parse_length@32. If no characters could be parsed, then the variable is set to zero.
`parse_int(<length>)`	Parse ASCII value; parses hexadecimal if the string starts with "0x", octal if the string starts with zero ("0") and decimal otherwise. <length> is the maximum number of the characters to parse. The actual number of parsed digits is available in the variable $parse_length@32. If no characters could be parsed, then the variable is set to zero.
`parse_oct(<length>)`	Parse ASCII octal value. <length> is the maximum number of the characters to parse. The actual number of parsed digits is available in the variable $parse_length@32. If no characters could be parsed, then the variable is set to zero.

Table E.9 Miscellaneous Input Stream Operations

Sequence	Description
CRC(<length>)	Calculates a 32-bit CRC value starting from the current byte up to number of bytes specified by <length> parameter. This function can be used as a space optimizer for probabilistically matching against a specific large binary block by its CRC. The CRC used is the 32-bit CRC with polynomial 0x104C11DB7 (used for example in Ethernet).
skip(<length>)	Skip <length> number of bytes.
regex(<regexp>)	Launch independent subexpression. See section "Independent Subexpression" for more information.

Illustration E.8 Example of Parsing a Value From the Traffic Stream

```
# This regular expression finds the string "&parameter1=", parses the
# following three bytes as an ASCII decimal number, and writes the values
# to the "var1@8" variable
# The regular expression matches only if the number is greater than 100
(?x)
.*&parameter1=(?[var1@8=parse_dec(3), var1@8>100 -> sid()])
```

System Variables

System variables are connection variables whose values are set by the Security Engine. A regular expression can only read the value of these variables. The two most commonly used variables are `$dport` and `$offset`. The `$dport` variable contains the destination port of the connection/datagram, and it is useful especially in "Any Application Protocols" contexts, which receive all traffic (any TCP/UDP port), and in "Unknown Application Protocols" contexts, which receive traffic that does not have a dedicated, protocol-specific context (mostly high TCP/UDP ports). The `$offset` variable contains the number of bytes that have been matched since the beginning of the traffic stream. The table below lists all system variables.

Table E.10 System Variables

Sequence	Description
$major	The major version number of the NGFW engine.
$minor	The minor version number of the NGFW engine.
$patch	The patch level number of the NGFW engine.
$build	The build number of the NGFW engine.
$dport	The current destination port of the connection. For TCP, $dport is the destination port of the SYN packet. For UDP, $dport is the destination port of the first UDP packet sent between two hosts.

Table E.10 System Variables

Sequence	Description
$offset	The byte that is under inspection when counted from the beginning of the traffic stream. For implementation-specific reasons, the value of $offset is increased only after the first byte of a traffic stream (after the first byte, the value of $offset is still 0). For this reason, the value of $offset is actually the real offset minus one.
$parse_length@32	Number of digits parsed by last parse_dec(), parse_hex(), parse_oct() or parse_in() expression. See Stream Operations below.

Illustration E.9 Example of System Variable Use

```
# This regular expression matches if hexadecimal bytes "0x01", "0x02" and
# "0x03" are seen in port 5000
.*\x01\x02\x03(?[$dport==5000 -> sid()])
```

Independent Subexpressions

Independent subexpressions allow launching another regular expression from inside a variable expression. The function used for starting the subexpression is “regex()”. The “cancel” function must always be called after a match in a subexpression. This stops the execution of the subexpression and frees resources. The “cancel” function is always called without parentheses “()” unlike other functions.

Subexpressions are useful for splitting a single complex regular expression into two. For example ".*&filename=[^&]{256}" breaks the guideline of not using ".*" or "<expr>{n,m}" with a large m in the middle of a regular expression. The following illustration shows how to circumvent this limitation by using an independent subexpression.

Illustration E.10 Example of Independent Subexpression Use

```
# This fingerprint detects an HTTP parameter filename with value longer than
# 256 bytes
(?x)
.*&filename=(?[
    regex(
        [^&]{256}(?[sid(),cancel])
    )
])
```

Parallel Matching Groups

You can set different regular expressions to be matched in parallel groups within one Situation Context. Normally, manual Situation group definitions are not needed and the engine automatically compiles all your custom Situations in the same group (group 0). Manual group definitions are needed if the IPS policy upload fails due to fingerprint/DFA compilation problems that may occur with complex regular expressions.

To use grouping, add a new preprocessing tag to the beginning of the regular expression:

Table E.11 Preprocessing Tag for Setting a Group for Matching

Syntax	Description

`#!!GROUP(X)`
`Comment`
`#!!#`
	'X' is the group number from 0 to 7. The comment is optional. If you do not specify the group with this tag, the Situation is processed in group zero.

Tips for Working With Regular Expressions

- For more examples of regular expressions, you can view the Context tab of the Situation Properties dialog.
- When adding a new Situation to an Inspection rule, it is often useful to select the “Excerpt” logging option. This option includes an excerpt of the traffic that the regular expression matches and also the matching position (“Excerpt position”) in the log entry. This helps in verifying that the regular expression works as expected.
- Freely available tools, such as wget, can be used for generating traffic for testing regular expressions.
- If a policy upload fails with an error message such as “Fingerprint compilation failed”, it indicates that a regular expression is too complex. In this case, the regular expression should be modified. For example, use a variable expression or an independent subexpression. If it is not possible to modify the regular expression, the regular expression can be moved to a parallel matching group as explained in Parallel Matching Groups.

APPENDIX F

SNMP TRAPS AND MIBS

Firewall/VPN, IPS, and Layer 2 Firewall engines can send SNMP traps on system events. The traps are configured using SNMP Agent elements. Additionally, Tester entries can be configured to send SNMP traps. The SNMP traps are listed in the table below.

Table F.1 SNMP Traps for Firewall/VPN, IPS, and Layer 2 Firewalls

Trap Name	Objects Included	Description
fwPolicyInstall	fwSecurityPolicy	(*Firewall and Layer 2 Firewall*) Policy was installed on the Firewall engine.
ipsPolicyInstall	ipsSecurityPolicy	(*IPS*) Policy was installed on the IPS engine.
nodeBoot	-	Node bootup complete.
nodeHwmon	nodeHwmonEvent	Hardware monitoring system has detected problems.
nodeOffline	nodeOperState	Node changed to offline or standby state.
nodeOnline	nodeOperState	Node changed to online state.
nodeShutdown	-	Node is shutting down.
nodeTestFailure	nodeTestIdentity	Test subsystem reported a test failure on the node.
nodeFailedUserLogin	nodeLastLogin	(*Firewall and Layer 2 Firewall*) Login failed on the firewall engine’s console or through SSH.
nodeUserLogin	nodeLastLogin	Login initiated on the engine’s console or through SSH.
nodeUserLogout	nodeLastLogin	(*Firewall and Layer 2 Firewall*) Logout on the firewall engine’s console or through SSH.

The STONESOFT-SMI-MIB defines the top-level enterprise registrations for the NGFW products in the .iso.org.dod.internet.private.enterprises.stonesoft branch (OID `.1.3.6.1.4.1.1369`). The NGFW-specific MIB files can be downloaded at http://www.stonesoft.com/

The NGFW-specific MIBs are:

- STONESOFT-FIREWALL-MIB: see Table F.2.
- STONESOFT-IPS-MIB: see Table F.3.
- STONESOFT-NETNODE-MIB: see Table F.4.

Security Engines in the Firewall/VPN and Layer 2 Firewall roles support objects in STONESOFT-FIREWALL-MIB. Security Engines in the IPS role support objects in STONESOFT-IPS-MIB. Security Engines in all roles support objects in STONESOFT-NETNODE-MIB.

Security Engines in the Firewall/VPN role also support objects in the following standard MIBs:

- IF-MIB (RFC 2863 and RFC 2233): see Table F.5.
- IP-MIB (RFC 2011): see Table F.6.
- SNMP-USER-BASED-SM-MIB (RFC 3414): see Table F.7.
- SNMPv2 MIB (RFC 3418): see Table F.8.

Table F.2 STONESOFT-FIREWALL-MIB Objects

Object Name	Object Description in MIB
fwPolicyTime	The time when the security policy was installed to the Firewall or Layer 2 Firewall
fwSecurityPolicy	Name of the current security policy on the Firewall or Layer 2 Firewall
fwSoftwareVersion	Version string of the Firewall or Layer 2 Firewall software
fwConnNumber	Number of current connections
fwAccepted	Number of accepted packets
fwDropped	Number of dropped packets
fwLogged	Number of logged packets
fwAccounted	Number of accounted packets
fwRejected	Number of rejected packets
fwIfTable	This table contains an entry for each interface in system
fwIfStatsEntry	Row for an interface
fwIfStatsIndex	A unique value, greater than zero, for each interface or interface sub-layer in the managed system
fwIfName	Name of interface
fwIfAcceptedPkts	Number of accepted packets by Firewall or Layer 2 Firewall rules
fwIfDroppedPkts	Number of dropped packets by Firewall or Layer 2 Firewall rules
fwIfForwardedPkts	Number of forwarded packets by Firewall or Layer 2 Firewall rules
fwIfLoggedPkts	Number of logged packets by Firewall or Layer 2 Firewall rules

Table F.2 STONESOFT-FIREWALL-MIB Objects (Continued)

Object Name	Object Description in MIB
fwIfRejectedPkts	Number of rejected packets by Firewall or Layer 2 Firewall rules
fwIfAccountedPkts	Number of accounted packets by Firewall or Layer 2 Firewall rules
fwIfAcceptedBytes	Number of accepted bytes by Firewall or Layer 2 Firewall rules
fwIfForwardedBytes	Number of forwarded bytes by Firewall or Layer 2 Firewall rules
fwIfDroppedBytes	Number of dropped bytes by Firewall or Layer 2 Firewall rules
fwIfLoggedBytes	Number of logged bytes by Firewall or Layer 2 Firewall rules
fwIfRejectedBytes	Number of rejected bytes by Firewall or Layer 2 Firewall rules
fwIfAccountedBytes	Number of accounted bytes by Firewall or Layer 2 Firewall rules
fwCpuTable	This table contains an entry for each CPU in a system and total usage of all CPUs
fwCpuStats	Row with information about CPU usage
fwCpuStatsId	A unique value, greater than zero, for each CPU in the managed system. First element with Id '0' is designed for total values
fwCpuName	Name of data current line concern
fwCpuTotal	The total CPU load percentage
fwCpuUser	The percentage of time the CPU has spent running users' processes that are not niced
fwCpuSystem	The percentage of time the CPU has spent running the kernel and its processes
fwCpuNice	The percentage of time the CPU has spent running user’s processes that have been niced
fwCpuIdle	The percentage of time the CPU was idle
fwCpuIoWait	The percentage of time the CPU has been waiting for I/O to complete
fwCpuHwIrq	The percentage of time the CPU has been servicing hardware interrupts
fwCpuSoftIrq	The percentage of time the CPU has been servicing software interrupts
fwSwapBytesTotal	Total swap space
fwSwapBytesUsed	Used space of swap
fwSwapBytesUnused	Amount of unused space of swap
fwMemBytesTotal	Number of available bytes of physical memory
fwMemBytesUsed	Amount of memory being in use
fwMemBytesUnused	Amount of unused bytes of physical memory
fwMemBytesBuffers	Amount of memory used as buffers
fwMemBytesCached	Amount of memory used as cache

Table F.2 STONESOFT-FIREWALL-MIB Objects (Continued)

Object Name	Object Description in MIB
fwDiskSpaceUsageTable	Table contains an entry for each partition mounted in a system
fwDiskStats	Row of information concerning one partition
fwPartitionIndex	A unique value, greater than zero, for each partition
fwPartitionDevName	A unique name of a device
fwMountPointName	Name of a mount point
fwPartitionSize	Total size of the partition
fwPartitionUsed	Amount of used space of the partition (in kilobytes)
fwPartitionAvail	Information about amount of free space on partition (in kilobytes)
fwVpnEp4Local	Local IPv4 end-point address
fwVpnEp4Remote	Remote IPv4 end-point address
fwVpnEp4RemoteType	The type of remote VPN end-point (static, dynamic or mobile)
fwVpnEp4ReceivedBytes	Number of received bytes between the end-point pair
fwVpnEp4SentBytes	Number of sent bytes between the end-point pair
fwVpnEp4IpsecSa	Number of currently established IPsec SAs between the end-point pair
fwVpnEp6Local	Local IPv6 end-point address
fwVpnEp6Remote	Remote IPv6 end-point address
fwVpnEp6RemoteType	The type of remote VPN end-point (static, dynamic or mobile)
fwVpnEp6ReceivedBytes	Number of received bytes between the end-point pair
fwVpnEp6SentBytes	Number of sent bytes between the end-point pair
fwVpnEp6IpsecSa	Number of currently established IPsec SAs between the end-point pair
adslModulation	Modulation protocol
adslChannel	Channel type
adslConnStatus	The status of the DSL link or communication status with DSL modem in case of communication error
adslConnUptime	Uptime of current ADSL connection
adslLineStatus	Current status of DSL line
adslInOctets	Number of bytes received by ADSL interface
adslOutOctets	Number of bytes transmitted by ADSL interface
adslSynchroSpeedUp	The actual rate at which data is flowing upstream

Table F.2 STONESOFT-FIREWALL-MIB Objects (Continued)

Object Name	Object Description in MIB
adslSynchroSpeedDown	The actual rate at which data is flowing downstream
adslAttenuationUp	An estimate of the average loop attenuation upstream
adslAttenuationDown	An estimate of the average loop attenuation downstream
adslNoiseMarginUp	This is a signal-to-noise ratio (SNR) margin for traffic going upstream
adslNoiseMarginDown	This is a signal-to-noise ratio (SNR) margin for traffic going downstream
adslHecErrorsUp	The total number of header error checksum errors upstream
adslHecErrorsDown	The total number of header error checksum errors downstream
adslOcdErrorsUp	The number of out-of-cell delineation errors upstream
adslOcdErrorsDown	The number of out-of-cell delineation errors downstream
adslLcdErrorsUp	The total of lost-cell-delineation errors upstream
adslLcdErrorsDown	The total of lost-cell-delineation errors downstream
adslBitErrorsUp	The number of bit errors upstream
adslBitErrorsDown	The number of bit errors downstream

Table F.3 STONESOFT IPS-MIB Objects

Object Name	Object Description in MIB
ipsPolicyTime	The time when the security policy was installed to the IPS engine
ipsSecurityPolicy	Name of the current security policy on the IPS engine
ipsSoftwareVersion	Version string of the IPS software

Table F.4 STONESOFT-NETNODE-MIB Objects

Object Name	Object Description in MIB
nodeClusterId	The identification number of the cluster this node belongs to
nodeCPULoad	The CPU load percentage on the node
nodeHwmonEvent	Reason for the hardware monitoring event
nodeLastLogin	The most recent login event on the node
nodeLastLoginTime	Timestamp of the most recent login event on the node
nodeMemberId	Node's member identification within the cluster
nodeOperState	The operative (clustering) state of the node
nodeTestIdentity	Identification string of a nodeTest

Table F.4 STONESOFT-NETNODE-MIB Objects (Continued)

Object Name	Object Description in MIB
nodeTestResult	The most recent result of the nodeTest
nodeTestResultTime	The timestamp of the most recent result of the nodeTest

Table F.5 IF-MIB Supported Objects

Object Name	Object Description in MIB
ifAdminStatus	The desired state of the interface. The testing(3) state indicates that no operational packets can be passed. When a managed system initializes, all interfaces start with ifAdminStatus in the down(2) state. As a result of either explicit management action or per configuration information retained by the managed system, ifAdminStatus is then changed to either the up(1) or testing(3) states (or remains in the down(2) state).
ifAlias	This object is an 'alias' name for the interface as specified by a network manager, and provides a non-volatile 'handle' for the interface. On the first instantiation of an interface, the value of ifAlias associated with that interface is the zero-length string. As and when a value is written into an instance of ifAlias through a network management set operation, then the agent must retain the supplied value in the ifAlias instance associated with the same interface for as long as that interface remains instantiated, including across all re- initializations or reboots of the network management system, including those which result in a change of the interface's ifIndex value. An example of the value which a network manager might store in this object for a WAN interface is the (Telco's) circuit number or identifier of the interface. Some agents may support write-access only for interfaces having particular values of ifType. An agent which supports write access to this object is required to keep the value in non-volatile storage, but it may limit the length of new values depending on how much storage is already occupied by the current values for other interfaces.
ifDescr	A textual string containing information about the interface. This string includes the name of the manufacturer, the product name and the version of the interface hardware or software.
ifHCInMulticastPkts	The 64-bit wide number of packets, delivered by this sub-layer to a higher (sub-)layer, which were addressed to a multicast address at this sub-layer. For a MAC layer protocol, this includes both Group and Functional addresses. This object is a 64-bit version of ifInMulticastPkts. Discontinuities in the value of this counter can occur at re-initialization of the network management system, and at other times as indicated by the value of ifCounterDiscontinuityTime. The 32-bit ifInMulticastPkts reports the low 32-bits of this counter’s value.
ifHCInOctets	The 64-bit wide total number of octets received on the interface, including framing characters. This object is a 64-bit version of ifInOctets. Discontinuities in the value of this counter can occur at re-initialization of the network management system, and at other times as indicated by the value of ifCounterDiscontinuityTime. The 32-bit ifInOctets reports the low 32-bits of this counter’s value.

Table F.5 IF-MIB Supported Objects (Continued)

Object Name	Object Description in MIB
ifHCInUcastPkts	The 64-bit wide number of packets, delivered by this sub-layer to a higher (sub-)layer, which were not addressed to a multicast or broadcast address at this sub-layer. This object is a 64-bit version of ifInUcastPkts. Discontinuities in the value of this counter can occur at re-initialization of the network management system, and at other times as indicated by the value of ifCounterDiscontinuityTime. The 32-bit ifInUcastPkts reports the low 32-bits of this counter's value.
ifHCOutOctets	The 64-bit wide total number of octets transmitted out of the interface, including framing characters. This object is a 64-bit version of ifOutOctets. Discontinuities in the value of this counter can occur at re-initialization of the network management system, and at other times as indicated by the value of ifCounterDiscontinuityTime. The 32-bit ifOutOctets reports the low 32-bits of this counter's value.
ifHCOutUcastPkts	The 64-bit wide total number of packets that higher-level protocols requested be transmitted, and which were not addressed to a multicast or broadcast address at this sub-layer, including those that were discarded or not sent. This object is a 64-bit version of ifOutUcastPkts. Discontinuities in the value of this counter can occur at re-initialization of the network management system, and at other times as indicated by the value of ifCounterDiscontinuityTime. The 32-bit ifOutUcastPkts reports the low 32-bits of this counter's value.
ifHighSpeed	An estimate of the interface's current bandwidth in units of 1,000,000 bits per second. If this object reports a value of ‘n' then the speed of the interface is somewhere in the range of ‘n-500,000' to ‘n+499,999'. For interfaces which do not vary in bandwidth or for those where no accurate estimation can be made, this object contains the nominal bandwidth. For a sub-layer which has no concept of bandwidth, this object must be zero.
ifIndex	A unique value, greater than zero, for each interface. It is recommended that values are assigned contiguously starting from 1. The value for each interface sub-layer must remain constant at least from one re-initialization of the entity's network management system to the next re- initialization.
ifInDiscards	The number of inbound packets which were chosen to be discarded even though no errors had been detected to prevent their being deliverable to a higher-layer protocol. One possible reason for discarding such a packet could be to free up buffer space. Discontinuities in the value of this counter can occur at re-initialization of the network management system, and at other times as indicated by the value of ifCounterDiscontinuityTime.
ifInErrors	For packet-oriented interfaces, the number of inbound packets that contained errors preventing them from being deliverable to a higher-layer protocol. For character-oriented or fixed-length interfaces, the number of inbound transmission units that contained errors preventing them from being deliverable to a higher-layer protocol. Discontinuities in the value of this counter can occur at re-initialization of the network management system, and at other times as indicated by the value of ifCounterDiscontinuityTime.

Table F.5 IF-MIB Supported Objects (Continued)

Object Name	Object Description in MIB
ifInMulticastPkts	The 32-bit wide number of packets, delivered by this sub-layer to a higher (sub-)layer, which were addressed to a multicast address at this sub-layer. For a MAC layer protocol, this includes both Group and Functional addresses. Discontinuities in the value of this counter can occur at re-initialization of the network management system, and at other times as indicated by the value of ifCounterDiscontinuityTime. This object reports the low 32-bits of the 64-bit ifHCInMulticastPkts counter's value.
ifInOctets	The 32-bit wide total number of octets received on the interface, including framing characters. Discontinuities in the value of this counter can occur at re-initialization of the network management system, and at other times as indicated by the value of ifCounterDiscontinuityTime. This object reports the low 32-bits of the 64-bit ifHCInOctets counter's value.
ifInUcastPkts	The 32-bit wide number of packets, delivered by this sub-layer to a higher (sub-)layer, which were not addressed to a multicast or broadcast address at this sub-layer. Discontinuities in the value of this counter can occur at re-initialization of the network management system, and at other times as indicated by the value of ifCounterDiscontinuityTime. This object reports the low 32-bits of the 64-bit ifHCInUcastPkts counter's value.
ifLastChange	The value of sysUpTime at the time the interface entered its current operational state. If the current state was entered prior to the last re-initialization of the local network management subsystem, then this object contains a zero value.
ifLinkUpDownTrapEnable	Indicates whether linkUp or linkDown traps are generated for this interface. By default, this object must have the value enabled(1) for interfaces which do not operate on 'top' of any other interface (as defined in the ifStackTable), and disabled(2) otherwise.
ifMtu	The size of the largest packet which can be sent or received on the interface, specified in octets. For interfaces that are used for transmitting network datagrams, this is the size of the largest network datagram that can be sent on the interface.
ifName	The textual name of the interface. The value of this object must be the name of the interface as assigned by the local device and must be suitable for use in commands entered at the device's ‘console'. This might be a text name, such as ‘le0' or a simple port number, such as ‘1', depending on the interface naming syntax of the device. If several entries in the ifTable together represent a single interface as named by the device, then each will have the same value of ifName. Note that for an agent which responds to SNMP queries concerning an interface on some other (proxied) device, then the value of ifName for such an interface is the proxied device's local name for it. If there is no local name, or this object is otherwise not applicable, then this object contains a zero-length string.
ifNumber	The number of network interfaces (regardless of their current state) present on this system.

Table F.5 IF-MIB Supported Objects (Continued)

Object Name	Object Description in MIB
ifOperStatus	The current operational state of the interface. The testing(3) state indicates that no operational packets can be passed. If ifAdminStatus is down(2) then ifOperStatus is down(2). If ifAdminStatus is changed to up(1) then ifOperStatus changes to up(1) if the interface is ready to transmit and receive network traffic; it changes to dormant(5) if the interface is waiting for external actions (such as a serial line waiting for an incoming connection); it remains in the down(2) state if and only if there is a fault that prevents it from going to the up(1) state; it remains in the notPresent(6) state if the interface has missing (typically, hardware) components.
ifOutDiscards	The number of outbound packets which were chosen to be discarded even though no errors had been detected to prevent their being transmitted. One possible reason for discarding such a packet could be to free up buffer space. Discontinuities in the value of this counter can occur at re-initialization of the network management system, and at other times as indicated by the value of ifCounterDiscontinuityTime.
ifOutErrors	For packet-oriented interfaces, the number of outbound packets that could not be transmitted because of errors. For character-oriented or fixed-length interfaces, the number of outbound transmission units that could not be transmitted because of errors. Discontinuities in the value of this counter can occur at re-initialization of the network management system, and at other times as indicated by the value of ifCounterDiscontinuityTime.
ifOutOctets	The 32-bit wide total number of octets transmitted out of the interface, including framing characters. Discontinuities in the value of this counter can occur at re-initialization of the network management system, and at other times as indicated by the value of ifCounterDiscontinuityTime. This object reports the low 32-bits of the 64-bit ifHCOutOctets counter’s value.
ifOutUcastPkts	The 32-bit wide total number of packets that higher-level protocols requested be transmitted, and which were not addressed to a multicast or broadcast address at this sub-layer, including those that were discarded or not sent. Discontinuities in the value of this counter can occur at re-initialization of the network management system, and at other times as indicated by the value of ifCounterDiscontinuityTime. This object reports the low 32-bits of the 64-bit ifHCOutUcastPkts counter’s value.
ifPhysAddress	The interface's address at its protocol sub-layer. For example, for an 802.x interface, this object normally contains a MAC address. The interface's media-specific MIB must define the bit and byte ordering and the format of the value of this object. For interfaces that do not have such an address (for example, a serial line), this object must contain an octet string of zero length.
ifPromiscuousMode	This object has a value of false(2) if this interface only accepts packets or frames that are addressed to this station. This object has a value of true(1) when the station accepts all packets or frames transmitted on the media. The value true(1) is only legal on certain types of media. If legal, setting this object to a value of true(1) may require the interface to be reset before becoming effective. The value of ifPromiscuousMode does not affect the reception of broadcast and multicast packets or frames by the interface.

Table F.5 IF-MIB Supported Objects (Continued)

Object Name	Object Description in MIB
ifSpeed	An estimate of the interface's current bandwidth in bits per second. For interfaces which do not vary in bandwidth or for those where no accurate estimation can be made, this object must contain the nominal bandwidth. If the bandwidth of the interface is greater than the maximum value reportable by this object then this object must report its maximum value (4,294,967,295) and ifHighSpeed must be used to report the interface's speed. For a sub-layer which has no concept of bandwidth, this object must be zero.
ifType	The type of interface. Additional values for ifType are assigned by the Internet Assigned Numbers Authority (IANA), through updating the syntax of the IANAifType textual convention.

Table F.6 IP-MIB Supported Objects

Object Name	Object Description in MIB
icmpInAddrMaskReps	The number of ICMP Address Mask Reply messages received.
icmpInAddrMasks	The number of ICMP Address Mask Request messages received.
icmpInDestUnreachs	The number of ICMP Destination Unreachable messages received.
icmpInEchoReps	The number of ICMP Echo Reply messages received.
icmpInEchos	The number of ICMP Echo (request) messages received.
icmpInErrors	The number of ICMP messages which the entity received but determined as having ICMP-specific errors (bad ICMP checksums, bad length, etc.).
icmpInMsgs	The total number of ICMP messages which the entity received. Note that this counter includes all those counted by icmpInErrors.
icmpInParmProbs	The number of ICMP Parameter Problem messages received.
icmpInRedirects	The number of ICMP Redirect messages received.
icmpInSrcQuenchs	The number of ICMP Source Quench messages received.
icmpInTimeExcds	The number of ICMP Time Exceeded messages received.
icmpInTimestampReps	The number of ICMP Timestamp Reply messages received.
icmpInTimestamps	The number of ICMP Timestamp (request) messages received.
icmpOutAddrMaskReps	The number of ICMP Address Mask Reply messages sent.
icmpOutAddrMasks	The number of ICMP Address Mask Request messages sent.
icmpOutDestUnreachs	The number of ICMP Destination Unreachable messages sent.
icmpOutEchoReps	The number of ICMP Echo Reply messages sent.
icmpOutEchos	The number of ICMP Echo (request) messages sent.

Table F.6 IP-MIB Supported Objects (Continued)

Object Name	Object Description in MIB
icmpOutErrors	The number of ICMP messages which this entity did not send due to problems discovered within ICMP such as a lack of buffers. This value must not include errors discovered outside the ICMP layer such as the inability of IP to route the resultant datagram. In some implementations there may be no types of error which contribute to this counter's value.
icmpOutMsgs	The total number of ICMP messages which this entity attempted to send. Note that this counter includes all those counted by icmpOutErrors.
icmpOutParmProbs	The number of ICMP Parameter Problem messages sent.
icmpOutRedirects	The number of ICMP Redirect messages sent. For a host, this object will always be zero, since hosts do not send redirects.
icmpOutSrcQuenchs	The number of ICMP Source Quench messages sent.
icmpOutTimeExcds	The number of ICMP Time Exceeded messages sent.
icmpOutTimestampReps	The number of ICMP Timestamp Reply messages sent.
icmpOutTimestamps	The number of ICMP Timestamp (request) messages sent.
ipAdEntAddr	The IP address to which this entry's addressing information pertains.
ipAdEntBcastAddr	The value of the least-significant bit in the IP broadcast address used for sending datagrams on the (logical) interface associated with the IP address of this entry. For example, when the Internet standard all-ones broadcast address is used, the value will be 1. This value applies to both the subnet and network broadcasts addresses used by the entity on this (logical) interface.
ipAdEntIfIndex	The index value which uniquely identifies the interface to which this entry is applicable. The interface identified by a particular value of this index is the same interface as identified by the same value of RFC 1573's ifIndex.
ipAdEntNetMask	The subnet mask associated with the IP address of this entry. The value of the mask is an IP address with all the network bits set to 1 and all the hosts bits set to 0.
ipAdEntReasmMaxSize	The size of the largest IP datagram which this entity can re-assemble from incoming IP fragmented datagrams received on this interface.
ipDefaultTTL	The default value inserted into the Time-To-Live field of the IP header of datagrams originated at this entity, whenever a TTL value is not supplied by the transport layer protocol.
ipForwarding	The indication of whether this entity is acting as an IP router in respect to the forwarding of datagrams received by, but not addressed to, this entity. IP routers forward datagrams. IP hosts do not (except those source-routed via the host).
ipForwDatagrams	The number of input datagrams for which this entity was not their final IP destination, as a result of which an attempt was made to find a route to forward them to that final destination. In entities which do not act as IP routers, this counter will include only those packets which were Source-Routed via this entity, and the Source-Route option processing was successful.

Table F.6 IP-MIB Supported Objects (Continued)

Object Name	Object Description in MIB
ipFragCreates	The number of IP datagram fragments that have been generated as a result of fragmentation at this entity.
ipFragFails	The number of IP datagrams that have been discarded because they needed to be fragmented at this entity but could not be, for example because their Don't Fragment flag was set.
ipFragOKs	The number of IP datagrams that have been successfully fragmented at this entity.
ipInAddrErrors	The number of input datagrams discarded because the IP address in their IP header's destination field was not a valid address to be received at this entity. This count includes invalid addresses (for example, 0.0.0.0) and addresses of unsupported Classes (for example, Class E). For entities which are not IP routers and therefore do not forward datagrams, this counter includes datagrams discarded because the destination address was not a local address.
ipInDelivers	The total number of input datagrams successfully delivered to IP user-protocols (including ICMP).
ipInDiscards	The number of input IP datagrams for which no problems were encountered to prevent their continued processing, but which were discarded (e.g., for lack of buffer space). Note that this counter does not include any datagrams discarded while awaiting re-assembly.
ipInHdrErrors	The number of input datagrams discarded due to errors in their IP headers, including bad checksums, version number mismatch, other format errors, time-to-live exceeded, errors discovered in processing their IP options, etc.
ipInReceives	The total number of input datagrams received from interfaces, including those received in error.
ipInUnknownProtos	The number of locally addressed datagrams received successfully but discarded because of an unknown or unsupported protocol.
ipNetToMediaIfIndex	The interface on which this entry's equivalence is effective. The interface identified by a particular value of this index is the same interface as identified by the same value of RFC 1573's ifIndex.
ipNetToMediaNetAddress	The IpAddress corresponding to the media-dependent ‘physical' address.
ipNetToMediaPhysAddress	The media-dependent ‘physical' address.
ipNetToMediaType	The type of mapping. Setting this object to the value invalid(2) has the effect of invalidating the corresponding entry in the ipNetToMediaTable. That is, it effectively disassociates the interface identified with said entry from the mapping identified with said entry. It is an implementation- specific matter as to whether the agent removes an invalidated entry from the table. Accordingly, management stations must be prepared to receive tabular information from agents that corresponds to entries not currently in use. Proper interpretation of such entries requires examination of the relevant ipNetToMediaType object.

Table F.6 IP-MIB Supported Objects (Continued)

Object Name	Object Description in MIB
ipOutDiscards	The number of output IP datagrams for which no problem was encountered to prevent their transmission to their destination, but which were discarded (e.g., for lack of buffer space). Note that this counter would include datagrams counted in ipForwDatagrams if any such packets met this (discretionary) discard criterion.
ipOutNoRoutes	The number of IP datagrams discarded because no route could be found to transmit them to their destination. Note that this counter includes any packets counted in ipForwDatagrams which meet this ‘no-route' criterion. Note that this includes any datagrams which a host cannot route because all of its default routers are down.
ipOutRequests	The total number of IP datagrams which local IP user- protocols (including ICMP) supplied to IP in requests for transmission. Note that this counter does not include any datagrams counted in ipForwDatagrams.
ipReasmFails	The number of failures detected by the IP re-assembly algorithm (for whatever reason: timed out, errors, etc.). Note that this is not necessarily a count of discarded IP fragments since some algorithms (notably the algorithm in RFC 815) can lose track of the number of fragments by combining them as they are received.
ipReasmOKs	The number of IP datagrams successfully re-assembled.
ipReasmReqds	The number of IP fragments received which needed to be reassembled at this entity.
ipReasmTimeout	The maximum number of seconds which received fragments are held while they are awaiting reassembly at this entity.

Table F.7 SNMP-USER-BASED-SM-MIB Objects

Object Name	Object Description in MIB
usmStatsDecryptionErrors	The total number of packets received by the SNMP engine which were dropped because they could not be decrypted.
usmStatsNotInTimeWindows	The total number of packets received by the SNMP engine which were dropped because they appeared outside of the authoritative SNMP engine's window.
usmStatsUnknownEngineIDs	The total number of packets received by the SNMP engine which were dropped because they referenced an snmpEngineID that was not known to the SNMP engine.
usmStatsUnknownUserNames	The total number of packets received by the SNMP engine which were dropped because they referenced a user that was not known to the SNMP engine.
usmStatsUnsupportedSecLevels	The total number of packets received by the SNMP engine which were dropped because they requested a security Level that was unknown to the SNMP engine or otherwise unavailable.
usmStatsWrongDigests	The total number of packets received by the SNMP engine which were dropped because they did not contain the expected digest value.
usmUserSpinLock	An advisory lock used to allow several cooperating Command Generator Applications to coordinate their use of facilities to alter secrets in the usmUserTable.

Table F.7 SNMP-USER-BASED-SM-MIB Objects (Continued)

Object Name	Object Description in MIB
usmUserStatus	The status of this conceptual row. Until instances of all corresponding columns are appropriately configured, the value of the corresponding instance of the usmUserStatus column is 'notReady'. In particular, a newly created row for a user who employs authentication, cannot be made active until the corresponding usmUserCloneFrom and usmUserAuthKeyChange have been set. Further, a newly created row for a user who also employs privacy, cannot be made active until the usmUserPrivKeyChange has been set. The RowStatus TC [RFC2579] requires that this DESCRIPTION clause states under which circumstances other objects in this row can be modified: The value of this object has no effect on whether other objects in this conceptual row can be modified, except for usmUserOwnAuthKeyChange and usmUserOwnPrivKeyChange. For these 2 objects, the value of usmUserStatus MUST be active.

Table F.8 SNMPv2-MIB Supported Objects

Object Name	Object Description in MIB
snmpEnableAuthenTraps	Indicates whether the SNMP entity is permitted to generate authenticationFailure traps. The value of this object overrides any configuration information; as such, it provides a means whereby all authenticationFailure traps may be disabled. Note that it is strongly recommended that this object be stored in non-volatile memory so that it remains constant across re-initializations of the network management system.
snmpInASNParseErrs	The total number of ASN.1 or BER errors encountered by the SNMP entity when decoding received SNMP messages.
snmpInBadCommunityNames	The total number of SNMP messages delivered to the SNMP entity which used a SNMP community name not known to said entity.
snmpInBadCommunityUses	The total number of SNMP messages delivered to the SNMP entity which represented an SNMP operation which was not allowed by the SNMP community named in the message.
snmpInBadVersions	The total number of SNMP messages which were delivered to the SNMP entity and were for an unsupported SNMP version.
snmpInPkts	The total number of messages delivered to the SNMP entity from the transport service.
snmpProxyDrops	The total number of GetRequest-PDUs, GetNextRequest-PDUs, GetBulkRequest-PDUs, SetRequest-PDUs, and InformRequest-PDUs delivered to the SNMP entity which were silently dropped because the transmission of the (possibly translated) message to a proxy target failed in a manner (other than a time-out) such that no Response-PDU could be returned.
snmpSetSerialNo	An advisory lock used to allow several cooperating SNMPv2 entities, all acting in a manager role, to coordinate their use of the SNMPv2 set operation. This object is used for coarse-grain coordination. To achieve fine-grain coordination, one or more similar objects might be defined within each MIB group, as appropriate.

Table F.8 SNMPv2-MIB Supported Objects (Continued)

Object Name	Object Description in MIB
snmpSilentDrops	The total number of GetRequest-PDUs, GetNextRequest-PDUs, GetBulkRequest-PDUs, SetRequest-PDUs, and InformRequest-PDUs delivered to the SNMP entity which were silently dropped because the size of a reply containing an alternate Response-PDU with an empty variable-bindings field was greater than either a local constraint or the maximum message size associated with the originator of the request.
sysContact	The textual identification of the contact person for this managed node, together with information on how to contact this person. If no contact information is known, the value is the zero-length string.
sysDescr	A textual description of the entity. This value must include the full name and version identification of the system's hardware type, software operating-system, and networking software.
sysLocation	The physical location of this node (e.g., 'telephone closet, 3rd floor'). If the location is unknown, the value is the zero-length string.
sysName	An administratively assigned name for this managed node. By convention, this is the node's fully qualified domain name. If the name is unknown, the value is the zero-length string.
sysObjectID	The vendor's authoritative identification of the network management subsystem contained in the entity. This value is allocated within the SMI enterprises subtree (1.3.6.1.4.1) and provides an easy and unambiguous means for determining `what kind of box' is being managed. For example, if vendor `Flintstones, Inc.' was assigned the subtree 1.3.6.1.4.1.4242, it could assign the identifier 1.3.6.1.4.1.4242.1.1 to its `Fred Router'.
sysServices	A value which indicates the set of services that this entity may potentially offers. The value is a sum. This sum initially takes the value zero, Then, for each layer, L, in the range 1 through 7, that this node performs transactions for, 2 raised to (L - 1) is added to the sum. For example, a node which performs only routing functions would have a value of 4 (2^(3-1)). In contrast, a node which is a host offering application services would have a value of 72 (2^(4-1) + 2^(7-1)). Note that in the context of the Internet suite of protocols, values must be calculated accordingly: layer functionality 1 physical (for example, repeaters) 2 datalink or subnetwork (for example, bridges) 3 Internet (for example, supports IP) 4 end-to-end (for example, supports TCP) 7 applications (for example, supports SMTP) For systems including OSI protocols, layers 5 and 6 may also be counted.
sysUpTime	The time (in hundredths of a second) since the network management portion of the system was last re-initialized.

APPENDIX G

TCP/IP PROTOCOL HEADERS

This appendix is a brief overview of the common TCP/IP protocol headers.

The following sections are included:

Internet Protocol (IP)

For the Internet Protocol (IP) specification, please refer to RFC791 available at http://www.rfc-editor.org/.

Table G.1 IP Datagram

<table>
<tr><th colspan="2">bits 0 - 7</th><th colspan="2">bits 8 - 15</th><th colspan="2">bits 16 - 23</th><th colspan="2">bits 24 - 31</th></tr>
<tr><td><b>Version</b>
(4 bits)</td><td><b>IP Header Length</b>
(4 bits)</td><td colspan="2"><b>Type of Service</b>
(8 bits)</td><td colspan="4"><b>Total Length</b> (in number of bytes)
(16 bits)</td></tr>
<tr><td colspan="4"><b>IP Identification Number</b>
(16 bits)</td><td><b>Flags</b>
(3 bits)</td><td colspan="3"><b>Fragment Offset</b>
(13 bits)</td></tr>
<tr><td colspan="2"><b>Time to Live</b>
(8 bits)</td><td colspan="2"><b>Protocol Number</b>
(8 bits)</td><td colspan="4"><b>Header Checksum</b>
(16 bits)</td></tr>
<tr><td colspan="8"><b>Source IP Address</b>
(32 bits)</td></tr>
<tr><td colspan="8"><b>Destination IP Address</b>
(32 bits)</td></tr>
<tr><td colspan="8"><b>Options</b> (if any) + <b>Padding</b> (matching to 32-bit boundary)</td></tr>
<tr><td colspan="8">(data . . .)</td></tr>
</table>

Internet Control Message Protocol (ICMP)

For the Internet Control Message Protocol (ICMP) specification, please refer to RFC792 available at http://www.rfc-editor.org/.

Table G.2 ICMP Message

<table>
<tr><th>bits 0 - 7</th><th>bits 8 - 15</th><th>bits 16 - 23</th><th>bits 24 - 31</th></tr>
<tr><td><b>Type</b>
(8 bits)</td><td><b>Code</b>
(8 bits)</td><td colspan="2"><b>Checksum</b>
(16 bits)</td></tr>
<tr><td colspan="4">(data . . .)</td></tr>
</table>

Transmission Control Protocol (TCP)

For the Transmission Control Protocol (TCP) specification, please refer to RFC793 available at http://www.rfc-editor.org/.

Table G.3 TCP Segment

<table>
<tr><th colspan="2">bits 0 - 7</th><th colspan="6">bits 8- 15</th><th>bits 16 - 23</th><th>bits 24 - 31</th></tr>
<tr><td colspan="8">**Source Port Number**
(16 bits)</td><td colspan="2">**Destination Port Number**
(16 bits)</td></tr>
<tr><td colspan="10">**Sequence Number**
(32 bits)</td></tr>
<tr><td colspan="10">**Acknowledgement Number**
(32 bits)</td></tr>
<tr><td>**TCP Header Length**
(4 bits)</td><td>reserved
(6 bits)</td><td>**URG**</td><td>**ACK**</td><td>**PSH**</td><td>**RST**</td><td>**SYN**</td><td>**FIN**</td><td colspan="2">**Window Size**
(16 bits)</td></tr>
<tr><td colspan="8">**Checksum**
(16 bits)</td><td colspan="2">**Urgent Pointer**
(16 bits)</td></tr>
<tr><td colspan="10">**Options** (if any) + **Padding** (matching to 32-bit boundary)</td></tr>
<tr><td colspan="10">(data . . .)</td></tr>
</table>

User Datagram Protocol (UDP)

For the User Datagram Protocol (UDP) specification, please refer to RFC768 available at http://www.rfc-editor.org/.

Table G.4 UDP Datagram

<table>
<tr><th>bits 0 - 7</th><th>bits 8 - 15</th><th>bits 16 - 23</th><th>bits 24 - 31</th></tr>
<tr><td colspan="2">**Source Port Number**
(16 bits)</td><td colspan="2">**Destination Port Number**
(16 bits)</td></tr>
<tr><td colspan="2">**User Datagram Length**
(16 bits)</td><td colspan="2">**Checksum**
(16 bits)</td></tr>
<tr><td colspan="4">(data . . .)</td></tr>
</table>

Appendix H

ASCII Character Codes

The decimal and hexadecimal values of the ASCII characters are presented for interpreting traffic captures and predefined Situation Contexts.

The following sections are included:

ASCII Character Codes

Table H.1 ASCII Character Codes

ASCII	Dec	Hex	ASCII	Dec	Hex	ASCII	Dec	Hex	ASCII	Dec	Hex
NUL	0	0x00	*SPACE*	32	0x20	@	64	0x40	`	96	0x60
SOH	1	0x01	!	33	0x21	A	65	0x41	a	97	0x61
STX	2	0x02	"	34	0x22	B	66	0x42	b	98	0x62
ETX	3	0x03	#	35	0x23	C	67	0x43	c	99	0x63
EOT	4	0x04	$	36	0x24	D	68	0x44	d	100	0x64
ENQ	5	0x05	%	37	0x25	E	69	0x45	e	101	0x65
ACK	6	0x06	&	38	0x26	F	70	0x46	f	102	0x66
BEL	7	0x07	'	39	0x27	G	71	0x47	g	103	0x67
BS	8	0x08	(	40	0x28	H	72	0x48	h	104	0x68
HT	9	0x09	)	41	0x29	I	73	0x49	i	105	0x69
LF	10	0x0A	*	42	0x2A	J	74	0x4A	j	106	0x6A
VT	11	0x0B	+	43	0x2B	K	75	0x4B	k	107	0x6B
FF	12	0x0C	,	44	0x2C	L	76	0x4C	l	108	0x6C
CR	13	0x0D	-	45	0x2D	M	77	0x4D	m	109	0x6D
SO	14	0x0E	.	46	0x2E	N	78	0x4E	n	110	0x6E
SI	15	0x0F	/	47	0x2F	O	79	0x4F	o	111	0x6F
DLE	16	0x10	0	48	0x30	P	80	0x50	p	112	0x70
DC1	17	0x11	1	49	0x31	Q	81	0x51	q	113	0x71
DC2	18	0x12	2	50	0x32	R	82	0x52	r	114	0x72
DC3	19	0x13	3	51	0x33	S	83	0x53	s	115	0x73
DC4	20	0x14	4	52	0x34	T	84	0x54	t	116	0x74
NAK	21	0x15	5	53	0x35	U	85	0x55	u	117	0x75
SYN	22	0x16	6	54	0x36	V	86	0x56	v	118	0x76
ETB	23	0x17	7	55	0x37	W	87	0x57	w	119	0x77
CAN	24	0x18	8	56	0x38	X	88	0x58	x	120	0x78
EM	25	0x19	9	57	0x39	Y	89	0x59	y	121	0x79

Table H.1 ASCII Character Codes (Continued)

ASCII	Dec	Hex	ASCII	Dec	Hex	ASCII	Dec	Hex	ASCII	Dec	Hex
SUB	26	0x1A	:	58	0x3A	Z	90	0x5A	z	122	0x7A
ESC	27	0x1B	;	59	0x3B	[	91	0x5B	{	123	0x7B
FS	28	0x1C	<	60	0x3C	\	92	0x5C	|	124	0x7C
GS	29	0x1D	=	61	0x3D	]	93	0x5D	}	125	0x7D
RS	30	0x1E	>	62	0x3E	^	94	0x5E	~	126	0x7E
US	31	0x1F	?	63	0x3F	_	95	0x5F	*DELETE*	127	0x7F

ASCII Control Codes

Table H.2 ASCII Control Codes

ASCII	Dec	Hex	Description
NUL	0	0x00	Null
SOH	1	0x01	Start of Heading
STX	2	0x02	Start of Text
ETX	3	0x03	End of Text
EOT	4	0x04	End of Transmission
ENQ	5	0x05	Enquiry
ACK	6	0x06	Acknowledge
BEL	7	0x07	Bell
BS	8	0x08	Backspace
HT	9	0x09	Horizontal Tabulation
LF	10	0x0A	Line Feed
VT	11	0x0B	Vertical Tabulation
FF	12	0x0C	Form Feed
CR	13	0x0D	Carrier Return
SO	14	0x0E	Shift Out
SI	15	0x0F	Shift In
DLE	16	0x10	Data Line Escape
DC1	17	0x11	Device Control 1

Table H.2 ASCII Control Codes (Continued)

ASCII	Dec	Hex	Description
DC2	18	0x12	Device Control 2
DC3	19	0x13	Device Control 3
DC4	20	0x14	Device Control 4
NAK	21	0x15	Negative Acknowledge
SYN	22	0x16	Synchronous Idle
ETB	23	0x17	End of Transmission Block
CAN	24	0x18	Cancel
EM	25	0x19	End of Medium
SUB	26	0x1A	Substitute
ESC	27	0x1B	Escape
FS	28	0x1C	File Separator
GS	29	0x1D	Group Separator
RS	30	0x1E	Record Separator
US	31	0x1F	Unit Separator

Glossary

A

Access Control List

A list of Elements that can be used to define the Elements that an Administrators with restricted permissions can access. See also Administrator Role and Granted Element.

Action

What the engine should do with a packet that matches the criteria for a particular rule in the Security Policy.

Action Option

Additional action-specific selections that affect how the traffic is handled set in the Action cell in rules.

Active Management Server

The Management Server that currently has control of all Domains in an environment that has at least one Additional Management Server.

Additional Log Server

A Log Server defined as a backup channel for components that primarily send their logs to some other Log Server.

Additional Management Server

A redundant Management Server that replicates the configuration data from the Active Management Server under normal conditions so that the services offered by the Management Server can be used without interruption if components fail or are otherwise unavailable.

Address Range

A Network Element that defines a range of IP addresses. Use to avoid having to repeatedly type in the same IP addresses when defining address ranges that do not correspond to whole networks.

Address Resolution Protocol (ARP)

An Internet standard (RFC 826) protocol used to associate IP addresses with the media hardware address of a network interface card on a local area network (LAN).

Administrator

An Element that defines the details of a single person that is allowed to log on to the SMC using the Management Client. If used as a general term, Web Portal Users are also considered as administrators.

Administrator Role

An Element that defines which actions an Administrator with restricted permissions is allowed to take. See also Granted Element and Permission Level.

Aggressive Mode

The authentication of two IPsec end-points with only three messages, as opposed to Main Mode's six. Aggressive mode also does not provide PFS support, and SA negotiation is limited. See Main Mode (page 289). See also Security Association (SA) (page 295).

AH (Authentication Header)

See Authentication Header (AH) (page 276).

Alert Chain

A list of rules defining which Alert Channels are used, and in which order, when an alert entry is directed to the Alert Chain from an Alert Policy to be escalated out from the SMC. See also Alert Escalation.

Alert Channel

A method of sending alerts out from the Log Server. You can send alerts via SMTP (e-mail), SNMP, SMS text messages, or some other action you define in a custom script. Alert Channels are defined in the Log Server's properties, after which they can be used in Alert Chains.

Alert Element

An Element that gives the name and description to an Alert Event. The Alert element can be used as a matching criteria in the rules of an Alert Policy.

Alert Entry

A log message with an alert status that has been raised based on some Situation (which you can see in the Logs View). Alert entries trigger Alert Escalation.

Alert Escalation

Sending alerts out from the SMC to administrators through Alert Channels (such as e-mail) according to a predefined Alert Chain until the original Alert Entry is acknowledged by some administrator in the Logs View.

Alert Event

A pattern in traffic or a problem in the system's operation that matches some Situation used in a policy or internally in the system, and triggers an Alert Entry.

Alert Policy

A list of rules defining if an Alert Entry is escalated and which Alert Chain is used for escalating which type of alert entries. See also Alert Escalation.

Alias

An Element that can be used to represent other network elements in configurations. It differs from a group element in that it does not represent all the elements at once: the value it takes in a configuration can be different on each engine where it is used.

Allow Action

An Action parameter that allows a connection matching that rule to pass through the Firewall to its destination.

Analyzer

1) A legacy IPS device that analyzes the log information from Sensors according to its policy to find patterns, so that separate log entries can be combined together. See also Log Server, Security Engine.

2) The legacy Element that represents an Analyzer device in the SMC.

Antispoofing

Technique used to protect against malicious packages whose IP header information has been altered. See also IP Spoofing (page 286).

Application

A category of Tags for Situations. Meant for grouping Situations that detect known vulnerabilities in a particular software application.

Application Layer Gateway; Application Level Firewall

A firewall system, or gateway, in which packets are examined based on the application protocol being used (e.g., telnet, FTP, SMTP). Proxies for each application-level service are installed on the gateway, and are often configured to relay a conversation between two systems. That is, a packet's destination is the gateway, which then establishes a separate connection to the other system to complete the connection.

Apply VPN Action

An Action in the Firewall Policy that directs traffic from protected local networks into the Virtual Private Network (VPN) tunnel and allows traffic that arrives through a VPN, but does not match non-VPN traffic from outside networks into the protected networks. See also Enforce VPN Action (page 282).

ARP (Address Resolution Protocol)

See Address Resolution Protocol (ARP) (page 273).

Asymmetric Encryption

A cryptographic technology that uses a pair of keys. The message is encrypted with the public half of a pair and can then be decrypted only with the matching private half of the key pair. Public key technology can be used to create digital signatures and deal with key management issues. Also referred to as public key encryption. See also Symmetric Encryption (page 298) and Public-key Cryptography (page 293).

Auditing

An SMC feature that logs administrators' actions and allows administrators with unrestricted permissions to view and manage these logs to keep track of system changes.

Authentication

The process of proving that someone or something is who or what they claim to be. For example, typing a simple username-password combination is a form of authentication.

Authentication Header (AH)

A security protocol supported by the IPsec protocol to enhance traffic security. It enables the authentication and integrity of data against packet corruption or tampering. AH protocol can use SHA-1 or MD5 to generate a hash signature based on a secret component from the SA, the packet payload and some parts of the packet header. See also Security Association (SA) (page 295).

Authentication Server

A component of the Security Management Center (SMC) that provides authentication services for end-user and Administrator logins.

Authentication Token/Authenticator

A portable device for authenticating a user. Authentication tokens typically operate by challenge/response, time-based code sequences, or other techniques. One of the most commonly used tokens is the RSA SecurID card.

Authorization

The process of giving someone or something permission to do or have something. Usually related to authentication; once a user has authenticated (proved who they are), they are authorized (given permission) to perform certain actions.

B

Balancing Mode

A Security Engine cluster mode that attempts to divide the traffic as equally as possible between the online engines participating in the cluster. Confer to Standby Mode (page 298).

Bandwidth Management

The process of determining and enforcing bandwidth limits and guarantees for different types of traffic either together with Traffic Prioritization or on its own. Also see QoS Class (page 293) and QoS Policy (page 293).

Blacklisting

1) The process of blocking unwanted network traffic either manually or automatically.

2) Persistently blocking access to certain URLs manually.

Bookmark

A stored link to a view or layout in the Management Client.

Bookmark Folder

A folder in the toolbar of the Management Client for storing and sharing Bookmarks.

Border Routing

Routing of connections between different autonomous systems.

BrightCloud

A URL Filtering categorization service that provides categories for malicious sites as well as several categories for different types of non-malicious content that may be considered objectionable.

Buffer Overflow

When a program's data in the memory of a computer exceeds the space reserved for it (the buffer), data may in some circumstances be written on other parts of the memory area. Attackers may use buffer overflows to execute harmful program code on a remote system.

Bugtraq

A mailing list for discussing network security related issues, such as vulnerabilities.

Bulk Encryption Algorithm

Describes symmetric encryption algorithms which operate on fixed-size blocks of plaintext and generates a block of ciphertext for each.

C

CA

See Certificate Authority (CA) (page 277).

CAN

A candidate for a CVE entry.

Capture Interface

An IPS Engine or Layer 2 Firewall interface that can listen to traffic passing in the network, but which is not used for routing traffic through the engine. See also Inline Interface.

Category

A way of organizing elements and policies to display a specific subset at a time when configuring a large SMC environment in the Management Client to make it easier to find the relevant elements. For example, a Managed Service Provider (MSP) who manages networks of several different customers can add a customer-specific category to each element and policy to be able to view one customer's elements and policies at a time.

Certificate

Electronic identification of a user or device. Certificates prove the user or device is who or what they claim to be. This is done through using public/private key pairs and digital signatures. Certificates are used in the SMC for authenticating communications between the SMC components and for Virtual Private Network (VPN) authentication. Digital certificates are granted and verified by a Certificate Authority (CA), such as the internal CA included in the Management Server.

Certificate Authority (CA)

A trusted third-party organization or company that issues digital certificates, used to create digital signatures and public-private key pairs. The role of the CA in this process is to guarantee that the individual granted the unique certificate is, in fact, who he or she claims to be.

Challenge/Response

An authentication technique whereby a server sends an unpredictable challenge to the user, who computes a response using some form of authentication token, which can be an authenticator, or pre-shared keys used to encrypt random data.

Checksum

A one-way function applied to a file to produce a unique “fingerprint” of the file for later reference. File tampering can then be discovered by verifying the checksum value in the future.

CIS

See Content Inspection Server (CIS) (page 279).

Client

In a client-server architecture, a client is usually an application running on a computer or a workstation that uses services provided by a Server.

Client Protection Certificate Authority

Contains the credentials that the engine uses to sign replacement server-side certificates the engine creates and presents to clients when inspecting the clients’ HTTPS connections with external servers. Also see Server Credentials (page 296).

Client-to-Gateway VPN

A Virtual Private Network (VPN) between a software client and a VPN Gateway. Allows connecting mobile and home office workers safely to corporate resources using a secure (authenticated and encrypted) connection through insecure networks.

Cluster

A group of devices, or nodes, that share a given work load. In the SMC, you can cluster Firewalls, IPS engines, and Layer 2 Firewalls to share the load and provide redundancy, allowing, for example, scheduled maintenance that takes one node out of service without interrupting services to the users.

Cluster Mode

Determines if all members of a cluster participate to traffic processing at all times (Balancing Mode) or if other members remain inactive until a traffic-processing member stops processing traffic (Standby Mode).

Cluster Virtual IP Address (CVI)

An IP and MAC address shared by all nodes in a cluster, which are used by every node in a cluster for communication. These interfaces give the cluster a single identity on the network, reducing the complexity of routing and network design. CVIs handle the traffic directed to the Firewall for inspection in Firewall Clusters.

Combined Sensor-Analyzer

1) A legacy IPS device that has both Sensor and Analyzer engines running simultaneously on the same hardware.

2) The legacy Element that represents a Combined Sensor-Analyzer device in the SMC.

See also IPS Engine.

Connection Tracking

The set of data maintained for a connection. Used for relating incoming packets to existing connections. Connection tracking information also includes information necessary for NAT (Network Address Translation), Load Balanced Routing, and Protocol Agents. May also contain accounting information.

Contact Address

The IP address that is needed to contact a device performing a function in the SMC when there is NAT (Network Address Translation) being performed in between the two devices and thus the actual IP address assigned to the network interface cannot be used directly.

Content Inspection Server (CIS)

A server that performs detailed examination of a connection's data and assists in the determination to allow or discard packets. Common examples include virus scanning or filtering of web URLs. Also known as *content screening*.

Continue Action

A policy parameter that sets default values to those used in the rule. The defaults are used in all subsequent rules except where specifically overridden until some other rule with the Continue action changes the values or the policy ends.

Context

An Element that is added to a Situation to define what the Situation should match. Provides a framework for defining parameters, which are most entered as a regular expression, or through a set of fields and options that the administrators adjust.

Correlation Situation

A Situation that defines the patterns that the Log Server and/or the Security Engines look for when it examines event data produced by engines.

CRL Server

A server that maintains a Certificate Revocation List (CRL), which can be used in Authentication to check if the certificate has been cancelled.

Custom Alert

An Alert Element that is defined by an SMC administrator, as opposed to a ready-made Default Element created by the SMC.

CVE

A dictionary that provides common names for publicly known information security vulnerabilities and exposures and thus a standardized description for each vulnerability that links the vulnerability information of different tools and databases.

CVI

See Cluster Virtual IP Address (CVI) (page 278).

Default Element

An Element that is present in the SMC at installation, or is added to the SMC during an upgrade or from a Dynamic Update (Package). Default elements cannot be modified or deleted by administrators, but they may be modified or deleted by dynamic update packages or upgrades.

Defragmentation

The process by which a large block of data is broken up into smaller pieces (datagrams), so that it can be packaged and transmitted by the underlying network technology (Fragmentation). Once the smaller pieces arrive at their destination, the datagrams are reassembled into the larger block of data (defragmentation).

DHCP (Dynamic Host Configuration Protocol)

A protocol for dynamically assigning IP addresses and other network information to an interface, based on BOOTP. A device on a network with no network information can broadcast a request for an IP address, subnet mask, default gateway and other information from a DHCP server on that same network. DHCP is defined in RFC 2131.

Diagram

An Element that contains one or more network diagrams created using the Diagram Editor.

Digital Certificate

See Certificate (page 277).

Discard Action

An Action parameter that stops all connections matching to the rule without sending any notification to the connecting host. Confer to Refuse Action (page 294).

Dispatch Clustering

See Packet Dispatch (page 291).

DMZ Network

A DMZ (DeMilitarized Zone Network) is a network separate from both internal and external networks, and connected through a gateway. Often used for isolating bastion hosts or publicly available machines, e.g., mail and HTTP servers are typically located on a DMZ network. Sometimes also referred to as a *screened subnetwork*.

DNS Spoofing

An attack method whereby the DNS name of a system is assumed by a malicious system, either by corrupting the name service cache of a victim, or by compromising a domain name server for a valid domain. The victim system is then directed to the malicious system instead of the original server.

Domain

Domains are administrative boundaries that allow you to separate the configuration details and other information in the SMC for the purpose of limiting administrator access.

DoS Attack (Denial of Service)

An attack with the objective of causing enough disruption in a computer system that its usability to legitimate users suffers. For example, and attacker may target a website so that it becomes overloaded, and slows down so much that it becomes unusable for people wishing to view it.

DSCP (DiffServ Code Point)

The Differentiated Services (DiffServ) Type of Service (ToS Flag) field added to packets in the network.

DSCP Mark

A field in QoS Policy rules that writes a particular DSCP (DiffServ Code Point) marker to the packets, if the QoS Policy is applied on the interface the packets use to exit the Firewall.

DSCP Match

A field in QoS Policy rules that assigns the QoS Class specified in the rule to incoming packets that have a specific DSCP (DiffServ Code Point) marker set, if the QoS Policy is applied on the interface the packets use to enter of the Firewall.

Dynamic IP address

An IP address that is assigned by using the DHCP (Dynamic Host Configuration Protocol).

Dynamic NAT

A way to translate network addresses, where for each original address, a translated address and possibly a port are selected dynamically from a predefined pool.

Dynamic Update (Package)

A file supplied by McAfee that provides updates to Default Elements and policies, most importantly to the Situation and Vulnerability information that is used for traffic inspection in Inspection Rules.

E

Element

An SMC object that represents the equipment in your physical networks or some area or concept of configuration. Elements may, for example, represent a single device such as a server, a range of IP addresses, or some configuration aid in the SMC, such as a Category. See also Network Element (page 290).

Encryption

Used for data security, encryption translates any data into a secret code. Public-key encryption and symmetric encryption are the main types of encryption. Decrypting ciphertext (encrypted data) into plaintext requires access to a secret key.

Encryption Domain

Networks that are defined to be behind a certain VPN gateway in a Virtual Private Network (VPN) configuration.

Encryption Key

The data that is used to convert plaintext to ciphertext. In symmetric algorithms, the same key is the decryption key as well. In public key algorithms, a different, but related key is used to convert the ciphertext back into plaintext.

Encryption Policy

Settings that define which encryption and authentication methods are used to establish a Virtual Private Network (VPN).

Enforce VPN Action

A Firewall Action parameter that directs traffic from protected local networks into the Virtual Private Network (VPN) tunnel and allows traffic that arrives through a VPN, and drops any non-VPN traffic from external networks to the local network that matches the rule. See also Apply VPN Action (page 275).

Ethernet Rules

A set of rules in the IPS Policy that define which Ethernet traffic is allowed or discarded by a Sensor in Transparent Access Control Mode.

Expression

An Element that can be used to accurately define a whole complex set of elements by including and excluding elements using logical expressions.

External Gateway

Any VPN Gateway that is managed by a different Management Server than the one on which the Virtual Private Network (VPN) is being configured.

F

Filter

A description of log fields and their values combined together using operators for the purpose of sorting in or out log, alert, and audit entries. Used, for example, to filter out logs from the display in the Logs View so that those entries that are interesting at the moment can be found more easily.

Firewall

1) An Element that represents the firewall device in the SMC. Either a Single Firewall or a Firewall Cluster.

2) The device running the Next Generation Firewall (NGFW) engine software in the Firewall/VPN role.

Firewall Cluster

A Group of two or more *Firewall Engines* that work together as if they were a single unit.

Firewall Engine

The device that runs the Next Generation Firewall (NGFW) engine software in the Firewall/VPN role. This can be a standard server, an engine installed on a virtualization platform, or a McAfee NGFW appliance. Represented by a Firewall Node in the Management Client.

Firewall Node

An individual Firewall Engine in the Management Client, representing a device that runs the Next Generation Firewall (NGFW) engine software in the Firewall/VPN role as part of a Firewall Cluster or a Single Firewall.

Forward Action

A Firewall Action parameter that directs traffic from protected local networks or from a Virtual Private Network (VPN) tunnel into another VPN tunnel.

Fragmentation

The process by which a large block of data is broken up into smaller pieces (datagrams), so that it can be packaged and transmitted by the underlying network technology (fragmentation). Once the smaller pieces arrive at their destination, the datagrams are reassembled into the larger block of data (Defragmentation).

G

Gateway

A device that provides VPN access for other devices.

Gateway Certificate

A Certificate used for authenticating a Gateway to other Gateways and VPN Clients in a VPN.

Gateway Profile

An element that defines a set of VPN-related capabilities that a VPN Gateway supports.

Gateway Settings

An element that contains general settings for McAfee Firewall/VPN engines related to VPN performance.

Gateway-to-Gateway VPN

In the SMC, a Virtual Private Network (VPN) element that is set up so that the VPN is established between two gateway devices providing connectivity to networks behind the gateways.

Geolocation

Elements that define a geographical location of an IP address. Used for illustrating networks and network traffic on a map and other informative purposes in the Management Client.

Granted Element

An Element or Security Policy that an administrator has been given permission to edit and install when their Administrator Role would otherwise prevent them from doing so.

Group

A Network Element that includes other elements and represents them all at once in policies and other parts of the configuration. For example, you can define a Group of several WWW-servers, and then use the Group element in policies when you need to make a rule that concerns all of the WWW-servers.

H

Hardware

A category of Tags for Situations. Meant for grouping Situations that detect known vulnerabilities in applications that run on a particular hardware platform.

Hash Signature

A cryptography-related concept that refers to a digital fingerprint associated with a given message and computed with one-way algorithms. Hash signatures are used to secure the integrity of encrypted data, ensuring that no tampering has taken place during transmission. See also Client-to-Gateway VPN (page 278), and SHA-1 (page 296).

Heartbeat

A protocol that the nodes of a Firewall Cluster or Sensor Cluster use to monitor each other and for other tasks that are needed for collaboration between each Node.

High Availability

The implementation of clustering technology, hot standby technology, or general redundancy in a system to increase the availability of an application, service, or network beyond what a single system is capable of providing. Increased availability is achieved by eliminating all single points of failure, with clustering technology providing the highest level of availability.

Host

1) A Network Element that represents any single device that has an IP address.

2) Any device connected to a TCP/IP network, including the Internet, with one or more IP addresses. Hosts are distinguishable from gateways or routers, in that they do not forward, or route, packets to other networks.

Hot Standby

A solution where one node handles the work load with the support of a back-up node, which takes over connections in case of failure in the first node.

Hybrid Authentication

A system using both Asymmetric Encryption and Symmetric Encryption. Asymmetric techniques are used for key management and digital signatures. The symmetric algorithms are used to encrypt the bulk of data with reduced strain on resources.

I

IKE Proposal

The suggested encryption algorithms, authentication methods, hash algorithms, and Diffie-Hellman information in the Security Association (SA) component of an IPsec VPN. The initiator of an IPsec tunnel can make multiple proposals, but the responder only sends one proposal in return. See also Internet Key Exchange (IKE) (page 285) and Security Association (SA) (page 295).

Incident Case

An Element that administrators can use to gather together all the data, actions, system configuration information, and files related to a specific incident of suspicious activity.

Incident History

A collection of all the logs and audit entries that track actions performed in a particular Incident Case window.

Info Panel

A tab in Management Client windows that shows information on the selected element or other object. The Info view shows, for example, the nodes belonging to a selected cluster.

Inherited Rule

A rule either hidden or shown on a grey background in a Security Policy or Template Policy which has been added in a template higher up in the policy hierarchy so that it has been passed down to the security policy or template policy. Inherited rules are enforced just as any other rules, but they can be edited only in the template where the rule was originally added.

Inline Interface

An IPS Engine or Layer 2 Firewall interface that combines together two physical interfaces, enabling the traffic to be routed through as if the engine were an extension of the network cable, but allowing the engine to actively monitor packets and connections and stop them according to its Actions and Inspection Rules.

Insert Point

The place in a Security Policy or Template Policy where new rules can be inserted when no rules have been inserted in that place yet (shown as a green row) or the place in a template policy where rules can be inserted in inheriting policies and template policies (shown as an orange row).

Inspection Rule

The definitions in an Inspection Policy that define options for deeper inspection and reactions to traffic accepted in Actions. The matching in Inspection rules is done based on matching information provided by Situation elements. See also Action (page 273).

Internal Gateway

A McAfee Firewall/VPN engine that is managed by the same Management Server on which the Virtual Private Network (VPN) is being configured.

Internal Network

The networks and network resources that the SMC is protecting.

Internet Key Exchange (IKE)

A protocol defined by the IPsec (IP Security) standard for securely exchanging key-related information between connecting hosts when establishing a Virtual Private Network (VPN).

Internet Service Provider (ISP)

A company that provides Internet connectivity to subscribers.

Intrusion Detection System (IDS)

A system that monitors network traffic for determining, and making administrators aware of data security exploits or attempts by providing logs or other network information. Confer to Intrusion Prevention System (IPS).

Intrusion Prevention System (IPS)

A system that monitors network traffic (like an Intrusion Detection System (IDS)) and has the capability of actively stopping traffic if it is deemed malicious or otherwise unwanted.

IP Address Bound License

A License file that includes the information on the IP address of the component it licenses.

IPComp (IP Payload Compression Protocol)

A protocol used to reduce the size of IP datagrams. Increases the overall communication performance between a pair of communicating gateways by compressing the datagrams, provided the nodes have sufficient computation power, and the communication is over slow or congested links. IPComp is defined in RFC 2393.

IP Splicing (or Hijacking)

An attack performed by intercepting and using an active, established session. Often occurs after the authentication phase of the connection is complete, giving the attacker the permissions of the original, authenticated user. Encryption at the session or network layer is typically the best defense from such an attack.

IP Spoofing

A technique used to obtain unauthorized access to computers by sending connection requests with tampered headers, simulating a trusted source.

IPsec (IP Security)

A set of protocols supporting secure exchange of packets. Used for the implementation of Virtual Private Network (VPN) solutions when high performance and/or support for a wide variety of protocols are needed. IPsec provides transport and tunnel encryption modes. IPsec is defined in RFC 2401.

IPsec Proposal

Suggested encryption algorithms, hash algorithms, authentication methods, etc. to be used for an IPsec (IP Security) tunnel. See also IKE Proposal (page 284).

IPS Cluster

Group of two or more IPS engine nodes that work together as if they were a single IPS.

IPS Engine

1) An IPS component that captures all the traffic from a physical network link, inspects it according to its policy, and if installed inline, selects which connections are allowed to continue.

2) The device that runs the Next Generation Firewall (NGFW) engine software in the IPS role. This can be a standard server, an engine installed on a virtualization platform, or a McAfee NGFW appliance.

IPS Policy

The Security Policy for IPS Engines that contains the Action and Rule definitions that determine how traffic is inspected and how the engine reacts when a match is found.

IPv4 Access Rule

A row in a Firewall or IPS policy that defines how one type of IPv4 connection is handled by providing matching criteria based on the source, destination, and protocol information. Confer to IPv6 Access Rule (page 287).

IPv6 Access Rule

A row in an IPS policy that defines how one type of IPv6 connection is handled by providing matching criteria based on the source, destination, and protocol information. Confer to Action (page 273).

ISAKMP (Internet Security Association Key Management Protocol)

An open-ended encoding protocol necessary for IKE negotiation when establishing Security Associations. See also Security Association (SA) (page 295).

ISP (Internet Service Provider)

See Internet Service Provider (ISP) (page 285).

J

Journal

A tool in the Incident Case window that allows administrators to create a permanent record of their actions while investigating an incident.

Jump Action

A Security Policy parameter that directs the inspection to a Sub-Policy, against which connections matching the rule with the Jump action are checked. Can be used to speed up traffic processing, as connections that do not match the Jump rules are not checked against rules in the sub-policies.

L

Layer 2 Firewall

1) A Layer 2 Firewall component that captures all the traffic from a physical network link, handles it according to its policy, and if installed inline, selects which connections are allowed to continue.

2) The device that runs the Next Generation Firewall (NGFW) engine software in the Layer 2 Firewall role. This can be a standard server, an engine installed on a virtualization platform, or a McAfee NGFW appliance.

License

Files you install in the SMC to tell the Management Server that the components you have installed have been legally purchased. You generate the licenses at the License Center web page and install them on the Management Server using the Management Client.

Lifetime

The interval at which the IPsec participants should begin to negotiate a replacement Security Association (SA) (soft lifetime) or the interval at which the current SA for an IPsec tunnel is no longer valid (hard lifetime) in a Virtual Private Network (VPN).

Load Balancing

A process for distributing work evenly across multiple, available devices to avoid overwhelming any single system.

Load-Balancing Filter

A software component that determines which network connections should be handled by a particular node in a cluster, based on address information, current load, performance of individual machines, and other factors.

Load Balanced Routing

A method for choosing routes to destinations based on determining the fastest response time through multiple gateways. The application of Multi-Link technology to determine which network link provides the best round trip time.

Load Sharing

The distribution of work between multiple devices. Similar to Load Balancing, but not as effective, since the techniques used do not ensure an *equal* distribution of the work load. Load sharing is typically a static method of distributing a load, whereas load balancing is often a dynamic method.

Location

An Element that groups together SMC components that are on the same side of a device doing NAT (Network Address Translation). Used to define Contact Addresses for components that communicate within the SMC.

Logging Options

A selection available in all rules in policies that determines if and how a record is created when the rule matches.

Logging Profile

Defines how the Log Server converts Syslog data received from a particular type of third-party component into SMC log entries.

Log Server

A component of the Security Management Center (SMC) responsible for storing and managing log (and alert) data, and analyzing and correlating events detected by multiple Security Engines.

Log Spool

A temporary storage area in an engine node for log data before it is sent to a Log Server.

Logical Interface

An IPS Element used in the IPS policies to represent one or more physical network interfaces as defined in the Sensor properties.

Logs View

A tool that allows browsing logs, alerts, audit data, and connections each in an adapted version of the same user interface.

Loopback IP address

An optional type of IP address that allows you to assign IP addresses that do not belong to any directly-connected networks to a Single Firewall, Firewall Cluster, or Virtual Firewall. Loopback IP addresses are not connected to any physical interface and they do not create connectivity to any network.

M

Main Mode

An IKE negotiation mode, which exchanges six messages between the end-points of an IPsec tunnel to complete the negotiation of authentication and keys for a Virtual Private Network (VPN). Optionally, Perfect Forward Secrecy (PFS) can be applied to protect further negotiations. See also Aggressive Mode (page 274) and Perfect Forward Secrecy (PFS) (page 292).

Malware

Malicious software designed to infiltrate or damage a computer system.

Management Bound License

A License file for McAfee NGFW engines that is based on information on the Management Server's Proof of License (POL) code.

Management Client

A graphical user interface component that provides the tools for configuring, managing, and monitoring the engines, and other components in the SMC. The Management Client connects to the Management Server to provide these services based on the Administrator information that you use when launching the Management Client software.

Management Network

The network used for communication between firewalls, Management Servers, Log Servers and the Management Client.

Management Server

An SMC component that stores all information about the configurations of all engines, and other components in the SMC, monitors their state, and provides access for Management Clients when administrators want to change the configurations or command the engines. The most important component in the SMC.

Master Engine

A physical engine device that provides resources for Virtual Security Engines.

Maximum Transmission Unit (MTU)

The largest physical size of a datagram that can be transmitted over a network without fragmentation. Often expressed in bytes, it can apply to frames, packets, cells or other media, depending on the underlying topology.

Modem Interface

A Firewall interface that defines the settings of a 3G modem that provides a wireless outbound link for a Single Firewall.

Monitored Element

An SMC server or engine component that is actively polled by the Management Server, so that administrators can keep track of whether it is working or not. All SMC components are monitored by default.

Monitoring Agent

A software component that can be installed on servers in a Server Pool to monitor the server's operation for the purposes of Traffic Management.

Multicast

A technique by which a set of packets are sent to a group of machines sharing a common address. Unlike broadcast, it does not include all machines, and unlike unicast, it usually has more than one member of the group.

Multi-Layer Inspection

A hybrid firewall technology that incorporates the best elements of application level and network level firewalls, with additional technology to enable the secure handling of many connection types.

Multi-Link

Patented technology to connect one site to another, or to the Internet, using more than one network link. Applications of Multi-Link technology include inbound and outbound traffic management for unencrypted as well as VPN traffic. See also Outbound Multi-link (page 291).

N

NAT (Network Address Translation)

A mechanism for assigning local networks a set of IP addresses for internal traffic and another for external traffic. It increases security by hiding internal IP addresses and enables hosts with "invalid" (non-routable) addresses to communicate on the Internet.

NDI

See Node Dedicated IP Address (NDI) (page 291).

NetLink

An Element used for implementing routing of Multi-Link features. NetLinks can represent any IP-based network links (such as ISP routers, xDSL, leased lines, dial-up modems). NetLinks are combined together into an Outbound Multi-link.

Network Element

1) All Elements that represent one or more components that have an IP address, that is, a general category ('Network Elements') for those elements that represent physical devices and networks in the SMC.

2) The Network Element called 'Network' that represents a (sub)network of computers. Used for rules and configurations that are common for all hosts in a specific (sub)network.

Network Scan

A stage of an attack in which the attacker scans the target to enumerate or map the directly-connected network(s).

Next Generation Firewall (NGFW)

The device that runs NGFW software in Firewall, IPS Engine, or Layer 2 Firewall mode. Can be a standard server, an engine installed on a virtualization platform, or a McAfee NGFW appliance. Represented in the SMC by Security Engine elements.

Node

The representation of an individual Security Engine in the SMC.

Node Dedicated IP Address (NDI)

A unique IP address for each machine. The only interface type for Single Firewalls. Not used for operative traffic in Firewall Clusters, IPS engines, and Layer 2 Firewalls. Firewall Clusters use a second type of interface, Cluster Virtual IP Address (CVI), for operative traffic. IPS engines and Layer 2 Firewalls have two types of interfaces for traffic inspection: the Capture Interface and the Inline Interface.

O

Operating System

A category of Tags for Situations. Meant for grouping Situations that detect known vulnerabilities in a particular operating system or applications that run on that operating system.

Outbound Multi-link

An Element used for combining NetLinks for load-balancing outbound traffic. The NetLinks included in a Outbound Multi-link element are frequently tested to determine which is the fastest NetLink for new outbound connections.

P

Packet

A segment of data sent across a network that includes a header with information necessary for the transmission, such as the source and destination IP addresses.

Packet Dispatch

A Cluster Virtual IP Address (CVI) mode in which only one node in the cluster receives packets. This dispatcher node then forwards the packets to the correct node according to Load Balancing, as well as handles traffic as a normal node. The recommended cluster mode for new installations.

Packet Filtering

A method of controlling access to a network, or set of networks, by examining packets for source and destination address information, and permitting those packets to pass, or halting them based on defined rules.

Packet Sniffer

See Sniffer (page 297).

Perfect Forward Secrecy (PFS)

A property of IKE transactions that enhances the secrecy of keys, but requires additional processing overhead. PFS ensures that the distribution of key-related information remains independent from previously existing key material. See also Internet Key Exchange (IKE) (page 285).

Permission Level

The general level of rights that an Administrator has. Permissions are customized with Administrator Roles and Granted Elements.

Permit Action

An Inspection Rule action that stops the inspection of all traffic that matches to the rule that uses the Permit action and lets the traffic continue to its destination.

Phishing

A Social Engineering attack in which a malicious e-mail or web page attempts to solicit sensitive information such as usernames, passwords, and credit card details by masquerading as coming from a trustworthy entity.

Player

Any element or IP address that was involved in an incident that is being investigated using the Incident Case element.

Policy

A container for the Access rules, Inspection rules, and NAT rules.

Policy Routing

User-defined routing based on information that is not normally used in routing, such as the source IP address, port information, or service type.

Policy Snapshot

A record of policy configuration that shows the configuration in the form that it was installed or refreshed, including the rules of the policy, the elements included and their properties, as well as the time when the policy was uploaded, and which administrator performed the upload. Helps in keeping track of configuration changes.

Port Address Translation (PAT)

A process, similar to NAT (Network Address Translation), where the source or destination port is changed to a different port. PAT is often used to disguise, or masquerade a service in place of another. See also NAT (Network Address Translation) (page 290).

Pre-shared Key

A string of characters that is stored on two (or more) systems and that is used for authenticating or encrypting communications between the systems.

Probing Profile

Settings that define how a Log Server monitors third-party components.

Proof of License (POL)

A code used for verifying the legitimate purchase of SMC and NGFW software products. Used for generating License files.

Proof of Serial Number (POS)

Identification code attached to McAfee NGFW appliances.

Protocol

An element that is used inside Service elements to specific a Protocol Agent for the Firewall Actions and the protocol of the traffic for the Inspection Rules.

Protocol Agent

A process on the engines that assists the engine in handling a particular Protocol. Protocol Agents ensure that related connections for a service are properly grouped and evaluated by the engine, as well as assisting the engine with content filtering or network address translation tasks. See also Connection Tracking (page 278).

Protocol Tag

A type for Protocol elements that are only used to define the protocol of traffic for inspection against the inspection rules. Confer to Protocol Agent.

Proxy ARP

Proxy ARP option on a device that does routing means that the device relays broadcast messages between two hosts that are in separate physical networks, but still have IP addresses from the same network. This proxy is needed for the ARP requests, as broadcast messages are not normally relayed from one network to another. See also Address Resolution Protocol (ARP) (page 273).

Pruning

Deleting log entries according to Filters either as the logs arrive on the Log Server or before they are stored (after displaying them in the current view in the Logs view).

Public-key Cryptography

A cryptographic system that uses a pair of keys: a public key, used to encrypt a message, and a private (secret) key that can decrypt the message. This is also called asymmetric encryption.

Q

QoS Class

An Element that works as a link between a rule in a QoS Policy and one or more Firewall Actions. The traffic allowed in the access rule is assigned the QoS Class defined for the rule, and the QoS class is used as the matching criteria for applying QoS Policy rules.

QoS Policy

A set of rules for Bandwidth Management and Traffic Prioritization for traffic that has a particular QoS Class, or rules for assigning QoS Classes based on a DSCP Match found in the traffic.

Refragmentation

A technique to fragment outbound packets from the engine in the same manner in which they were fragmented when the engine received them. See also Virtual Defragmentation (page 301).

Refuse Action

An Action parameter that blocks the packet that matches the rule and sends an error message to the originator of the packet. Confer to Discard Action (page 280).

Regular Expression

A string that describes a set of strings. Used in many text editors and utilities to search for text patterns and, for example, replace them with some other string. In the SMC, regular expressions are used, for example, for defining patterns in traffic that you want a certain Situation to match when you give the Situation a Context that calls for a Regular Expression.

Related Connection

A connection that has a relationship to another connection defined by a Service. For example, the FTP protocol defines a relationship between a control connection, and one or more data connections at the application level. The engine may be required to allow a connection that would otherwise be discarded if it is related to an already allowed connection.

Request for Comments (RFC)

A document that outlines a proposed standard for a protocol. RFCs define how the protocol should function, and are developed by working groups of the Internet Engineering Task Force (IETF), and reviewed and approved by the Internet Engineering Steering Group (IESG). See `http://www.rfc-editor.org/`.

Retained License

A Management Bound License that has been used to install a policy on an engine and has then been unbound without relicensing or deleting the engine the license was bound to. Retained licenses cannot be bound to any engine before the engine the license was previously bound to is deleted or has a new policy refresh with a valid license.

RFC

See Request for Comments (RFC).

Rootkit

A set of tools that intruders to computer systems use for hiding their presence and the traces of their actions.

Route

The set of routers or gateways a packet travels through in order to reach its destination. In TCP/IP networks, individual packets for a connection may travel through different routes to reach the destination host.

Router

A Network Element representing a physical router in your network. Most often used to indicate next-hop routers in the Routing view and in Network Diagrams.

Routing Table

A database maintained on every router and gateway with information on paths to different networks. In the SMC, the routing table is represented graphically in the Routing view.

Rule

An expression used to define the eventual outcome of packets arriving at the engine, which match certain conditions (e.g., source and destination address, protocol, user).

Rules Tree

The main configuration tool for adjusting Inspection Rule definitions.

S

SA (Security Association)

See Security Association (SA) (page 295).

Scan

See Network Scan (page 290).

Secondary IP address

An IP address used for identifying an element with multiple addresses as a source or destination of traffic, defined in addition to a primary IP address.

Secret Key Cryptography

See Symmetric Encryption (page 298).

Security Association (SA)

A unidirectional, logical connection established for securing Virtual Private Network (VPN) communications between two sites. A security association records the information required by one site to support one direction of the IPsec connection whether inbound or outbound. It uses transport mode for communications between two hosts and tunnel mode for communication between VPN gateways. See also Authentication Header (AH) (page 276).

Security Engine

A type of Element that represents an Next Generation Firewall (NGFW) engine device in the SMC. See also Firewall, IPS Engine, and Layer 2 Firewall.

Security Management Center (SMC)

The system consisting of a Management Server, one or more Log Servers and none to several Web Portal Servers that is used to manage the Security Engines, and to store and manage traffic and system-related data.

Security Parameter Index (SPI)

A value used by AH and ESP protocols to help the Firewall Cluster select the security association that will process an incoming packet. See also Authentication Header (AH) (page 276).

Security Policy

The set of templates, policies, and sub-policies together or individually that define what traffic is acceptable and what traffic is unwanted. Policies are defined using the Management Client, stored on the Management Server and installed on Security Engines, which then use their installed version of the policies to determine the appropriate action to take regarding packets in the network.

Sensor

A legacy IPS component that captures all the traffic from a physical network link, inspects it according to its policy, and if installed inline, selects which connections are allowed to continue. Provides data for the Analyzer (see Analyzer (page 275)).

Sensor Cluster

Group of two or more legacy IPS Sensor nodes that work together as if they were a single Sensor.

Server

1) A Network Element representing a physical server in your network. Generally, server elements are only defined to configure a specific server for use with the Security Management Center (SMC) (such as a RADIUS server used for authenticating administrators), but generic Servers can be used in Network Diagrams instead of Host elements to better illustrate the network layout.

2) In a client-server architecture, a computer that is dedicated for running services used by Client computers. The services may include, for example, file storage, e-mail, or web pages.

Server Pool

A Network Element representing a group of Servers. Used for inbound traffic management.

Server Credentials

An element that stores the private key and certificate of an internal server. In TLS inspection, the private key and certificate allow the engine to decrypt TLS traffic for which the internal server is the destination. The certificate can also be used to secure a Web Portal Server's connections using HTTPS or to authenticate an Authentication Server to the client in CHAPv2 RADIUS authentication. See Client Protection Certificate Authority (page 278).

Service

An Element that is used for matching traffic to an application level protocol, for example, FTP, HTTP or SMTP. The TCP and UDP Services also determine the port number. Service elements are used in policies to make the rule match only a particular protocol, to enable Protocol Agents, and select traffic to be matched against Inspection Rules.

Session Stealing

See IP Splicing (or Hijacking) (page 286).

SHA-1

A cryptographic algorithm used for hash functions. It generates a 160-bit signature from an input of any length. See also Hash Signature (page 284).

Single Firewall

A firewall that has only one Firewall Engine.

Single Point of Failure

The point at which the failure of a single device or component of a system will lead to either the failure of the entire system, or the inability to use services normally provided by that system. Redundant systems, using high availability technologies, eliminate single points of failure.

Site

A set of resources protected by the SMC.

Situation

1) An Element that identifies and describes detected events in the traffic or in the operation of the system. Situations contain the Context information, i.e., a pattern that the system is to look for in the inspected traffic.

2) An Inspection Rule cell where Situation elements are inserted.

Situation Type

A category of Tags for Situations. Meant for indicating what kind of events the associated Situations detect (for example, Attacks, Suspicious Traffic).

Sniffer

A device or program that captures data traveling over a network. Sniffers are often used for troubleshooting network problems, as they can show the packet flow taking place. They can also be used maliciously to steal data off a network.

SNMP Agent

A software component that sends SNMP traps when specific events are encountered.

Social Engineering

An attack involving trickery or deception for the purpose of manipulating people into performing actions or divulging confidential information.

SPI (Security Parameter Index)

See Security Parameter Index (SPI) (page 295).

SSH (Secure Shell)

A program to log into another computer over a network, to execute commands in a remote machine, and to move files from one machine to another. It provides strong authentication and secure communications over insecure channels. Often used as a replacement for insecure programs such as `telnet` or `rsh`. In the SMC, SSH can be used for remotely accessing the engine command line.

SSL VPN

A VPN technology that utilizes SSL encryption to secure users’ remote access to specific applications. Allow authenticated users to establish secure connections to a limited number of specific internal services through a standard web browser (“clientless” access) or through a client application that allows a wider range of services.

Standby Mode

An operating state of a Security Engine cluster that keeps one node online and the rest in standby, so that State Synchronization is done, but node does not process the traffic. If the online node is taken offline or fails, one of the standby nodes takes over the existing connections.

State Synchronization

The communication of connection tracking information between several Firewall nodes in a cluster. Can be either a full synchronization, where all connection tracking information is transferred to the other nodes of a cluster, or an incremental synchronization, where only the information on connections changed after the last synchronization are transferred. See also Connection Tracking (page 278).

Static IP address

IP address that is typed in by a user or an administrator, and which does not change without their action.

Static NAT

NAT (Network Address Translation) where for each original address, there is a single, predefined translated address.

Static Routing

A form of routing that has permanent routes between networks programmed into every Routing Table.

Sub-Policy

A set of rules that are separated from the main policy, based on some common category, such as the service or the destination IP address. In this way, related rules can be grouped together to make the entire policy easier to understand. Because subrules are only processed if the general rule in the main policy matches, the overall processing time is improved.

Subtunnel

The actual tunnels that are combined logically within a multi-route VPN tunnel in a Multi-Link environment in the SMC. They represent all possible routes that connect the end-points of the VPN gateways between which a Virtual Private Network (VPN) is formed. The individual subtunnels may connect the two gateways through different network links.

Symmetric Encryption

An Encryption mechanism that uses the same shared secret key for encrypting and decrypting messages. It is often referred to as symmetric bulk encryption since it processes large amounts of data rather quickly. Also known as conventional or secret key cryptography. There are two main types of symmetric encryption algorithms, bulk and stream encryption (also known as block ciphers and stream ciphers). Common symmetric algorithms are DES and 3DES. See also Asymmetric Encryption (page 275).

Syslog

A standard protocol for exchanging logs between network components. Defined in RFC 5424.

System Summary

A panel in the System Status view that provides a general summary view of the current status of the monitored elements according to the component type.

T

Tag

An Element for organizing Situations. Tags can also be used in Inspection Rules, in the Situation cell, to represent all Situations marked with that Tag.

Takeover Period

The time interval during which the active nodes in a Security Engine cluster collaborate to redistribute the work load of a failed node.

Task

An Element that allows you to schedule commands to run automatically at a convenient time.

Template Policy

A combination of rules and Insert Points, which is used as a basis when creating policies or other template policies. Policies and template policies created from a particular template policy then inherit all the rules from that template policy and any of the template policies higher up in the inheritance hierarchy. The Inherited Rules cannot be edited within the inheriting policy. Used, for example, by high-privilege Administrators to restrict changes administrators with a lower Administrator Role can make to rules.

Temporary Filter

A log filter that is created from details of entries in the Logs View or the Connections view, and which is only available until the view is closed.

Terminate Action

An Inspection Rule parameter that stops or attempts to stop the connection matching to the rule according to the Action Option selected and the whether the Security Engine where the rule matching occurs is capable of stopping the connection.

Tester

A tool that can automatically run tests on Next Generation Firewall (NGFW) engines to check system or network operation and take action based on the results of those tests.

Timeline

A tool in the Logs View that allows you to select and change the time range for the logs that are displayed.

ToS Flag

A data field in IP packet headers that provides a number representing the type of the service the packet is a part of. The ToS flag is used for Traffic Prioritization and is also know as DSCP (DiffServ Code Point).

Traffic Handler

The set of Network Elements used for inbound and outbound traffic management. Includes NetLinks, Outbound Multi-links, and Server Pools.

Traffic Management

The control, definition, and management of how packets or connections should flow through firewalls, routers, network links, VPNs or other gateway objects, based on load balancing, clusters, availability of links and more.

Traffic Prioritization

The process of assigning traffic a priority value, which is used to determine the order in which queued packets are sent forward, overriding the standard first-come-first-served operation of network devices. Used for assuring Quality of Service (QoS) for time-critical connections. Can be used together with Bandwidth Management or on its own. See also DSCP (DiffServ Code Point) (page 281), QoS Class (page 293) and QoS Policy (page 293).

Transparent Access Control Mode

A Security Engine configuration in which the IPS Engine or Layer 2 Firewall examines Ethernet traffic according to the Ethernet Rules.

Transparent Proxy

A technique whereby a connection is routed to a proxy server, which then establishes a second connection to the original destination host, but the entire transaction takes place without notifying the user, or requiring the user to perform any additional actions.

Transport Protocol

Any protocol that communicates and functions on the transport layer of the TCP/IP protocol stack. These protocols function above the network layer, and are usually responsible for error correction, quality of service, and other characteristics not handled by the network layer. TCP, UDP, and IPsec are common examples of transport protocols.

Tunneling

A technology that enables one network to send its data through another, perhaps dissimilar, network. Tunneling works by encapsulating, or packaging, a network protocol within packets carried by the second network.

U

Use IPsec VPN Action

A Firewall Action parameter that directs traffic matching to the rule to a VPN. Can be either an Apply VPN Action or an Enforce VPN Action.

UDP Tracking

Information maintained by the Firewall engines to group together UDP requests and replies, handling them as a single virtual connection. See also Virtual Connection Tracking (page 301).

URL Filtering

A feature that compares the URLs that users attempt to open to a list of URLs to prevent users from intentionally or accidentally accessing most websites that are objectionable or potentially harmful.

User

An Element that defines an end-user in your network. Used for defining Authentication with or without Client-to-Gateway VPN access. Confer to Administrator (page 273).

User Response

Defines additional notification actions for rule matches, such as redirecting access to a forbidden URL to a page on an internal web server instead.

UTM (Unified Threat Management)

A device that combines different types of traffic filtering in one physical appliance. The features offered in a UTM device vary greatly from vendor to vendor. The UTM solution comprises a Firewall, deep packet inspection (IDS), and anti-virus.

V

Virtual Adapter

A component of the Stonesoft IPsec VPN Client, or a third-party VPN client, that allows using a second, Virtual IP address for Virtual Private Network (VPN) traffic. Shown as a network adapter in the operating system.

Virtual Connection Tracking

A superset of UDP tracking, ICMP tracking, etc. A technology that is used by the Firewall engines for connectionless network protocols like UDP and ICMP. The Firewall engines keep track of virtual connections by grouping together packets that are related, based on information in the packet headers. See also Related Connection (page 294).

Virtual Defragmentation

A procedure in which incoming packet fragments are collected. The packet is defragmented for processing by the engine, and refragmented before it is transmitted again. See also Fragmentation (page 283).

Virtual Firewall

A Virtual Security Engine in the Firewall role.

Virtual IP address

A second IP address that is given to a VPN Client that has a Virtual Adapter enabled, and that is connecting to a VPN gateway using Client-to-Gateway VPN. A virtual IP address enables the use of certain services that require the client to have an IP address belonging to a specific address range, while enabling it to retain its primary IP address for maintaining other connections. The Virtual IP address for Stonesoft IPsec VPN Clients is always assigned by DHCP (Dynamic Host Configuration Protocol).

Virtual IPS

A Virtual Security Engine in the IPS role.

Virtual Layer 2 Firewall

A Virtual Security Engine in the Layer 2 Firewall role.

Virtual Local Area Network (VLAN)

A local area network which is defined through software in a switch or other networking device, rather than by the more traditional hardware division.

Virtual Private Network (VPN)

Refers to a confidential connection that is established through unsecured networks by the means of authentication, encryption, and integrity checking. The two major VPN technologies are IPsec (IP Security), which is better suited when a wide variety of network services and large traffic volumes are involved, and SSL VPN, which is used to provide access to a limited number of services to individual users without client-side device configuration.

Virtual Resource

An element that defines the set of resources on the Master Engine that are allocated to each Virtual Security Engine.

Virtual Security Engine

Logically-separate engines that run as virtual engine instances on a Master Engine. See also Virtual Firewall, Virtual IPS, and Virtual Layer 2 Firewall.

VPN Client

Software that can be used to establish a Virtual Private Network (VPN) with a VPN gateway device to securely access remote resources over insecure networks.

VPN Gateway

A device, typically a firewall, that performs encryption or decryption on Virtual Private Network (VPN) packets sent between Sites through untrusted networks.

VPN Profile

An element that defines the IPsec (IP Security)-related settings for one or more VPNs.

Vulnerability

An IPS element that contains information on a publicly known flaw that affects the security of some system. Vulnerabilities are attached to Situations to provide you more information on what has happened when the Situation matches.

W

Web Portal

Browser-based service that allows users to view logs, Policy Snapshots, and reports.

Whitelisting

The process of exempting specific traffic from being blocked by Blacklisting or URL Filtering.

Index