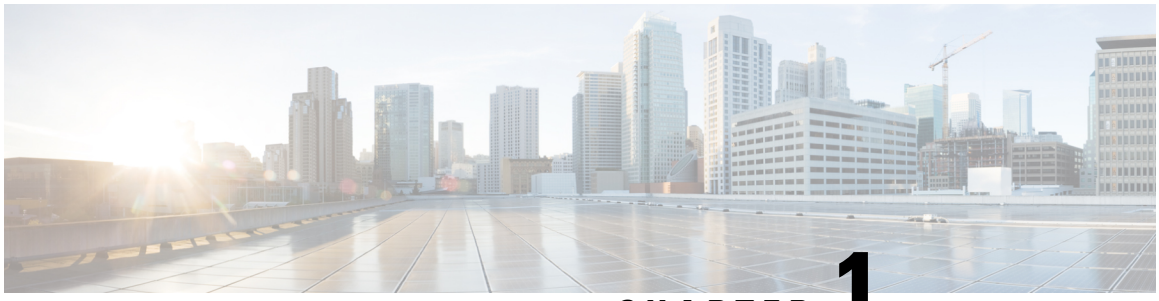
Custom Snort 2 Intrusion Policies for Access Control

First Published: 2026-02-24

Last Modified: 2026-02-24

Americas Headquarters
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
http://www.cisco.com
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 527-0883

CONTENTS



CHAPTER 1

Intrusion Prevention Overview

This chapter provides an overview of the Snort 2 inspection engine, and the network analysis and intrusion policies. It also provides an insight into system-provided and custom network analysis and intrusion policies. In most cases, the system-provided policies are all you need. For advanced users with specific needs, you can create custom rules.

Note .

To configure custom Snort 3 rules, see *Custom Snort 3 Intrusion Policies for Access Control*.

Network Analysis and Intrusion Policy Basics

Network analysis and intrusion policies work together as part of the system's intrusion detection and prevention feature.

- The term *intrusion detection* generally refers to the process of passively monitoring and analyzing network traffic for potential intrusions and storing attack data for security analysis. This is sometimes referred to as "IDS."
- The term *intrusion prevention* includes the concept of intrusion detection, but adds the ability to block or alter malicious traffic as it travels across your network. This is sometimes referred to as "IPS."

Note

- You must configure Network Analysis Policy (NAP) in **Prevention** mode if you are using Snort 3 and SSL decryption or TLS Server Identity. The SSL functionality does not work when Snort 3 NAP is in detection mode.
- We strongly recommend that your intrusion policy (IPS) and network analysis policy (NAP) have the same settings. If IPS is in detection mode, set the NAP in detection mode, and conversely as well.

In an intrusion prevention deployment, when the system examines packets:

- A **network analysis policy** governs how traffic is *decoded* and *preprocessed* so it can be further evaluated, especially for anomalous traffic that might signal an intrusion attempt.
- An **intrusion policy** uses *intrusion and preprocessor rules* (sometimes referred to collectively as *intrusion rules*) to examine the decoded packets for attacks based on patterns. Intrusion policies are paired with *variable sets*, which allow you to use named values to accurately reflect your network environment.

Both network analysis and intrusion policies are invoked by a parent access control policy, but at different times. As the system analyzes traffic, the network analysis (decoding and preprocessing) phase occurs before and separately from the intrusion prevention (additional preprocessing and intrusion rules) phase. Together, network analysis and intrusion policies provide broad and deep packet inspection. They can help you detect, alert on, and protect against network traffic that could threaten the availability, integrity, and confidentiality of hosts and their data.

The system is delivered with several similarly named network analysis and intrusion policies (for example, Balanced Security and Connectivity) that complement and work with each other. By using system-provided policies, you can take advantage of the experience of the Talos Intelligence Group. For these policies, Talos sets intrusion and preprocessor rule states, as well as provides the initial configurations for preprocessors and other advanced settings.

You can also create custom network analysis and intrusion policies. You can tune settings in custom policies to inspect traffic in the way that matters most to you so that you can improve both the performance of your managed devices and your ability to respond effectively to the events they generate.

You create, edit, save, and manage network analysis and intrusion policies using similar policy editors in the web interface. When you are editing either type of policy, a navigation panel appears on the left side of the web interface; the right side displays various configuration pages.

Attention

Detection mode deprecation: From management center Version 7.4.0, for a network analysis policy (NAP), the **Detection** inspection mode is deprecated and will be removed in an upcoming release.

The **Detection** mode was intended to be used as a test mode so that you can enable inspections and see how they behave in your network before setting it to drop traffic, that is, to show traffic that would be dropped.

This behavior is improved where all inspector drops are controlled by the rule state, and you can set each one to generate events. This is done to test them before configuring the rule state to drop traffic. As we now have granular control over traffic drops in Snort 3, the **Detection** mode only adds more complexity to the product and is not needed, so the detection mode is deprecated.

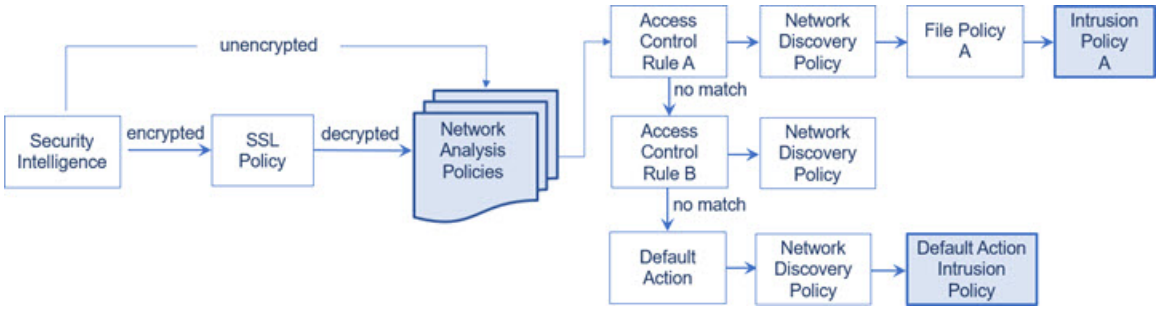
If you change a NAP in **Detection** mode to **Prevention**, the NAP that processes the traffic of intrusion events and have the result "will be dropped" will now be "dropped" and the corresponding traffic will drop the traffic from these events. This is applicable for rules whose GIDs are not 1 or 3. GIDs 1 and 3 are text/compiled rules (typically provided by Talos or from your custom/imported rules) and all other GIDs are inspections for anomalies. These are more uncommon rules to trigger in a network. Changing to **Prevention** mode is unlikely to have any impact on the traffic. You need to just disable the intrusion rule that is applicable for the dropped traffic and set it to just generate or disable.

We recommend you choose **Prevention** as the inspection mode, but if you choose **Prevention**, you cannot revert to **Detection** mode.

How Policies Examine Traffic For Intrusions

When the system analyzes traffic as part of your access control deployment, the network analysis (decoding and preprocessing) phase occurs before and separately from the intrusion prevention (intrusion rules and advanced settings) phase.

The following diagram shows, in a simplified fashion, the order of traffic analysis in an inline, intrusion prevention and malware defense deployment. It illustrates how the access control policy invokes other policies to examine traffic, and in which order those policies are invoked. The network analysis and intrusion policy selection phases are highlighted.



In an inline deployment (that is, where relevant configurations are deployed to devices using routed, switched, or transparent interfaces, or inline interface pairs), the system can block traffic without further inspection at almost any step in the illustrated process. Security Intelligence, the SSL policy, network analysis policies, file policies, and intrusion policies can all either drop or modify traffic. Only the network discovery policy, which passively inspects packets, cannot affect the flow of traffic.

Similarly, at each step of the process, a packet could cause the system to generate an event. Intrusion and preprocessor events (sometimes referred to collectively as *intrusion events*) are indications that a packet or its contents may represent a security risk.

Tip The diagram does not reflect that access control rules handle encrypted traffic when your SSL inspection configuration allows it to pass, or if you do not configure SSL inspection. By default, the system disables intrusion and file inspection of encrypted payloads. This helps reduce false positives and improve performance when an encrypted connection matches an access control rule that has intrusion and file inspection configured.

Note that for a single connection, although the system selects a network analysis policy before an access control rule as shown in the diagram, some preprocessing (notably application layer preprocessing) occurs after access control rule selection. This does **not** affect how you configure preprocessing in custom network analysis policies.

Decoding, Normalizing, and Preprocessing: Network Analysis Policies

Without decoding and preprocessing, the system could not appropriately evaluate traffic for intrusions because protocol differences would make pattern matching impossible. Network analysis policies govern these traffic-handling tasks:

- **after** traffic is filtered by Security Intelligence
- **after** encrypted traffic is decrypted by an optional SSL policy
- **before** traffic can be inspected by file or intrusion policies

A network analysis policy governs packet processing in phases. First the system decodes packets through the first three TCP/IP layers, then continues with normalizing, preprocessing, and detecting protocol anomalies:

- The packet decoder converts packet headers and payloads into a format that can be easily used by the preprocessors and later, intrusion rules. Each layer of the TCP/IP stack is decoded in turn, beginning with the data link layer and continuing through the network and transport layers. The packet decoder also detects various anomalous behaviors in packet headers.
- In inline deployments, the inline normalization preprocessor reformats (normalizes) traffic to minimize the chances of attackers evading detection. It prepares packets for examination by other preprocessors and intrusion rules, and helps ensure that the packets the system processes are the same as the packets received by the hosts on your network.

Note In a passive deployment, Cisco recommends that you enable adaptive profile updates at the access control policy level, instead of inline normalization at the network analysis level.

- Various network and transport layers preprocessors detect attacks that exploit IP fragmentation, perform checksum validation, and perform TCP and UDP session preprocessing.

  Note that some advanced transport and network preprocessor settings apply globally to all traffic handled by the target devices of an access control policy. You configure these in the access control policy rather than in a network analysis policy.

- Various application-layer protocol decoders normalize specific types of packet data into formats that the intrusion rules engine can analyze. Normalizing application-layer protocol encodings allows the system to effectively apply the same content-related intrusion rules to packets whose data is represented differently, and to obtain meaningful results.
- The Modbus, DNP3, CIP, and s7commplus SCADA preprocessors detect traffic anomalies and provide data to intrusion rules. Supervisory Control and Data Acquisition (SCADA) protocols monitor, control, and acquire data from industrial, infrastructure, and facility processes such as manufacturing, production, water treatment, electric power distribution, airport and shipping systems, and so on.
- Several preprocessors allow you to detect specific threats, such as Back Orifice, portscans, SYN floods and other rate-based attacks.

  Note that you configure the sensitive data preprocessor, which detects sensitive data such as credit card numbers and Social Security numbers in ASCII text, in intrusion policies.

In a newly created access control policy, one default network analysis policy governs preprocessing for *all* traffic for *all* intrusion policies invoked by the same parent access control policy. Initially, the system uses the Balanced Security and Connectivity network analysis policy as the default, but you can change it to another system-provided or custom network analysis policy. In a more complex deployment, advanced users can tailor traffic preprocessing options to specific security zones, networks, and VLANs by assigning different custom network analysis policies to preprocess matching traffic.

Note For an access control policy with rule action as **Trust** and a prefilter rule with action as **Fastpath** with logging options disabled, you will observe that the end-of-flow events are still generated in the system. The events are not visible on the management center event pages.

Access Control Rules: Intrusion Policy Selection

After initial preprocessing, access control rules (when present) evaluate traffic. In most cases, the first access control rule that a packet matches is the rule that handles that traffic; you can monitor, trust, block, or allow matching traffic.

When you allow traffic with an access control rule, the system can inspect the traffic for discovery data, malware, prohibited files, and intrusions, in that order. Traffic not matching any access control rule is handled by the access control policy's default action, which can also inspect for discovery data and intrusions.

Note All packets, **regardless** of which network analysis policy preprocesses them, are matched to configured access control rules—and thus are potentially subject to inspection by intrusion policies—in top-down order.

The diagram in How Policies Examine Traffic For Intrusions, on page 3 shows the flow of traffic through a device in an inline, intrusion prevention and malware defense deployment, as follows:

- Access Control Rule A allows matching traffic to proceed. The traffic is then inspected for discovery data by the network discovery policy, for prohibited files and malware by File Policy A, and then for intrusions by Intrusion Policy A.
- Access Control Rule B also allows matching traffic. However, in this scenario, the traffic is not inspected for intrusions (or files or malware), so there are no intrusion or file policies associated with the rule. Note

that by default, traffic that you allow to proceed is inspected by the network discovery policy; you do not need to configure this.

- In this scenario, the access control policy's default action allows matching traffic. The traffic is then inspected by the network discovery policy, and then by an intrusion policy. You can (but do not have to) use a different intrusion policy when you associate intrusion policies with access control rules or the default action.

The example in the diagram does not include any blocking or trusting rules because the system does not inspect blocked or trusted traffic.

Intrusion Inspection: Intrusion Policies, Rules, and Variable Sets

You can use intrusion prevention as the system's last line of defense before traffic is allowed to proceed to its destination. Intrusion policies govern how the system inspects traffic for security violations and, in inline deployments, can block or alter malicious traffic. The main function of intrusion policies is to manage which intrusion and preprocessor rules are enabled and how they are configured.

Intrusion and Preprocessor Rules

An intrusion rule is a specified set of keywords and arguments that detects attempts to exploit vulnerabilities on your network; the system uses an intrusion rule to analyze network traffic to check if it matches the criteria in the rule. The system compares packets against the conditions specified in each rule and, if the packet data matches all the conditions specified in a rule, the rule triggers.

The system includes the following types of rules created by Talos Intelligence Group:

- *shared object intrusion rules*, which are compiled and cannot be modified (except for rule header information such as source and destination ports and IP addresses)
- *standard text intrusion rules*, which can be saved and modified as new custom instances of the rule.
- *preprocessor rules*, which are rules associated with preprocessors and packet decoder detection options in the network analysis policy. You cannot copy or edit preprocessor rules. Most preprocessor rules are disabled by default; you must enable them to use preprocessors to generate events and, in an inline deployment, drop offending packets.

When the system processes packets according to an intrusion policy, first a rule optimizer classifies all activated rules in subsets based on criteria such as: transport layer, application protocol, direction to or from the protected network, and so on. Then, the intrusion rules engine selects the appropriate rule subsets to apply to each packet. Finally, a multi-rule search engine performs three different types of searches to determine if the traffic matches the rule:

- The protocol field search looks for matches in particular fields in an application protocol.
- The generic content search looks for ASCII or binary byte matches in the packet payload.
- The packet anomaly search looks for packet headers and payloads that, rather than containing specific content, violate well-established protocols.

In a custom intrusion policy, you can tune detection by enabling and disabling rules, as well as by writing and adding your own standard text rules. You can also use Cisco recommendations to associate the operating systems, servers, and client application protocols detected on your network with rules specifically written to protect those assets.

Variable Sets

Whenever the system uses an intrusion policy to evaluate traffic, it uses an associated *variable set*. Most variables in a set represent values commonly used in intrusion rules to identify source and destination IP addresses and ports. You can also use variables in intrusion policies to represent IP addresses in rule suppressions and dynamic rule states.

The system provides a single default variable set, which is comprised of predefined default variables. Most system-provided shared object rules and standard text rules use these predefined default variables to define networks and port numbers. For example, the majority of the rules use the variable `$HOME_NET` to specify the protected network and the variable `$EXTERNAL_NET` to specify the unprotected (or outside) network. In addition, specialized rules often use other predefined variables. For example, rules that detect exploits against web servers use the `$HTTP_SERVERS` and `$HTTP_PORTS` variables.

Tip Even if you use system-provided intrusion policies, Cisco **strongly** recommends that you modify key default variables in the default set. When you use variables that accurately reflect your network environment, processing is optimized and the system can monitor relevant systems for suspicious activity. Advanced users can create and use custom variable sets for pairing with one or more custom intrusion policies.

Intrusion Event Generation

When the system identifies a possible intrusion, it generates an *intrusion or preprocessor event* (sometimes collectively called *intrusion events*). Managed devices transmit their events to the Firewall Management Center, where you can view the aggregated data and gain a greater understanding of the attacks against your network assets. In an inline deployment, managed devices can also drop or replace packets that you know to be harmful.

Each intrusion event in the database includes an event header and contains information about the event name and classification; the source and destination IP addresses; ports; the process that generated the event; and the date and time of the event, as well as contextual information about the source of the attack and its target. For packet-based events, the system also logs a copy of the decoded packet header and payload for the packet or packets that triggered the event.

The packet decoder, the preprocessors, and the intrusion rules engine can all cause the system to generate an event. For example:

- If the packet decoder (configured in the network analysis policy) receives an IP packet that is less than 20 bytes, which is the size of an IP datagram without any options or payload, the decoder interprets this as anomalous traffic. If, later, the accompanying decoder rule in the intrusion policy that examines the packet is enabled, the system generates a preprocessor event.
- If the IP defragmentation preprocessor encounters a series of overlapping IP fragments, the preprocessor interprets this as a possible attack and, when the accompanying preprocessor rule is enabled, the system generates a preprocessor event.
- Within the intrusion rules engine, most standard text rules and shared object rules are written so that they generate intrusion events when triggered by packets.

As the database accumulates intrusion events, you can begin your analysis of potential attacks. The system provides you with the tools you need to review intrusion events and evaluate whether they are important in the context of your network environment and your security policies.

System-Provided and Custom Network Analysis and Intrusion Policies

Creating a new access control policy is one of the first steps in managing traffic flow using the system. By default, a newly created access control policy invokes system-provided network analysis and intrusion policies to examine traffic.

The following diagram shows how a newly created access control policy in an inline, intrusion-prevention deployment initially handles traffic. The preprocessing and intrusion prevention phases are highlighted.

Figure 1: New Access Control Policy: Intrusion Prevention

Note how:

- A default network analysis policy governs the preprocessing of *all* traffic handled by the access control policy. Initially, the system-provided *Balanced Security and Connectivity network analysis policy* is the default.
- The default action of the access control policy allows all non-malicious traffic, as determined by the system-provided *Balanced Security and Connectivity intrusion policy*. Because the default action allows traffic to pass, the discovery feature can examine it for host, application, and user data before the intrusion policy can examine and potentially block malicious traffic.
- The policy uses default Security Intelligence options (global Block and Do Not Block lists only), does not decrypt encrypted traffic with an SSL policy, and does not perform special handling and inspection of network traffic using access control rules.

A simple step you can take to tune your intrusion prevention deployment is to use a different set of system-provided network analysis and intrusion policies as your defaults. Cisco delivers several pairs of these policies with the system.

Or, you can tailor your intrusion prevention deployment by creating and using custom policies. You may find that the preprocessor options, intrusion rule, and other advanced settings configured in those policies do not address the security needs of your network. By tuning your network analysis and intrusion policies you can configure, at a very granular level, how the system processes and inspects the traffic on your network for intrusions.

System-Provided Network Analysis and Intrusion Policies

Cisco delivers several pairs of network analysis and intrusion policies with the system. By using system-provided network analysis and intrusion policies, you can take advantage of the experience of the Talos Intelligence Group. For these policies, Talos provides intrusion and preprocessor rule states as well as initial configurations for preprocessors and other advanced settings.

No system-provided policy covers every network profile, traffic mix, or defensive posture. Each covers common cases and network setups that provide a starting point for a well-tuned defensive policy. Although you can use system-provided policies as-is, Cisco strongly recommends that you use them as the base for custom policies that you tune to suit your network.

Tip Even if you use system-provided network analysis and intrusion policies, you should configure the system's intrusion variables to accurately reflect your network environment. At a minimum, modify key default variables in the default set.

As new vulnerabilities become known, Talos releases intrusion rule updates (also known as *Snort Rule Updates*). These rule updates can modify any system-provided network analysis or intrusion policy, and can provide new and updated intrusion rules and preprocessor rules, modified states for existing rules, and modified default policy settings. Rule updates may also delete rules from system-provided policies and provide new rule categories, as well as modify the default variable set.

If a rule update affects your deployment, the web interface marks affected intrusion and network analysis policies as out of date, as well as their parent access control policies. You must re-deploy an updated policy for its changes to take effect.

For your convenience, you can configure rule updates to automatically re-deploy affected intrusion policies, either alone or in combination with affected access control policies. This allows you to easily and automatically keep your deployment up-to-date to protect against recently discovered exploits and intrusions.

To ensure up-to-date preprocessing settings, you **must** re-deploy access control policies, which also deploys any associated SSL, network analysis, and file policies that are different from those currently running, and can also can update default values for advanced preprocessing and performance options.

Cisco delivers the following network analysis and intrusion policies with the system:

Balanced Security and Connectivity network analysis and intrusion policies

These policies are built for both speed and detection. Used together, they serve as a good starting point for most organizations and deployment types. The system uses the Balanced Security and Connectivity policies and settings as defaults in most cases.

Connectivity Over Security network analysis and intrusion policies

These policies are built for organizations where connectivity (being able to get to all resources) takes precedence over network infrastructure security. The intrusion policy enables far fewer rules than those enabled in the Security over Connectivity policy. Only the most critical rules that block traffic are enabled.

Security Over Connectivity network analysis and intrusion policies

These policies are built for organizations where network infrastructure security takes precedence over user convenience. The intrusion policy enables numerous network anomaly intrusion rules that could alert on or drop legitimate traffic.

Maximum Detection network analysis and intrusion policies

These policies are built for organizations where network infrastructure security is given even more emphasis than is given by the Security Over Connectivity policies, with the potential for even greater operational impact. For example, the intrusion policy enables rules in a large number of threat categories including malware, exploit kit, old and common vulnerabilities, and known in-the-wild exploits.

No Rules Active intrusion policy

In the No Rules Active intrusion policy, all intrusion rules, and all advanced settings except intrusion rule thresholds, are disabled. This policy provides a starting point if you want to create your own intrusion policy instead of basing it on the enabled rules in one of the other system-provided policies.

Note Depending on the system-provided base policy that is selected, the settings of the policy vary. To view the policy settings, click the **Edit** icon next to the policy and then click the **Manage Base Policy** link.

Benefits of Custom Network Analysis and Intrusion Policies

You may find that the preprocessor options, intrusion rules, and other advanced settings configured in the system-provided network analysis and intrusion policies do not fully address the security needs of your organization.

Building custom policies can improve the performance of the system in your environment and can provide a focused view of the malicious traffic and policy violations occurring on your network. By creating and tuning custom policies you can configure, at a very granular level, how the system processes and inspects the traffic on your network for intrusions.

All custom policies have a base policy, also called a base layer, which defines the default settings for all configurations in the policy. A layer is a building block that you can use to efficiently manage multiple network analysis or intrusion policies.

In most cases, you base custom policies on system-provided policies, but you can use another custom policy. However, all custom policies have a system-provided policy as the eventual base in a policy chain. Because rule updates can modify system-provided policies, importing a rule update may affect you even if you are using a custom policy as your base. If a rule update affects your deployment, the web interface marks affected policies as out of date.

Benefits of Custom Network Analysis Policies

By default, one network analysis policy preprocessors all unencrypted traffic handled by the access control policy. That means that all packets are decoded and preprocessed according to the same settings, regardless of the intrusion policy (and therefore intrusion rule set) that later examines them.

Initially, the system-provided Balanced Security and Connectivity network analysis policy is the default. A simple way to tune preprocessing is to create and use a custom network analysis policy as the default.

Tuning options available vary by preprocessor, but some of the ways you can tune preprocessors and decoders include:

- You can disable preprocessors that do not apply to the traffic you are monitoring. For example, the HTTP Inspect preprocessor normalizes HTTP traffic. If you are confident that your network does not include any web servers using Microsoft Internet Information Services (IIS), you can disable the preprocessor option that looks for IIS-specific traffic and thereby reduce system processing overhead.

Note If you disable a preprocessor in a custom network analysis policy, but the system needs to use that preprocessor to later evaluate packets against an enabled intrusion or preprocessor rule, the system automatically enables and uses the preprocessor although the preprocessor remains disabled in the network analysis policy web interface.

- Specify ports, where appropriate, to focus the activity of certain preprocessors. For example, you can identify additional ports to monitor for DNS server responses or encrypted SSL sessions, or ports on which you decode telnet, HTTP, and RPC traffic.

For advanced users with complex deployments, you can create multiple network analysis policies, each tailored to preprocess traffic differently. Then, you can configure the system to use those policies to govern the preprocessing of traffic using different security zones, networks, or VLANs.

Note Tailoring preprocessing using custom network analysis policies—especially multiple network analysis policies—is an advanced task. Because preprocessing and intrusion inspection are so closely related, you **must** be careful to allow the network analysis and intrusion policies examining a single packet to complement each other.

Benefits of Custom Intrusion Policies

In a newly created access control policy initially configured to perform intrusion prevention, the default action allows all traffic, but first inspects it with the system-provided Balanced Security and Connectivity intrusion policy. Unless you add access control rules or change the default action, all traffic is inspected by that intrusion policy.

To customize your intrusion prevention deployment, you can create multiple intrusion policies, each tailored to inspect traffic differently. Then, configure an access control policy with rules that specify which policy inspects which traffic. Access control rules can be simple or complex, matching and inspecting traffic using multiple criteria including security zone, network or geographical location, VLAN, port, application, requested URL, or user.

The main function of intrusion policies is to manage which intrusion and preprocessor rules are enabled and how they are configured, as follows:

- Within each intrusion policy, you should verify that all rules applicable to your environment are enabled, and improve performance by disabling rules that are not applicable to your environment. In an inline deployment, you can specify which rules should drop or modify malicious packets.
- Cisco recommendations allow you to associate the operating systems, servers, and client application protocols detected on your network with rules specifically written to protect those assets.
- You can modify existing rules and write new standard text rules as needed to catch new exploits or to enforce your security policies.

Other customizations you might make to an intrusion policy include:

- The sensitive data preprocessor detects sensitive data such as credit card numbers and Social Security numbers in ASCII text. Note that other preprocessors that detect specific threats (back orifice attacks, several portscan types, and rate-based attacks that attempt to overwhelm your network with excessive traffic) are configured in network analysis policies.
- Global thresholds cause the system to generate events based on how many times traffic matching an intrusion rule originates from or is targeted to a specific address or address range within a specified time period. This helps prevent the system from being overwhelmed with a large number of events.
- Suppressing intrusion event notifications and setting thresholds for individual rules or entire intrusion policies can also can prevent the system from being overwhelmed with a large number of events.
- In addition to the various views of intrusion events within the web interface, you can enable logging to syslog facilities or send event data to an SNMP trap server. Per policy, you can specify intrusion event notification limits, set up intrusion event notification to external logging facilities, and configure external responses to intrusion events. Note that in addition to these per-policy alerting configurations, you can

globally enable or disable email alerting on intrusion events for each rule or rule group. Your email alert settings are used regardless of which intrusion policy processes a packet.

Limitations of Custom Policies

Because preprocessing and intrusion inspection are so closely related, you **must** be careful that your configuration allows the network analysis and intrusion policies processing and examining a single packet to complement each other.

By default, the system uses one network analysis policy to preprocess all traffic handled by managed devices using a single access control policy. The following diagram shows how a newly created access control policy in an inline, intrusion-prevention deployment initially handles traffic. The preprocessing and intrusion prevention phases are highlighted.

Figure 2: New Access Control Policy: Intrusion Prevention

Notice how a default network analysis policy governs the preprocessing of *all* traffic handled by the access control policy. Initially, the system-provided Balanced Security and Connectivity network analysis policy is the default.

A simple way to tune preprocessing is to create and use a custom network analysis policy as the default. However, if you disable a preprocessor in a custom network analysis policy but the system needs to evaluate preprocessed packets against an enabled intrusion or preprocessor rule, the system automatically enables and uses the preprocessor although it remains disabled in the network analysis policy web interface.

Note In order to get the performance benefits of disabling a preprocessor, you **must** make sure that none of your intrusion policies have enabled rules that require that preprocessor.

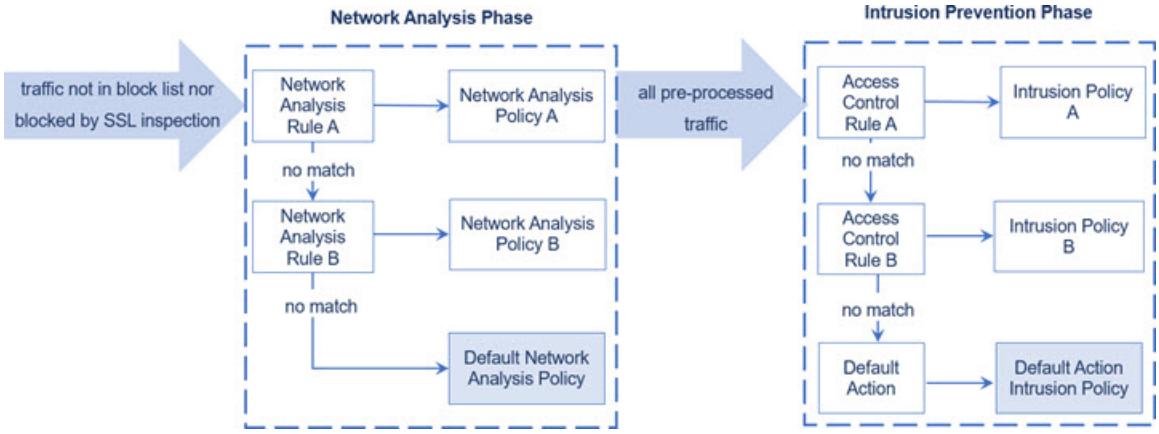
An additional challenge arises if you use multiple custom network analysis policies. For advanced users with complex deployments, you can tailor preprocessing to specific security zones, networks, and VLANs by assigning custom network analysis policies to preprocess matching traffic. To accomplish this, you add custom *network analysis rules* to your access control policy. Each rule has an associated network analysis policy that governs the preprocessing of traffic that matches the rule.

Tip You configure network analysis rules as an advanced setting in an access control policy. Unlike other types of rules in the system, network analysis rules invoke—rather than being contained by—network analysis policies.

The system matches packets to any configured network analysis rules in top-down order by rule number. Traffic that does not match any network analysis rule is preprocessed by the default network analysis policy. While this allows you a great deal of flexibility in preprocessing traffic, keep in mind that all packets, **regardless** of which network analysis policy preprocessed them, are subsequently matched to access control rules—and thus to potential inspection by intrusion policies—in their own process. In other words, preprocessing a packet with a particular network analysis policy does **not** guarantee that the packet will be examined with any

particular intrusion policy. You **must** carefully configure your access control policy so it invokes the correct network analysis and intrusion policies to evaluate a particular packet.

The following diagram shows in focused detail how the network analysis policy (preprocessing) selection phase occurs before and separately from the intrusion prevention (rules) phase. For simplicity, the diagram eliminates the discovery and file/malware inspection phases. It also highlights the default network analysis and default-action intrusion policies.



In this scenario, an access control policy is configured with two network analysis rules and a default network analysis policy:

- Network Analysis Rule A preprocessors matching traffic with Network Analysis Policy A. Later, you want this traffic to be inspected by Intrusion Policy A.
- Network Analysis Rule B preprocessors matching traffic with Network Analysis Policy B. Later, you want this traffic to be inspected by Intrusion Policy B.
- All remaining traffic is preprocessed with the default network analysis policy. Later, you want this traffic to be inspected by the intrusion policy associated with the access control policy's default action.

After the system preprocessors traffic, it can examine the traffic for intrusions. The diagram shows an access control policy with two access control rules and a default action:

- Access Control Rule A allows matching traffic. The traffic is then inspected by Intrusion Policy A.
- Access Control Rule B allows matching traffic. The traffic is then inspected by Intrusion Policy B.
- The access control policy's default action allows matching traffic. The traffic is then inspected by the default action's intrusion policy.

Each packet's handling is governed by a network analysis policy and intrusion policy pair, but the system does **not** coordinate the pair for you. Consider a scenario where you misconfigure your access control policy so that Network Analysis Rule A and Access Control Rule A do not process the same traffic. For example, you could intend the paired policies to govern the handling of traffic on a particular security zone, but you mistakenly use different zones in the two rules' conditions. This could cause traffic to be incorrectly preprocessed. For this reason, tailoring preprocessing using network analysis rules and custom policies is an **advanced** task.

Note that for a single connection, although the system selects a network analysis policy before an access control rule, some preprocessing (notably application layer preprocessing) occurs after access control rule selection. This does **not** affect how you configure preprocessing in custom network analysis policies.

License Requirements for Network Analysis and Intrusion Policies

Threat Defense License

IPS

Requirements and Prerequisites for Network Analysis and Intrusion Policies

Model support

Any.

Supported domains

Any

User roles

- Admin
- Intrusion Admin

Guidelines and Limitations for Network Analysis and Intrusion Policies

- A high percentage of traffic with small packets causes Snort performance to decrease. This behaviour is observed even when all the preprocessors are disabled.
- When you attempt to deploy a configuration change on a threat defense device with low memory, snort deployment is also triggered. This results in high consumption of RSS memory. Snort memory usage is also impacted if you deploy large configurations on the device, such as multiple IPS policies containing a large number of snort IPS rules, network objects, and access-control lists. You can mitigate such memory issues by optimizing the configuration. For best practices on how to configure access control rules to optimize the configuration, see Best Practices for Access Control Rules.
- If you increase the memory of a Threat Defense Virtual instance, you must redeploy the configuration for Snort 3 to utilize the additional memory.

Note The Snort 3 memory allocation is not automatically adjusted when you increase the memory of the Threat Defense Virtual instance. You must redeploy the configuration to regenerate relevant configuration files, such as `memory_allocation.lua`, which apply the updated resource limits to Snort 3.

- If an SIP stream is followed by RTP streams from the same connection, Snort inspects the initial SIP communication that is sent for connection establishment and allows SIP traffic. The RTP streams that follow the SIP communication are also trusted by default and bypass the configured rules. To prevent such scenarios, trusting the parent SIP connection or adding the parent SIP connection to a prefilter rule ensures that only the SIP stream bypasses Snort inspection and allows the subsequent RTP streams to be evaluated separately against the corresponding rules.
- If several Snort rules with the same action match a packet in the Management Center, an event is logged for each rule. However, if the matching rules have different actions, only the rule with the highest priority action generates an event. The action priority from highest to lowest is: pass, reset, block, drop, alert, log.
- When intrusion packet events are forwarded via syslog or the eStreamer fully-qualified event feed, the packet data field may be truncated due to limitations in the buffer size available for syslog generation and the eStreamer. In such scenarios, the packet length will not match the actual packet data that is sent.
- You cannot make policy changes, switch Snort versions, and deploy both these changes at the same time. You must make the required policy changes and deploy, or switch the snort versions and deploy.

The Navigation Panel: Network Analysis and Intrusion Policies

Network analysis and intrusion policies use similar web interfaces to edit and save changes to their configurations.

A navigation panel appears on the left side of the web interface when you are editing either type of policy. The following graphic shows the navigation panel for the network analysis policy (left) and the intrusion policy (right).

A dividing line separates the navigation panel into links to policy settings you can configure with (below) or without (above) direct interaction with policy layers. To navigate to any settings page, click its name in the navigation panel. Dark shading of an item in the navigation panel highlights your current settings page. For example, in the illustration above the Policy Information page would be displayed to the right of the navigation panel.

Policy Information

The Policy Information page provides configuration options for commonly used settings. As shown in the illustration for the network analysis policy panel above, a **Policy Change icon** appears next to **Policy Information** in the navigation panel when the policy contains unsaved changes. The icon disappears when you save your changes.

Rules (intrusion policy only)

The Rules page in an intrusion policy allows you to configure rule states and other settings for shared object rules, standard text rules, and preprocessor rules.

Cisco Recommendations (intrusion policy only)

The Cisco Recommendations page in an intrusion policy allows you to associate the operating systems, servers, and client application protocols detected on your network with intrusion rules specifically written to protect those assets. This allows you to tailor your intrusion policy to the specific needs of your monitored network.

Settings (network analysis policy) and Advanced Settings (intrusion policy)

The Settings page in a network analysis policy allows you to enable or disable preprocessors and access preprocessor configuration pages. Expanding the **Settings** link displays sublinks to individual configuration pages for all enabled preprocessors in the policy.

The Advanced Settings page in an intrusion policy allows you to enable or disable advanced settings and access configuration pages for those advanced settings. Expanding the **Advanced Settings** link displays sublinks to individual configuration pages for all enabled advanced settings in the policy.

Policy Layers

The Policy Layers page displays a summary of the layers that comprise your network analysis or intrusion policy. Expanding the Policy Layers link displays sublinks to summary pages for the layers in your policy. Expanding each layer sublink displays further sublinks to the configuration pages for all rules, preprocessors, or advanced settings that are enabled in the layer.

Conflicts and Changes: Network Analysis and Intrusion Policies

When you edit a network analysis or intrusion policy, a **Policy Change icon** appears next to **Policy Information** in the navigation panel to indicate that the policy contains unsaved changes. You must save (or *commit*) your changes before the system recognizes them.

Note After you save, you must deploy the network analysis or intrusion policy for your changes to take effect. If you deploy a policy without saving, the system uses the most recently saved configuration.

Resolving Editing Conflicts

The Network Analysis Policy page (**Policies** > **Access Control heading** > **Access Control**, and then click **Network Analysis Policy** or **Policies** > **Access Control heading** > **Intrusion**, and then click **Network Analysis Policies**) and Intrusion Policy page (**Policies** > **Access Control heading** > **Intrusion**) display whether each policy has unsaved changes, as well as information about who is currently editing the policy. Cisco recommends that only one person edit a policy at a time. If you are performing simultaneous editing, the consequences are as follows:

- If you are editing a network analysis or intrusion policy at the same time another user is editing the same policy, and the other user saves their changes to the policy, you are warned when you commit the policy that you will overwrite the other user's changes.
- If you are editing the same network analysis or intrusion policy via multiple web interface instances as the same user, and you save your changes for one instance, you cannot save your changes for the other instance.

Resolving Configuration Dependencies

To perform their particular analysis, many preprocessors and intrusion rules require that traffic first be decoded or preprocessed in a certain way, or have other dependencies. When you save a network analysis or intrusion policy, the system either automatically enables required settings, or warns you that disabled settings will have no effect on traffic, as follows:

- You cannot save an intrusion policy if you added an SNMP rule alert but did not configure SNMP alerting. You must either configure SNMP alerting or disable the rule alert, then save again.
- You cannot save an intrusion policy if it includes enabled sensitive data rules but you have not enabled the sensitive data preprocessor. You must either allow the system to enable the preprocessor and save the policy, or disable the rules and save again.
- If you disable a required preprocessor in a network analysis policy, you can still save the policy. However, the system automatically uses the disabled preprocessor with its current settings, even though the preprocessor remains disabled in the web interface.
- If you disable inline mode in a network analysis policy but enable the Inline Normalization preprocessor, you can still save the policy. However, the system warns you that normalization settings will be ignored. Disabling inline mode also causes the system to ignore other settings that allow preprocessors to modify or block traffic, including checksum verification and rate-based attack prevention.

Committing, Discarding, and Caching Policy Changes

While editing a network analysis or intrusion policy, if you exit the policy editor without saving your changes, the system caches those changes. Your changes are cached even when you log out of the system or experience a system crash. The system cache can store unsaved changes for one network analysis and one intrusion policy per user; you must commit or discard your changes before editing another policy of the same type. The system discards the cached changes when you edit another policy without saving your changes to the first policy, or when you import an intrusion rule update.

You can commit or discard policy changes on the Policy Information page of either the network analysis or intrusion policy editor.

In the Secure Firewall Management Center configuration, you can control:

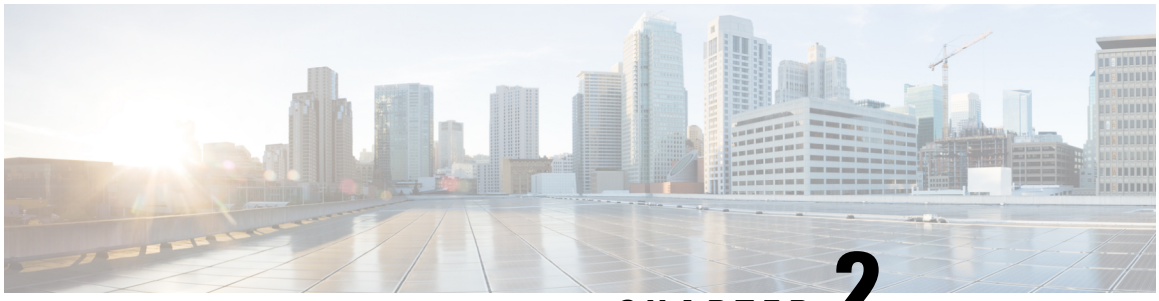
- whether you are prompted (or required) to comment on your network analysis or intrusion policy changes when you commit them
- whether changes and comments are recorded in the audit log

Exiting a Network Analysis or Intrusion Policy

Procedure

If you want to exit the network analysis or intrusion policy advanced editor, you have the following choices:

- Cache — To exit the policy and cache changes, choose any menu or other path to another page. On exiting, click **Leave page** when prompted, or click **Stay on page** to remain in the advanced editor.
- Discard — To discard unsaved changes, click **Discard Changes** on the Policy Information page, then click **OK**.
- Save — To save changes to the policy, click **Commit Changes** on the Policy Information page. If prompted, enter a comment, and then click **OK**.



CHAPTER 2

Getting Started with Intrusion Policies

The following topics explain how to get started with intrusion policies:

Intrusion Policy Basics

Intrusion policies are defined sets of intrusion detection and prevention configurations that inspect traffic for security violations and, in inline deployments, can block or alter malicious traffic. Intrusion policies are invoked by your access control policy and are the system's last line of defense before traffic is allowed to its destination.

At the heart of each intrusion policy are the intrusion rules. An enabled rule causes the system to generate intrusion events for (and optionally block) traffic matching the rule. Disabling a rule stops processing of the rule.

The system delivers several base intrusion policies, which enable you to take advantage of the experience of the Talos Intelligence Group. For these policies, Talos sets intrusion and preprocessor rule states (enabled or disabled), as well as provides the initial configurations for other advanced settings.

Tip System-provided intrusion and network analysis policies are similarly named but contain different configurations. For example, the Balanced Security and Connectivity network analysis policy and the Balanced Security and Connectivity intrusion policy work together and can both be updated in intrusion rule updates. However, the network analysis policy governs mostly preprocessing options, whereas the intrusion policy governs mostly intrusion rules.

If you create a custom intrusion policy, you can:

- Tune detection by enabling and disabling rules, as well as by writing and adding your own rules.
- Use Cisco recommendations to associate the operating systems, servers, and client application protocols detected on your network with rules specifically written to protect those assets.
- Configure various advanced settings such as external alerting, sensitive data preprocessing, and global rule thresholding.
- Use layers as building blocks to efficiently manage multiple intrusion policies.

In an inline deployment, an intrusion policy can block and modify traffic:

- *Drop rules* can drop matching packets and generate intrusion events. To configure an intrusion or preprocessor drop rule, set its state to Drop and Generate Events.
- Intrusion rules can use the `replace` keyword to replace malicious content.

For intrusion rules to affect traffic, you must correctly configure drop rules and rules that replace content, as well as correctly deploy managed devices inline, that is, with inline interface sets. Finally, you must enable the intrusion policy's *drop behavior*, or **Drop when Inline** setting.

When tailoring your intrusion policy, especially when enabling and adding rules, keep in mind that some intrusion rules require that traffic first be decoded or preprocessed in a certain way. Before an intrusion policy examines a packet, the packet is preprocessed according to configurations in a network analysis policy. If you disable a required preprocessor, the system automatically uses it with its current settings, although the preprocessor remains disabled in the network analysis policy web interface.

Caution Because preprocessing and intrusion inspection are so closely related, the network analysis and intrusion policies examining a single packet **must** complement each other. Tailoring preprocessing, especially using multiple custom network analysis policies, is an **advanced** task.

After you configure a custom intrusion policy, you can use it as part of your access control configuration by associating the intrusion policy with one or more access control rules or an access control policy's default action. This forces the system to use the intrusion policy to examine certain allowed traffic before the traffic passes to its final destination. A variable set that you pair with the intrusion policy allows you to accurately reflect your home and external networks and, as appropriate, the servers on your network.

Note that by default, the system disables intrusion inspection of encrypted payloads. This helps reduce false positives and improve performance when an encrypted connection matches an access control rule that has intrusion inspection configured.

License Requirements for Intrusion Policies

Threat Defense License

IPS

Requirements and Prerequisites for Intrusion Policies

Model support

Any.

Supported domains

Any

User roles

- Admin
- Intrusion Admin

Managing Intrusion Policies

On the Intrusion Policy page (**Policies** > **Access Control heading** > **Intrusion**) you can view your current custom intrusion policies, along with the following information:

- the time and date the policy was last modified (in local time) and the user who modified it
- whether the **Drop when Inline** setting is enabled, which allows you to drop and modify traffic in an inline deployment. An inline deployment could be configurations that are deployed to devices using routed, switched, or transparent interfaces, or inline interface pairs.
- which access control policies and devices are using the intrusion policy to inspect traffic
- whether a policy has unsaved changes, as well as information about who (if anyone) is currently editing the policy

Procedure

Step 1 Choose **Policies** > **Access Control heading** > **Intrusion**.

Step 2 Manage your intrusion policy:

- Compare—Click **Compare Policies**; see Comparing policies.
- Create — Click **Create Policy**; see:

- Creating a Custom Snort 2 Intrusion Policy, on page 22 for Snort 2 policies.
- *Creating a Custom Snort 3 Intrusion Policy* topic in the latest version of the *Custom Snort 3 Intrusion Policies for Access Control* for Snort 3 policies.

- Delete — Click **Delete** ( ) next to the policy you want to delete. The system prompts you to confirm and informs you if another user has unsaved changes in the policy. Click **OK** to confirm.

  If the controls are dimmed, the configuration belongs to an ancestor domain, or you do not have permission to modify the configuration.

- Edit — Choose:
  - **Snort 2 Version**; see Editing Snort 2 Intrusion Policies, on page 23.
  - **Snort 3 Version**; see *Editing Snort 3 Intrusion Policies* topic in the latest version of the *Custom Snort 3 Intrusion Policies for Access Control*.

  If **View** ( ) appears instead, the configuration belongs to an ancestor domain, or you do not have permission to modify the configuration.

- Export — If you want to export an intrusion policy to import on another Secure Firewall Management Center, click **YouTube EDU** ( ); see .
- Deploy—Choose **Deploy** > **Deploy**; see Deploy Configuration Changes.
- Report—Click **Report**( ) ; see Generate Current Policy Reports.

Custom Intrusion Policy Creation

When you create a new intrusion policy you must give it a unique name, specify a base policy, and specify drop behavior.

The base policy defines the intrusion policy's default settings. Modifying a setting in the new policy overrides—but does not change—the settings in the base policy. You can use either a system-provided or custom policy as your base policy.

Creating a Custom Snort 2 Intrusion Policy

Procedure

Step 1 Choose **Policies** > **Access Control heading** > **Intrusion**.

Step 2 Click **Create Policy**. If you have unsaved changes in another policy, click **Cancel** when prompted to return to the Intrusion Policy page.

Ensure the **Intrusion Policies** tab is selected.

Step 3 Enter a unique **Name** and, optionally, a **Description**.

Step 4 Choose the **Inspection Mode**.

The selected action determines whether intrusion rules block and alert (**Prevention** mode) or only alert (**Detection** mode).

Step 5 Choose the initial **Base Policy**.

You can use either a system-provided or another custom policy as your base policy.

Step 6 Click **Save**.

The new policy has the same settings as its base policy.

Editing Snort 2 Intrusion Policies

Procedure

Step 1 Choose **Policies** > **Access Control heading** > **Intrusion**.

Step 2 Ensure the **Intrusion Policies** tab is selected.

Step 3 Click **Snort 2 Version** next to the intrusion policy you want to configure.

Step 4 Edit your policy:

- Change the base policy—Choose a base policy from the **Base Policy** drop-down list; see Changing the Base Policy, on page 201.
- Configure advanced settings—Click **Advanced Settings** in the navigation panel; see Intrusion Policy Advanced Settings, on page 27.
- Configure Cisco recommended intrusion rules—Click **Cisco Recommendations** in the navigation panel; see Generating and Applying Cisco Recommendations, on page 55.
- Drop behavior in an inline deployment—Check or clear **Drop when Inline**; see Setting Drop Behavior in an Inline Deployment, on page 26.
- Filter rules by recommended rule state—After you generate recommendations, click **View** next to each recommendation type. Click **View Recommended Changes** to view all recommendations.
- Filter rules by current rule state—Click **View** next to each rule state type (generate events, drop and generate events); see Intrusion Rule Filters in an Intrusion Policy, on page 35.
- Manage policy layers—Click **Policy Layers** in the navigation panel; see Layer Management, on page 202.
- Manage intrusion rules—Click **Manage Rules**; see Viewing Intrusion Rules in an Intrusion Policy, on page 30.
- View settings in base policy—Click **Manage Base Policy**; see The Base Layer, on page 199.

Step 5 To save changes you made in this policy since the last policy commit, choose **Policy Information**, then click **Commit Changes**.

If you leave the policy without committing changes, changes since the last commit are discarded if you edit a different policy.

Intrusion Policy Changes

When you create a new intrusion policy, it has the same intrusion rule and advanced settings as its base policy.

The system caches one intrusion policy per user. While editing an intrusion policy, if you choose any menu or other path to another page, your changes stay in the system cache even if you leave the page.

Access Control Rule Configuration to Perform Intrusion Prevention

An access control policy can have multiple access control rules associated with intrusion policies. You can configure intrusion inspection for any Allow or Interactive Block access control rule, which permits you to match different intrusion inspection profiles against different types of traffic on your network before it reaches its final destination.

Whenever the system uses an intrusion policy to evaluate traffic, it uses an associated *variable set*. Variables in a set represent values commonly used in intrusion rules to identify source and destination IP addresses and ports. You can also use variables in intrusion policies to represent IP addresses in rule suppressions and dynamic rule states.

Tip Even if you use system-provided intrusion policies, Cisco **strongly** recommends you configure the system's intrusion variables to accurately reflect your network environment. At a minimum, modify default variables in the default set.

Understanding System-Provided and Custom Intrusion Policies

Cisco delivers several intrusion policies with the system. By using system-provided intrusion policies, you can take advantage of the experience of the Talos Intelligence Group. For these policies, Talos sets intrusion and preprocessor rule states, as well as provides the initial configurations for advanced settings. You can use system-provided policies as-is, or you can use them as the base for custom policies. Building custom policies can improve the performance of the system in your environment and provide a focused view of the malicious traffic and policy violations occurring on your network.

Connection and Intrusion Event Logging

When an intrusion policy invoked by an access control rule detects an intrusion and generates an intrusion event, it saves that event to the Secure Firewall Management Center. The system also automatically logs the end of the connection where the intrusion occurred to the Secure Firewall Management Center database, regardless of the logging configuration of the access control rule.

Access Control Rule Configuration and Intrusion Policies

The number of unique intrusion policies you can use in a single access control policy depends on the model of the target devices; more powerful devices can handle more. Every unique **pair** of intrusion policy and variable set counts as one policy. Although you can associate a different intrusion policy-variable set pair with each Allow and Interactive Block rule (as well as with the default action), you cannot deploy an access control policy if the target devices have insufficient resources to perform inspection as configured.

Configuring an Access Control Rule to Perform Intrusion Prevention

You must be an Admin, Access Admin, or Network Admin to perform this task.

Caution Changing the total number of intrusion policies used by an access control policy restarts the Snort process when you deploy configuration changes, temporarily interrupting traffic inspection. Whether traffic drops during this interruption or passes without further inspection depends on how the assigned device handles traffic. You change the total number of intrusion policies by adding an intrusion policy that is not currently used, or by removing the last instance of an intrusion policy. You can use an intrusion policy in an access control rule, as the default action, or as the default intrusion policy.

Procedure

Step 1 In the access control policy editor, create a new rule or edit an existing rule.

Step 2 Ensure the rule action is set to **Allow**, **Interactive Block**, or **Interactive Block with reset**.

Step 3 Click **Inspection**.

Step 4 Choose a system-provided or custom **Intrusion Policy**, or choose **None** to disable intrusion inspection for traffic that matches the access control rule.

Step 5 If you want to change the variable set associated with the intrusion policy, choose a value from the **Variable Set** drop-down list.

Step 6 Click **Save** to save the rule.

Step 7 Click **Save** to save the policy.

Drop Behavior in an Inline Deployment

If you want to assess how your configuration would function in an inline deployment (that is, where relevant configurations are deployed to devices using routed, switched, or transparent interfaces, or inline interface pairs) without actually affecting traffic, you can disable drop behavior. In this case, the system generates intrusion events but does not drop packets that trigger the drop rules. When you are satisfied with the results, you can enable drop behavior.

Note that in passive or inline deployments in tap mode, the system cannot affect traffic regardless of the drop behavior. In a passive deployment, rules set to **Drop and Generate Events** behave identically to rules set to **Generate Events**. The system generates intrusion events but cannot drop packets.

Note Suppose a file Block action causes a Block or Pending file policy verdict on a packet, and later, an IPS event is generated on the same packet. In that case, the IPS event is marked as Dropped instead of Would have dropped even if the IPS policy is in detection mode (IDS).

Note To block the transfer of malware over FTP, you must not only correctly configure malware defense, but also enable **Drop when Inline** in your access control policy's default intrusion policy.

When you view intrusion events, workflows can include the *inline result*, which indicates whether traffic was actually dropped, or whether it only would have dropped.

Setting Drop Behavior in an Inline Deployment

Procedure

Step 1 Choose **Policies** > **Access Control heading** > **Intrusion**.

Step 2 Click **Snort 2 Version** next to the policy you want to edit.

If **View** ( ) appears instead, the configuration belongs to an ancestor domain, or you do not have permission to modify the configuration.

Step 3 Set the policy's drop behavior:

- Check the **Drop when Inline** check box to allow intrusion rules to affect traffic and generate events.
- Clear the **Drop when Inline** check box to prevent intrusion rules from affecting traffic while still generating events.

Step 4 Click **Commit Changes** to save changes you made in this policy since the last policy commit.

If you leave the policy without committing changes, changes since the last commit are discarded if you edit a different policy.

What to do next

- Deploy configuration changes.

Drop Behavior in a Dual System Deployment

When there are two systems connected back to back in a network, it is normal to see the first system drop events and still record a drop or "would have dropped" event on the second system. The first system decides to drop the packets by the time it scans the last packet of the file, while the second system also investigates and identifies the traffic as "to be dropped".

For example, a 5 packet HTTP GET request whose first packet triggers a rule is blocked by the first system and only the last packet is dropped. The second system receives only 4 packets and the connection gets dropped, but when the second system finally flushes the partial GET request while it is pruning the session, it triggers the same rule with "would have dropped" as the inline result.

Intrusion Policy Advanced Settings

An intrusion policy's *advanced settings* require specific expertise to configure. The base policy for your intrusion policy determines which advanced settings are enabled by default and the default configuration for each.

When you choose **Advanced Settings** in the navigation panel of an intrusion policy, the policy lists its advanced settings by type. On the Advanced Settings page, you can enable or disable advanced settings in your intrusion policy, as well as access advanced setting configuration pages. An advanced setting must be enabled for you to configure it.

When you disable an advanced setting, the sublink and **Edit** link no longer appear, but your configurations are retained. Note that some intrusion policy configurations (sensitive data rules, SNMP alerts for intrusion rules) require enabled and correctly configured advanced settings.

Modifying the configuration of an advanced setting requires an understanding of the configuration you are modifying and its potential impact on your network.

Specific Threat Detection

The sensitive data preprocessor detects sensitive data such as credit card numbers and Social Security numbers in ASCII text.

Note that other preprocessors that detect specific threats (back orifice attacks, several portscan types, and rate-based attacks that attempt to overwhelm your network with excessive traffic) are configured in network analysis policies.

Intrusion Rule Thresholds

Global rule thresholding can prevent your system from being overwhelmed with a large number of events by allowing you to use thresholds to limit the number of times the system logs and displays intrusion events.

External Responses

In addition to the various views of intrusion events in the web interface, you can enable logging to system log (syslog) facilities or send event data to an SNMP trap server. Per policy, you can specify intrusion event notification limits, set up intrusion event notification to external logging facilities, and configure external responses to intrusion events.

Note that in addition to these per-policy alerting configurations, you can globally enable or disable email alerting on intrusion events for each rule or rule group. Your email alert settings are used regardless of which intrusion policy processes a packet.

Optimizing Performance for Intrusion Detection and Prevention

If you want the system to perform intrusion detection and prevention but do not need to take advantage of discovery data, you can optimize performance by disabling new discovery as described below.

Before you begin

To perform this task, you must have one of the following user roles:

- Admin, Access Admin, or Network Admin for access control.
- Admin or Discovery Admin for network discovery.

Procedure

Step 1 Modify or delete rules associated with the access control policy deployed at the target device. None of the access control rules associated with that device can have user, application, or URL conditions; see Create and edit access control rules.

Step 2 Delete all rules from the network discovery policy for the target device; see Configuring Network Discovery Rules.

Step 3 Deploy the changed configuration to the target device; see Deploy Configuration Changes.

Tuning Intrusion Policies Using Rules

The following topics explain how to use rules to tune intrusion policies:

Related Topics

Setting Drop Behavior in an Inline Deployment, on page 26

Intrusion Rule Tuning Basics

You can use the Rules page in an intrusion policy to configure rule states and other settings for shared object rules, standard text rules, and preprocessor rules.

You enable a rule by setting its rule state to Generate Events or to Drop and Generate Events. Enabling a rule causes the system to generate events on traffic matching the rule. Disabling a rule stops processing of the rule. You can also set your intrusion policy so that a rule set to Drop and Generate Events in an inline deployment generates events on, and drops, matching traffic. In a passive deployment, a rule set to Drop and Generate Events just generates events on matching traffic.

You can filter rules to display a subset of rules, enabling you to select the exact set of rules where you want to change rule states or rule settings.

When an intrusion rule or rule argument requires a disabled preprocessor, the system automatically uses it with its current configuration even though it remains disabled in the network analysis policy's web interface.

Intrusion Rule Types

An intrusion rule is a specified set of keywords and arguments that the system uses to detect attempts to exploit vulnerabilities in your network. As the system analyzes network traffic, it compares packets against the conditions specified in each rule, and triggers the rule if the data packet meets all the conditions specified in the rule.

An intrusion policy contains:

- *intrusion rules*, which are subdivided into *shared object rules* and *standard text rules*
- *preprocessor rules*, which are associated with a detection option of the packet decoder or with one of the preprocessors included with the system

The following table summarizes attributes of these rule types:

Table 1: Intrusion Rule Types

Type	Generator ID (GID)	Snort ID (SID)	Source	Can Copy?	Can Edit?
shared object rule	3	lower than 1000000	Talos Intelligence Group	yes	limited
standard text rule	1 (Global domain or legacy GID)	lower than 1000000	Talos	yes	limited
	1000 - 2000 (descendant domain)	1000000 or higher	Created or imported by user	yes	yes
preprocessor rule	decoder- or preprocessor-specific	lower than 1000000	Talos	no	no
		1000000 or higher	Generated by the system during option configuration	no	no

You cannot save changes to any rule created by Talos, but you can save a copy of a modified rule as a custom rule. You can modify either variables used in the rule or rule header information (such as source and destination ports and IP addresses).

For the rules it creates, Talos assigns default rule states in each default intrusion policy. Most preprocessor rules are disabled by default and must be enabled if you want the system to generate events for preprocessor rules and, in an inline deployment, drop offending packets.

In a multidomain deployment, the system prepends a domain number to the SID of any custom rule created in or imported into a descendant domain. For example, a rule added in the Global domain would have a SID of 1000000 or greater, and rules added in descendant domains would have SIDs of [domain number]000000 or greater.

License Requirements for Intrusion Rules

Threat Defense License

IPS

Requirements and Prerequisites for Intrusion Rules

Model support

Any.

Supported domains

Any

User roles

- Admin
- Intrusion Admin

Viewing Intrusion Rules in an Intrusion Policy

You can adjust how rules are displayed in the intrusion policy, and can sort rules by several criteria. You can also display the details for a specific rule to see rule settings, rule documentation, and other rule specifics.

Procedure

Step 1 Choose **Policies** > **Access Control heading** > **Intrusion**.

Step 2 Click **Snort 2 Version** next to the policy you want to edit.

If **View** ( ) appears instead, the configuration belongs to an ancestor domain, or you do not have permission to modify the configuration.

Step 3 Click **Rules** under **Policy Information** in the navigation panel.

Step 4 While viewing the rules, you can:

- Filter the rules as described in Setting a Rule Filter in an Intrusion Policy, on page 41.
- Sort the rules by clicking the title in the top of the column you want to sort by.
- View an intrusion rule's details as described in Viewing Intrusion Rule Details, on page 32.
- View rules in different policy layers by choosing a layer from the **Policy** drop-down list.

Intrusion Rules Page Columns

The Intrusion Rules page uses the same icons in its menu bar and column headers. For example, the Rule State menu uses the same **Generate Events** as the Rule State column in the rule listing.

Table 2: Rules Page Columns

Heading	Description
GID	Integer that indicates the Generator ID (GID) for the rule.
SID	Integer that indicates the Snort ID (SID), which acts a unique identifier for the rule. For custom rules, the SID is 1000000 or higher.
Message	Message included in events generated by this rule, which also acts as the name of the rule.

Heading	Description
Generate Events	The rule state for the rule: • **Drop and Generate Events** • **Generate Events** • **Disabled** Note the icon for a disabled rule is a dimmed version of the icon for a rule that is set to generate events without dropping traffic. Also, clicking the rule state icon for a rule allows you to change the rule state.
Cisco Recommended rule state	Cisco recommended rule state for the rule.
Event Filter	Event filter, including event thresholds and event suppression, applied to the rule.
Dynamic state	Dynamic rule state for the rule, which goes into effect if specified rate anomalies occur.
Errors ( )	Alerts configured for the rule (currently SNMP alerts only).
Comment ( )	Comments added to the rule.

You can also use the layer drop-down list to switch to the Rules page for other layers in your policy. Note that, unless you add layers to your policy, the only editable views listed in the drop-down list are the policy Rules page and the Rules page for a policy layer that is originally named `My Changes`; note also that making changes in one of these views is the same as making the changes in the other. The drop-down list also lists the Rules page for the read-only base policy.

Intrusion Rule Details

You can view rule documentation, Cisco recommendations, and rule overhead from the Rule Detail view. You can also view and add rule-specific features.

Table 3: Rule Details

Item	Description
Summary	The rule summary. For rule-based events, this row appears when the rule documentation contains summary information.
Rule State	The current rule state for the rule. Also indicates the layer where the rule state is set.
Cisco Recommendation	If Cisco recommendations have been generated, an icon that represents the recommended rule state; see Intrusion Rules Page Columns, on page 30. If the recommendation is to enable the rule, the system also indicates the network assets or configurations that triggered the recommendation.
Rule Overhead	The rule's potential impact on system performance and the likelihood that the rule might generate false positives. Local rules do not have an assigned overhead, unless they are mapped to a vulnerability.
Thresholds	Thresholds currently set for this rule, as well as the facility to add a threshold for the rule.

Item	Description
Suppressions	Suppression settings currently set for this rule, as well as the facility to add suppressions for the rule.
Dynamic State	Rate-based rule states currently set for this rule, as well as the facility to add dynamic rule states for the rule.
Alerts	SNMP alerts set for this rule, as well as the facility to add an alert for the rule.
Comments	Comments added to this rule, as well as the facility to add comments for the rule.
Documentation	The rule documentation for the current rule, supplied by the Talos Intelligence Group. Optionally, click **Rule Documentation** to view more-specific rule details.

Viewing Intrusion Rule Details

Procedure

Step 1 Choose **Policies** > **Access Control heading** > **Intrusion**.

Step 2 Click **Snort 2 Version** next to the policy you want to edit.

If **View** () appears instead, the configuration belongs to an ancestor domain, or you do not have permission to modify the configuration.

Step 3 On the navigation pane, click **Rules**.

Step 4 Click the rule whose rule details you want to view, then click **Show details** at the bottom of the page.
Rule details appear, as described in Intrusion Rule Details, on page 31.

Step 5 From the rule details, you can configure:

- Alerts—See Setting an SNMP Alert for an Intrusion Rule, on page 34.
- Comments—See Adding a Comment to an Intrusion Rule, on page 35.
- Dynamic rule states—See Setting a Dynamic Rule State from the Rule Details Page, on page 34.
- Thresholds—See Setting a Threshold for an Intrusion Rule, on page 32.
- Suppressions—See Setting Suppression for an Intrusion Rule, on page 33.

Setting a Threshold for an Intrusion Rule

You can set a single threshold for a rule from the Rule Detail page. Adding a threshold overwrites any existing threshold for the rule.

Note that a **Revert** appears in a field when you enter an invalid value; click it to revert to the last valid value for that field or to clear the field if there was no previous value.

Procedure

Step 1 From an intrusion rule's details, click **Add** next to **Thresholds**.

Step 2 From the **Type** drop-down list, choose the type of threshold you want to set:

- Choose **Limit** to limit notification to the specified number of event instances per time period.
- Choose **Threshold** to provide notification for each specified number of event instances per time period.
- Choose **Both** to provide notification once per time period after a specified number of event instances.

Step 3 From the **Track By** drop-down list, choose **Source** or **Destination** to indicate whether you want the event instances tracked by source or destination IP address.

Step 4 In the **Count** field, enter the number of event instances you want to use as your threshold.

Step 5 In the **Seconds** field, enter a number that specifies the time period, in seconds, for which event instances are tracked.

Step 6 Click **OK**.

Tip
The system displays an **Event Filter** next to the rule in the Event Filtering column. If you add multiple event filters to a rule, the system includes an indication of the number of event filters.

Setting Suppression for an Intrusion Rule

You can set one or more suppressions for a rule in your intrusion policy.

Note that a **Revert** appears in a field when you type an invalid value; click it to revert to the last valid value for that field or to clear the field if there was no previous value.

Procedure

Step 1 From an intrusion rule's details, click **Add** next to **Suppressions**.

Step 2 From the **Suppression Type** drop-down list, choose one of the following options:

- Choose **Rule** to completely suppress events for a selected rule.
- Choose **Source** to suppress events generated by packets originating from a specified source IP address.
- Choose **Destination** to suppress events generated by packets going to a specified destination IP address.

Step 3 If you chose **Source** or **Destination** for the suppression type, in the **Network** field enter the IP address, an address block, or a comma-separated list comprised of any combination of these.

If the intrusion policy is associated with the default action of an access control policy, you can also specify or list a network variable in the default action variable set.

Step 4 Click **OK**.

Tip

The system displays an **Event Filter** next to the rule in the Event Filtering column next the suppressed rule. If you add multiple event filters to a rule, a number over the filter indicates the number of filters.

Setting a Dynamic Rule State from the Rule Details Page

You can set one or more dynamic rule states for a rule. The first dynamic rule state listed has the highest priority. When two dynamic rule states conflict, the action of the first is carried out.

Dynamic rule states are policy-specific.

Note that a **Revert** appears in a field when you enter an invalid value; click it to revert to the last valid value for that field or to clear the field if there was no previous value.

Procedure

Step 1 From an intrusion rule's details, click **Add** next to **Dynamic State**.

Step 2 From the **Track By** drop-down list, choose an option to indicate how you want the rule matches tracked:

- Choose **Source** to track the number of hits for that rule from a specific source or set of sources.
- Choose **Destination** to track the number of hits for that rule to a specific destination or set of destinations.
- Choose **Rule** to track all matches for that rule.

Step 3 If you set **Track By** to **Source** or **Destination**, enter the IP address of each host you want to track in the **Network** field.

Step 4 Next to **Rate**, specify the number of rule matches per time period to set the attack rate:

- In the **Count** field, specify the number of rule matches you want to use as your threshold.
- In the **Seconds** field, specify the number of seconds that make up the time period for which attacks are tracked.

Step 5 From the **New State** drop-down list, choose the new action to be taken when the conditions are met.

Step 6 Enter a value in the **Timeout** field.

After the timeout occurs, the rule reverts to its original state. Enter `0` to prevent the new action from timing out.

Step 7 Click **OK**.

Tip

The system displays a dynamic state () next to the rule in the Dynamic State column. If you add multiple dynamic rule state filters to a rule, a number over the filters indicates the number of filters.

Setting an SNMP Alert for an Intrusion Rule

You can set an SNMP alert for a rule from the Rule Detail page.

Procedure

From an intrusion rule's details, click **Add SNMP Alert** next to **Alerts**.

Tip

The system displays an alert **Errors** ( ) next to the rule in the Alerting column. If you add multiple alerts to a rule, the system includes an indication of the number of alerts.

Adding a Comment to an Intrusion Rule

Procedure

Step 1 From an intrusion rule's details, click **Add** next to **Comments**.

Step 2 In the **Comment** field, enter the rule comment.

Step 3 Click **OK**.

Tip

The system displays a **Comment** ( ) next to the rule in the Comments column. If you add multiple comments to a rule, a number over the comment indicates the number of comments.

Step 4 To delete a rule comment, click **Delete** in the rule comments section. You can only delete a comment if the comment is cached with uncommitted intrusion policy changes.

What to do next

- Deploy configuration changes.

Intrusion Rule Filters in an Intrusion Policy

You can filter the rules you display on the Rules page by a single criteria, or a combination of one or more criteria.

Rule filter keywords help you find the rules for which you want to apply rule settings, such as rule states or event filters. You can filter by a keyword and simultaneously select the argument for the keyword by selecting the argument you want from the Rules page filter panel.

Intrusion Rule Filters Notes

The filter you construct is shown in the Filter text box. You can click keywords and keyword arguments in the filter panel to construct a filter. When you choose multiple keywords, the system combines them using AND logic to create a compound search filter. For example, if you choose **preprocessor** under **Category** and then choose **Rule Content** > **GID** and enter `116`, you get a filter of `Category: "preprocessor" GID:"116"`, which retrieves all rules that are preprocessor rules **and** have a GID of 116.

The Category, Microsoft Vulnerabilities, Microsoft Worms, Platform Specific, Preprocessor, and Priority filter groups allow you to submit more than one argument for a keyword, separated by commas. For example, you can choose **os-linux** and **os-windows** from **Category** to produce the filter `Category:"os-windows,os-linux"`, which retrieves any rules in the `os-linux` category or in the `os-windows` category.

To show the filter panel, click the **Show icon**.

To hide the filter panel, click the **Hide icon**.

Intrusion Policy Rule Filters Construction Guidelines

In most cases, when you are building a filter, you can use the filter panel to the left of the Rules page in the intrusion policy to choose the keywords/arguments you want to use.

Rule filters are grouped into rule filter groups in the filter panel. Many rule filter groups contain sub-criteria so that you can more easily find the specific rules you are looking for. Some rule filters have multiple levels that you can expand to drill down to individual rules.

Items in the filter panel sometimes represent filter type groups, sometimes represent keywords, and sometimes represent the argument to a keyword. Note the following:

- When you choose a filter type group heading that is not a keyword (Rule Configuration, Rule Content, Platform Specific, and Priority), it expands to list the available keywords.

  When you choose a keyword by clicking on a node in the criteria list, a pop-up window appears, where you supply the argument you want to filter by.

  If that keyword is already used in the filter, the argument you supply replaces the existing argument for that keyword.

  For example, if you click **Drop and Generate Events** under **Rule Configuration** > **Recommendation** in the filter panel, `Recommendation:"Drop and Generate Events"` is added to the filter text box. If you then click **Generate Events** under Rule **Configuration** > **Recommendation**, the filter changes to `Recommendation:"Generate Events"`.

- When you choose a filter type group heading that is a keyword (Category, Classifications, Microsoft Vulnerabilities, Microsoft Worms, Priority, and Rule Update), it lists the available arguments.

  When you choose an item from this type of group, the argument and the keyword it applies to are immediately added to the filter. If the keyword is already in the filter, it replaces the existing argument for the keyword that corresponds to that group.

  For example, if you click **os-linux** under **Category** in the filter panel, `Category:"os-linux"` is added to the filter text box. If you then click **os-windows** under **Category**, the filter changes to `Category:"os-windows"`.

- Reference under Rule Content is a keyword, and so are the specific reference ID types listed below it. When you choose any of the reference keywords, a pop-up window appears, where you supply an argument and the keyword is added to the existing filter. If the keyword is already in use in the filter, the new argument you supply replaces the existing argument.

  For example, if you click **Rule Content** > **Reference** > **CVE ID** in the filter panel, a pop-up window prompts you to supply the CVE ID. If you enter `2007`, then `CVE:"2007"` is added to the filter text box. In another example, if you click **Rule Content** > **Reference** in the filter panel, a pop-up window prompts you to supply the reference. If you enter `2007`, then `Reference:"2007"` is added to the filter text box.

- When you choose rule filter keywords from different groups, each filter keyword is added to the filter and any existing keywords are maintained (unless overridden by a new value for the same keyword).

  For example, if you click **os-linux** under **Category** in the filter panel, `Category:"os-linux"` is added to the filter text box. If you then click **MS00-006** under **Microsoft Vulnerabilities**, the filter changes to `Category:"os-linux" MicrosoftVulnerabilities:"MS00-006"`.

- When you choose multiple keywords, the system combines them using AND logic to create a compound search filter. For example, if you choose **preprocessor** under **Category** and then choose **Rule Content** > **GID** and enter `116`, you get a filter of `Category: "preprocessor" GID:"116"`, which retrieves all rules that are preprocessor rules **and** have a GID of 116.
- The Category, Microsoft Vulnerabilities, Microsoft Worms, Platform Specific, and Priority filter groups allow you to submit more than one argument for a keyword, separated by commas. For example, you can choose **os-linux** and **os-windows** from **Category** to produce the filter `Category:"os-windows,app-detect"`, which retrieves any rules in the `os-linux` category or in the `os-windows` category.

The same rule may be retrieved by more than one filter keyword/argument pair. For example, the DOS Cisco attempt rule (SID 1545) appears if rules are filtered by the **dos** category, and also if you filter by the **High** priority.

Note The Talos Intelligence Group may use the rule update mechanism to add and remove rule filters.

Note that the rules on the Rules page may be either shared object rules (generator ID 3) or standard text rules (generator ID 1, Global domain or legacy GID; 1000 - 2000, descendant domains). The following table describes the different rule filters.

Table 4: Rule Filter Groups

Filter Group	Description	Multiple Argument Support?	Heading is...	Items in List are...
Rule Configuration	Finds rules according to the configuration of the rule.	No	A grouping	keywords
Rule Content	Finds rules according to the content of the rule.	No	A grouping	keywords
Category	Finds rules according to the rule categories used by the rule editor. Note that local rules appear in the local sub-group.	Yes	A keyword	arguments
Classifications	Finds rules according to the attack classification that appears in the packet display of an event generated by the rule.	No	A keyword	arguments
Microsoft Vulnerabilities	Finds rules according to Microsoft bulletin number.	Yes	A keyword	arguments
Microsoft Worms	Finds rules based on specific worms that affect Microsoft Windows hosts.	Yes	A keyword	arguments
Platform Specific	Finds rules according to their relevance to specific versions of operating systems. Note that a rule may affect more than one operating system or more than one version of an operating system. For example, enabling SID 2260 affects multiple versions of Mac OS X, IBM AIX, and other operating systems.	Yes	A keyword	arguments Note that if you pick one of the items from the sub-list, it adds a modifier to the argument.

Filter Group	Description	Multiple Argument Support?	Heading is...	Items in List are...
Preprocessors	Finds rules for individual preprocessors. Note that you must enable preprocessor rules associated with a preprocessor option to generate events and, in an inline deployment, drop offending packets for the option when the preprocessor is enabled.	Yes	A grouping	sub-groupings
Priority	Finds rules according to high, medium, and low priorities. The classification assigned to a rule determines its priority. These groups are further grouped into rule categories. Note that local rules (that is, rules that you import or create) do not appear in the priority groups.	Yes	A keyword	arguments Note that if you pick one of the items from the sub-list, it adds a modifier to the argument.
Rule Update	Finds rules added or modified through a specific rule update. For each rule update, view all rules in the update, only new rules imported in the update, or only existing rules changed by the update.	No	A keyword	arguments

Intrusion Rule Configuration Filters

You can filter the rules listed in the Rules page by several rule configuration settings. For example, if you want to view the set of rules whose rule state does not match the recommended rule state, you can filter on rule state by selecting **Does not match recommendation**.

When you choose a keyword by clicking on a node in the criteria list, you can supply the argument you want to filter by. If that keyword is already used in the filter, the argument you supply replaces the existing argument for that keyword.

For example, if you click **Drop and Generate Events** under **Rule Configuration** > **Recommendation** in the filter panel, `Recommendation:"Drop and Generate Events"` is added to the filter text box. If you then click **Generate Events** under **Rule Configuration** > **Recommendation**, the filter changes to `Recommendation:"Generate Events"`.

Intrusion Rule Content Filters

You can filter the rules listed in the Rules page by several rule content items. For example, you can quickly retrieve a rule by searching for the rule's SID. You can also find all rules that inspect traffic going to a specific destination port.

When you select a keyword by clicking on a node in the criteria list, you can supply the argument you want to filter by. If that keyword is already used in the filter, the argument you supply replaces the existing argument for that keyword.

For example, if you click **SID** under **Rule Content** in the filter panel, a pop-up window appears, prompting you to supply a SID. If you type `1045`, then `SID:"1045"`is added to the filter text box. If you then click **SID** again and change the SID filter to `1044`, the filter changes to `SID:"1044"`.

Table 5: Rule Content Filters

This filter...	Finds rules that...
Message	contain the supplied string in the message field.
SID	have the specified SID.
GID	have the specified GID.
Reference	contain the supplied string in the reference field. You can also filter by a specific type of reference and supplied string.
Action	start with `alert` or `pass`.
Protocol	include the selected protocol.
Direction	are based on whether the rule includes the indicated directional setting.
Source IP	use the specified addresses or variables for the source IP address designation in the rule. You can filter by a valid IP address, a CIDR block/prefix length, or using variables such as `$HOME_NET` or `$EXTERNAL_NET`.
Destination IP	use the specified addresses or variables for the source IP address designation in the rule. You can filter by a valid IP address, a CIDR block/prefix length, or using variables such as `$HOME_NET` or `$EXTERNAL_NET`.
Source port	include the specified source port. The port value must be an integer between 1 and 65535 or a port variable.
Destination port	include the specified destination port. The port value must be an integer between 1 and 65535 or a port variable.
Rule Overhead	have the selected rule overhead.
Metadata	have metadata containing the matching *key value* pair. For example, type `metadata:"service http"` to locate rules with metadata relating to the HTTP application protocol.

Intrusion Rule Categories

The system places rules in categories based on the type of traffic the rule detects. On the Rules page, you can filter by rule category, so you can set a rule attribute for all rules in a category. For example, if you do not have Linux hosts on your network, you could filter by the **os-linux** category, then disable all the rules showing to disable the entire **os-linux** category.

You can hover your pointer over a category name to display the number of rules in that category.

Note The Talos Intelligence Group may use the rule update mechanism to add and remove rule categories.

Intrusion Rule Filter Components

You can edit your filter to modify the special keywords and their arguments that are supplied when you click on a filter in the filter panel. Custom filters on the Rules page function like those used in the rule editor, but you can also use any of the keywords supplied in the Rules page filter, using the syntax displayed when you select the filter through the filter panel. To determine a keyword for future use, click on the appropriate argument in the filter panel on the right. The filter keyword and argument syntax appear in the filter text box. Remember that comma-separated multiple arguments for a keyword are only supported for the Category and Priority filter types.

You can use keywords and arguments, character strings, and literal character strings in quotes, with spaces separating multiple filter conditions. A filter cannot include regular expressions, wild card characters, or any special operator such as a negation character (!), a greater than symbol (>), less than symbol (<), and so on. When you type in search terms without a keyword, without initial capitalization of the keyword, or without quotes around the argument, the search is treated as a string search and the category, message, and SID fields are searched for the specified terms.

Except for the `gid` and `sid` keywords, all arguments and strings are treated as partial strings. Arguments for `gid` and `sid` return only exact matches.

Each rule filter can include one or more keywords in the format:

`keyword``:"`*`argument`*`"`

where keyword is one of the keywords in the intrusion rule filter groups and argument is enclosed in double quotes and is a single, case-insensitive, alphanumeric string to search for in the specific field or fields relevant to the keyword. Note that keywords should be typed with initial capitalization.

Arguments for all keywords except `gid` and `sid` are treated as partial strings. For example, the argument `123` returns `"12345"`, `"41235"`, `"45123",` and so on. The arguments for `gid` and `sid` return only exact matches; for example, `sid:3080` returns only `SID 3080`.

Each rule filter can also include one or more alphanumeric character strings. Character strings search the rule Message field, Snort ID (SID), and Generator ID (GID). For example, the string `123` returns the strings `"Lotus123"`, `"123mania"`, and so on in the rule message, and also returns `SID 6123`, `SID 12375`, and so on. You can search for a partial SID by filtering with one or more character strings.

All character strings are case-insensitive and are treated as partial strings. For example, any of the strings `ADMIN`, `admin`, or `Admin` return `"admin"`, `"CFADMIN"`, `"Administrator"` and so on.

You can enclose character strings in quotes to return exact matches. For example, the literal string `"overflow attempt"` in quotes returns only that exact string, whereas a filter comprised of the two strings `overflow` and `attempt` without quotes returns `"overflow attempt"`, `"overflow multipacket attempt"`, `"overflow with evasion attempt"`, and so on.

You can narrow filter results by entering any combination of keywords, character strings, or both, separated by spaces. The result includes any rule that matches all the filter conditions.

You can enter multiple filter conditions in any order. For example, each of the following filters returns the same rules:

- `url:at login attempt cve:200`
- `login attempt cve:200 url:at`
- `login cve:200 attempt url:at`

Intrusion Rule Filter Usage

You can select predefined filter keywords from the filter panel on the left side of the Rules page in the intrusion policy. When you select a filter, the page displays all matching rules, or indicates when no rules match.

You can add keywords to a filter to further constrain it. Any filter you enter searches the entire rules database and returns all matching rules. When you enter a filter while the page still displays the result of a previous filter, the page clears and returns the result of the new filter instead.

You can also type a filter using the same keyword and argument syntax supplied when you select a filter, or modify argument values in a filter after you select it. When you type in search terms without a keyword, without initial capitalization of the keyword, or without quotes around the argument, the search is treated as a string search and the category, message, and SID fields are searched for the specified terms.

Setting a Rule Filter in an Intrusion Policy

You can filter the rules on the Rules page to display a subset of rules. You can then use any of the page features, including choosing any of the features available in the context menu. This can be useful, for example, when you want to set a threshold for all the rules in a specific category. You can use the same features with rules in a filtered or unfiltered list. For example, you can apply new rule states to rules in a filtered or unfiltered list.

All filter keywords, keyword arguments, and character strings are case-insensitive. If you click an argument for a keyword already in the filter, it replaces the existing argument.

Procedure

Step 1 Choose **Policies** > **Access Control heading** > **Intrusion**.

Step 2 Click **Snort 2 Version** next to the policy you want to edit.

If **View** ( ) appears instead, the configuration belongs to an ancestor domain, or you do not have permission to modify the configuration.

Step 3 Construct a filter using any of the following methods, separately or in combination:

- Enter a value in the **Filter** text box, and press Enter.
- Expand any of the predefined keywords. For example, click **Rule Configuration**.
- Click a keyword, and specify an argument value if prompted. For example:
  - Under **Rule Configuration**, you could click **Rule State**, choose `Generate Events` from the drop-down-list, and click **OK**.
  - Under **Rule Configuration**, you could click **Comment**, enter the string of comment text to filter by, and click **OK**.
  - Under **Category**, you could click **app-detect**, which the system uses as the argument value.
- Expand a keyword, and click an argument value. For example, expand **Rule State** and click **Generate Events**.

Intrusion Rule States

Intrusion rule states allow you to enable or disable the rule within an individual intrusion policy, as well as specify which action the system takes if monitored conditions trigger the rule.

The Talos Intelligence Group sets the default state of each intrusion and preprocessor rule in each default policy. For example, a rule may be enabled in the Security over Connectivity default policy and disabled in the Connectivity over Security default policy. Talos sometimes uses a rule update to change the default state of one or more rules in a default policy. If you allow rule updates to update your base policy, you also allow the rule update to change the default state of a rule in your policy when the default state changes in the default policy you used to create your policy (or in the default policy it is based on). Note, however, that if you have changed the rule state, the rule update does not override your change.

When you create an intrusion rule, it inherits the default states of the rules in the default policy you use to create your policy.

Intrusion Rule State Options

In an intrusion policy, you can set a rule's state to the following values:

Generate Events

You want the system to detect a specific intrusion attempt and generate an intrusion event when it finds matching traffic. When a malicious packet crosses your network and triggers the rule, the packet is sent to its destination and the system generates an intrusion event. The malicious packet reaches its target, but you are notified via the event logging.

Drop and Generate Events

You want the system to detect a specific intrusion attempt, drop the packet containing the attack, and generate an intrusion event when it finds matching traffic. The malicious packet never reaches its target, and you are notified via the event logging.

Note that rules set to this rule state generate events but do not drop packets in a passive deployment. For the system to drop packets, **Drop when Inline** must also be enabled (the default setting) in your intrusion policy and you must deploy your device inline.

Disable

You do not want the system to evaluate matching traffic.

Note Choosing either the **Generate Events** or **Drop and Generate Events** options enables the rule. Choosing **Disable** disables the rule.

Cisco **strongly** recommends that you **do not** enable all the intrusion rules in an intrusion policy. The performance of your managed device is likely to degrade if all rules are enabled. Instead, tune your rule set to match your network environment as closely as possible.

Setting Intrusion Rule States

Intrusion rule states are policy-specific.

Procedure

Step 1 Choose **Policies** > **Access Control heading** > **Intrusion**.

Step 2 Click **Snort 2 Version** next to the policy you want to edit.

If **View** ( ) appears instead, the configuration belongs to an ancestor domain, or you do not have permission to modify the configuration.

Tip
This page indicates the total number of enabled rules, the total number of enabled rules set to Generate Events, and the total number set to Drop and Generate Events. Note also that in a passive deployment, rules set to Drop and Generate Events only generate events.

Step 3 Click **Rules** immediately under **Policy Information** in the navigation panel.

Step 4 Choose the rule or rules where you want to set the rule state.

Step 5 Choose one of the following:

- **Rule State** > **Generate Events**
- **Rule State** > **Drop and Generate Events**
- **Rule State** > **Disable**

Step 6 To save changes you made in this policy since the last policy commit, click **Policy Information** in the navigation panel, then click **Commit Changes**.

If you leave the policy without committing changes, changes since the last commit are discarded if you edit a different policy.

What to do next

- Deploy configuration changes.

Intrusion Event Notification Filters in an Intrusion Policy

The importance of an intrusion event can be based on frequency of occurrence, or on source or destination IP address. In some cases you may not care about an event until it has occurred a certain number of times. For example, you may not be concerned if someone attempts to log into a server until they fail a certain number of times. In other cases, you may only need to see a few occurrences to know there is a widespread problem. For example, if a DoS attack is launched against your web server, you may only need to see a few occurrences of an intrusion event to know that you need to address the situation. Seeing hundreds of the same event only overwhelms your system.

Intrusion Event Thresholds

You can set thresholds for individual rules, per intrusion policy, to limit the number of times the system logs and displays an intrusion event based on how many times the event is generated within a specified time period. This can prevent you from being overwhelmed with a large number of identical events. You can set thresholds per shared object rule, standard text rule, or preprocessor rule.

Intrusion Event Thresholds Configuration

To set a threshold, first specify the thresholding type.

Table 6: Thresholding Options

Option	Description
Limit	Logs and displays events for the specified number of packets (specified by the Count argument) that trigger the rule during the specified time period. For example, if you set the type to **Limit**, the **Count** to `10`, and the **Seconds** to `60`, and 14 packets trigger the rule, the system stops logging events for the rule after displaying the first 10 that occur within the same minute.
Threshold	Logs and displays a single event when the specified number of packets (specified by the Count argument) trigger the rule during the specified time period. Note that the counter for the time restarts after you hit the threshold count of events and the system logs that event. For example, you set the type to **Threshold**, **Count** to `10`, and **Seconds** to `60`, and the rule triggers 10 times by second 33. The system generates one event, then resets the Seconds and Count counters to 0. The rule then triggers another 10 times in the next 25 seconds. Because the counters reset to 0 at second 33, the system logs another event.
Both	Logs and displays an event once per specified time period, after the specified number (count) of packets trigger the rule. For example, if you set the type to **Both**, **Count** to two, and **Seconds** to `10`, the following event counts result: • If the rule is triggered once in 10 seconds, the system does not generate any events (the threshold is not met) • If the rule is triggered twice in 10 seconds, the system generates one event (the threshold is met when the rule triggers the second time) • If the rule is triggered four times in 10 seconds, the system generates one event (the threshold is met when the rule triggers the second time, and following events are ignored)

Next, specify tracking, which determines whether the event threshold is calculated per source or destination IP address.

Table 7: Thresholding IP Options

Option	Description
Source	Calculates event instance count per source IP address.
Destination	Calculates event instance count per destination IP address.

Finally, specify the number of instances and time period that define the threshold.

Table 8: Thresholding Instance/Time Options

Option	Description
Count	The number of event instances per specified time period per tracking IP address required to meet the threshold.

Option	Description
Seconds	The number of seconds that elapse before the count resets. If you set the threshold type to **limit**, the tracking to **Source IP**, the **count** to `10`, and the **seconds** to `10`, the system logs and displays the first 10 events that occur in 10 seconds from a given source port. If only 7 events occur in the first 10 seconds, the system logs and displays those; if 40 events occur in the first 10 seconds, the system logs and displays 10, then begins counting again when the 10-second time period elapses.

Note that you can use intrusion event thresholding alone or in any combination with rate-based attack prevention, the `detection_filter` keyword, and intrusion event suppression.

Tip You can also add thresholds from within the packet view of an intrusion event.

Related Topics

The detection_filter Keyword, on page 182

Adding and Modifying Intrusion Event Thresholds

You can set a threshold for one or more specific rules in an intrusion policy. You can also separately or simultaneously modify existing threshold settings. You can set a single threshold for each. Adding a threshold overwrites any existing threshold for the rule.

You can also modify the global threshold that applies by default to all rules and preprocessor-generated events associated with the intrusion policy.

A **Revert** appears in a field when you enter an invalid value; click it to revert to the last valid value for that field or to clear the field if there was no previous value.

Tip A global or individual threshold on a managed device with multiple CPUs may result in a higher number of events than expected.

Procedure

Step 1 Choose **Policies** > **Access Control heading** > **Intrusion**.

Step 2 Click **Snort 2 Version** next to the policy you want to edit.

If **View** ( ) appears instead, the configuration belongs to an ancestor domain, or you do not have permission to modify the configuration.

Step 3 Click **Rules** immediately under **Policy Information** in the navigation pane.

Step 4 Choose the rule or rules where you want to set a threshold.

Step 5 Choose **Event Filtering** > **Threshold**.

Step 6 Choose a threshold type from the **Type** drop-down list.

Step 7 From the **Track By** drop-down list, choose whether you want the event instances tracked by **Source** or **Destination** IP address.

Step 8 Enter a value in the **Count** field.

Step 9 Enter a value in the **Seconds** field.

Step 10 Click **OK**.

Tip
The system displays an **Event Filter** next to the rule in the Event Filtering column. If you add multiple event filters to a rule, a number over the filter indicates the number of event filters.

Step 11 To save changes you made in this policy since the last policy commit, click **Policy Information**, then click **Commit Changes**.

If you leave the policy without committing changes, changes since the last commit are discarded if you edit a different policy.

What to do next

- Deploy configuration changes.

Viewing and Deleting Intrusion Event Thresholds

You may want to view or delete an existing threshold setting for a rule. You can use the Rules Details view to display the configured settings for a threshold to see if they are appropriate for your system. If they are not, you can add a new threshold to overwrite the existing values.

Note that you can also modify the global threshold that applies by default to all rules and preprocessor-generated events logged by the intrusion policy.

Procedure

Step 1 Choose **Policies** > **Access Control heading** > **Intrusion**.

Step 2 Click **Snort 2 Version** next to the policy you want to edit.

If **View** ( ) appears instead, the configuration belongs to an ancestor domain, or you do not have permission to modify the configuration.

Step 3 Click **Rules** immediately under **Policy Information** in the navigation pane.

Step 4 Choose the rule or rules with a configured threshold you want to view or delete.

Step 5 To remove the threshold for each selected rule, choose **Event Filtering** > **Remove Thresholds**.

Step 6 Click **OK**.

Step 7 To save changes you made in this policy since the last policy commit, click **Policy Information**, then click **Commit Changes**.

If you leave the policy without committing changes, changes since the last commit are discarded if you edit a different policy.

What to do next

- Deploy configuration changes.

Intrusion Policy Suppression Configuration

You can suppress intrusion event notification when a specific IP address or range of IP addresses triggers a specific rule or preprocessor. This is useful for eliminating false positives. For example, if you have a mail server that transmits packets that look like a specific exploit, you might suppress event notification for that event when it is triggered by your mail server. The rule triggers for all packets, but you only see events for legitimate attacks.

Intrusion Policy Suppression Types

Note that you can use intrusion event suppression alone or in any combination with rate-based attack prevention, the `detection_filter` keyword, and intrusion event thresholding.

Tip You can add suppressions from within the packet view of an intrusion event. You can also access suppression settings by using the right-click context menu on the intrusion rules editor page (**Objects** > **Intrusion Rules**) and on any intrusion event page (if the event was triggered by an intrusion rule).

Related Topics

The detection_filter Keyword, on page 182

Suppressing Intrusion Events for a Specific Rule

You can suppress intrusion event notification for a rule or rules in your intrusion policy. When notification is suppressed for a rule, the rule triggers but events are not generated. You can set one or more suppressions for a rule. The first suppression listed has the highest priority. When two suppressions conflict, the action of the first is carried out.

Note that a **Revert** appears in a field when you enter an invalid value; click it to revert to the last valid value for that field or to clear the field if there was no previous value.

Procedure

Step 1 Choose **Policies** > **Access Control heading** > **Intrusion**.

Step 2 Click **Snort 2 Version** next to the policy you want to edit.

If **View** ( ) appears instead, the configuration belongs to an ancestor domain, or you do not have permission to modify the configuration.

Step 3 Click **Rules** immediately under **Policy Information** in the navigation panel.

Step 4 Choose the rule or rules for which you want to configure suppression conditions.

Step 5 Choose **Event Filtering** > **Suppression**.

Step 6 Choose a **Suppression Type**.

Step 7 If you chose **Source** or **Destination** for the suppression type, in the **Network** field enter the IP address, address block, or variable you want to specify as the source or destination IP address, or a comma-separated list comprised of any combination of these.

Step 8 Click **OK**.

Tip

The system displays an **Event Filter** next to the rule in the Event Filtering column next the suppressed rule. If you add multiple event filters to a rule, a number over the filter indicates the number of event filters.

Step 9 To save changes you made in this policy since the last policy commit, click **Policy Information**, then click **Commit Changes**.

If you leave the policy without committing changes, changes since the last commit are discarded if you edit a different policy.

What to do next

- Deploy configuration changes.

Viewing and Deleting Suppression Conditions

You may want to view or delete an existing suppression condition. For example, you can suppress event notification for packets originating from a mail server IP address because the mail server normally transmits packets that look like exploits. If you then decommission that mail server and reassign the IP address to another host, you should delete the suppression conditions for that source IP address.

Procedure

Step 1 Choose **Policies** > **Access Control heading** > **Intrusion**.

Step 2 Click **Snort 2 Version** next to the policy you want to edit.

If **View** ( ) appears instead, the configuration belongs to an ancestor domain, or you do not have permission to modify the configuration.

Step 3 Click **Rules** immediately under **Policy Information** in the navigation panel.

Step 4 Choose the rule or rules for which you want to view or delete suppressions.

Step 5 You have the following choices:

- To remove all suppression for a rule, choose **Event Filtering** > **Remove Suppressions**.
- To remove a specific suppression setting, click the rule, then click **Show details**. Expand the suppression settings and click **Delete** next to the suppression settings you want to remove.

Step 6 Click **OK**.

Step 7 To save changes you made in this policy since the last policy commit, click **Policy Information**, then click **Commit Changes**.

If you leave the policy without committing changes, changes since the last commit are discarded if you edit a different policy.

What to do next

- Deploy configuration changes.

Dynamic Intrusion Rule States

Rate-based attacks attempt to overwhelm a network or host by sending excessive traffic toward the network or host, causing it to slow down or deny legitimate requests. You can use rate-based prevention to change the action of a rule in response to excessive rule matches for specific rules.

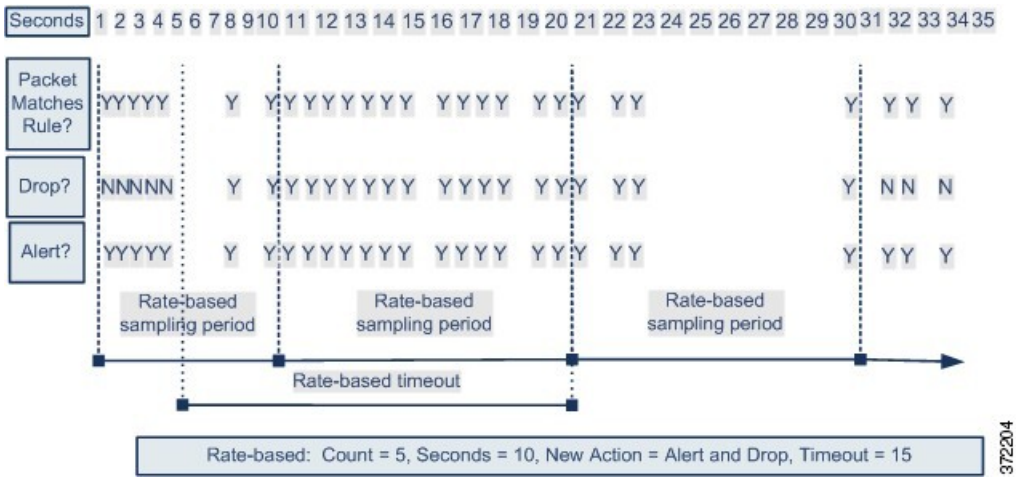
You can configure your intrusion policies to include a rate-based filter that detects when too many matches for a rule occur in a given time period. You can use this feature on managed devices deployed inline to block rate-based attacks for a specified time, then revert to a rule state where rule matches only generate events and do not drop traffic.

Rate-based attack prevention identifies abnormal traffic patterns and attempts to minimize the impact of that traffic on legitimate requests. You can identify excessive rule matches in traffic going to a particular destination IP address or addresses or coming from a particular source IP address or addresses. You can also respond to excessive matches for a particular rule across all detected traffic.

In some cases, you may not want to set a rule to the Drop and Generate Events state because you do not want to drop every packet that matches the rule, but you do want to drop packets matching the rule if a particular rate of matches occurs in a specified time. Dynamic rule states let you configure the rate that should trigger a change in the action for a rule, what the action should change to when the rate is met, and how long the new action should persist.

The following diagram shows an example where an attacker is attempting to access a host. Repeated attempts to find a password trigger a rule which has rate-based attack prevention configured. The rate-based settings change the rule attribute to Drop and Generate Events after rule matches occur five times in a 10-second span. The new rule attribute times out after 15 seconds.

After the timeout, note that packets are still dropped in the rate-based sampling period that follows. If the sampled rate is above the threshold in the current or previous sampling period, the new action continues. The new action reverts to Generate Events only after a sampling period completes where the sampled rate was below the threshold rate.



Dynamic Intrusion Rule State Configuration

In the intrusion policy, you can configure a rate-based filter for any intrusion or preprocessor rule. The rate-based filter contains three components:

- the rule matching rate, which you configure as a count of rule matches within a specific number of seconds

- a new action to be taken when the rate is exceeded, with three available actions: Generate Events, Drop and Generate Events, and Disable
- the duration of the action, which you configure as a timeout value

Note that when started, the new action occurs until the timeout is reached, even if the rate falls below the configured rate during that time period. When the timeout is reached, if the rate has fallen below the threshold, the action for the rule reverts to the action initially configured for the rule.

You can configure rate-based attack prevention in an inline deployment to block attacks, either temporarily or permanently. Without rate-based configuration, rules set to Generate Events do generate events, but the system does not drop packets for those rules. However, if the attack traffic matches rules that have rate-based criteria configured, the rate action may cause packet dropping to occur for the period of time that the rate action is active, even if those rules are not initially set to Drop and Generate Events.

Note Rate-based actions cannot enable disabled rules or drop traffic that matches disabled rules.

You can define multiple rate-based filters on the same rule. The first filter listed in the intrusion policy has the highest priority. Note that when two rate-based filter actions conflict, the action of the first rate-based filter is carried out.

Setting a Dynamic Rule State from the Rules Page

You can set one or more dynamic rule states for a rule. The first dynamic rule state listed has the highest priority. When two dynamic rule states conflict, the action of the first is carried out.

Dynamic rule states are policy-specific.

A **Revert** appears in a field when you enter an invalid value; click it to revert to the last valid value for that field or to clear the field if there was no previous value.

Note Dynamic rule states cannot enable disabled rules or drop traffic that matches disabled rules.

Procedure

Step 1 Choose **Policies** > **Access Control heading** > **Intrusion**.

Step 2 Click **Snort 2 Version** next to the policy you want to edit.

If **View** ( ) appears instead, the configuration belongs to an ancestor domain, or you do not have permission to modify the configuration.

Step 3 Click **Rules** immediately under **Policy Information** in the navigation pane.

Step 4 Choose the rule or rules where you want to add a dynamic rule state.

Step 5 Choose **Dynamic State** > **Add Rate-Based Rule State**.

Step 6 Choose a value from the **Track By** drop-down list.

Step 7 If you set **Track By** to **Source** or **Destination**, enter the address of each host you want to track in the **Network** field. You can specify a single IP address, address block, variable, or a comma-separated list comprised of any combination of these.

Step 8 Next to **Rate**, specify the number of rule matches per time period to set the attack rate:

- Enter a value in the **Count** field.
- Enter a value in the **Seconds** field.

Step 9 From the **New State** drop-down list, specify the new action to be taken when the conditions are met.

Step 10 Enter a value in the **Timeout** field.

After the timeout occurs, the rule reverts to its original state. Specify `0` or leave the **Timeout** field blank to prevent the new action from timing out.

Step 11 Click **OK**.

Tip
The system displays a **Dynamic State** next to the rule in the Dynamic State column. If you add multiple dynamic rule state filters to a rule, a number over the filter indicates the number of filters.

Tip
To delete all dynamic rule settings for a set of rules, choose the rules on the Rules page, then choose **Dynamic State** > **Remove Rate-Based States**. You can also delete individual rate-based rule state filters from the rule details for the rule by choosing the rule, clicking **Show details**, then clicking **Delete** by the rate-based filter you want to remove.

Step 12 To save changes you made in this policy since the last policy commit, click **Policy Information**, then click **Commit Changes**.

If you leave the policy without committing changes, changes since the last commit are discarded if you edit a different policy.

What to do next

- Deploy configuration changes.

Adding Intrusion Rule Comments

You can add comments to rules in your intrusion policy. Comments added this way are policy-specific; that is, comments you add to a rule in one intrusion policy are not visible in other intrusion policies. Any comments you add can be seen in the Rule Details view on the Rules page for the intrusion policy.

After you commit the intrusion policy changes containing the comment, you can also view the comment by clicking **Rule Comment** on the rule Edit page.

Procedure

Step 1 Choose **Policies** > **Access Control heading** > **Intrusion**.

Step 2 Click **Snort 2 Version** next to the policy you want to edit.

If **View** ( ) appears instead, the configuration belongs to an ancestor domain, or you do not have permission to modify the configuration.

Step 3 Click **Rules** immediately under **Policy Information** in the navigation panel.

Step 4 Choose the rule or rules where you want to add a comment.

Step 5 Choose **Comments** > **Add Rule Comment**.

Step 6 In the **Comment** field, enter the rule comment.

Step 7 Click **OK**.

Tip
The system displays a **Comment** ( ) next to the rule in the Comments column. If you add multiple comments to a rule, a number over the comment indicates the number of comments.

Step 8 Optionally, delete a rule comment by clicking **Delete** next to the comment.

You can only delete a comment if the comment is cached with uncommitted intrusion policy changes. After intrusion policy changes are committed, the rule comment is permanent.

Step 9 To save changes you made in this policy since the last policy commit, click **Policy Information**, then click **Commit Changes**.

If you leave the policy without committing changes, changes since the last commit are discarded if you edit a different policy.

What to do next

- Deploy configuration changes.

Tailoring Intrusion Protection to Your Network Assets

The following topics describe how to use Cisco recommended rules:

About Cisco Recommended Rules

You can use intrusion rule recommendations to associate the operating systems, servers, and client application protocols detected on your network with rules specifically written to protect those assets. This allows you to tailor your intrusion policy to the specific needs of your monitored network.

The system makes an individual set of recommendations for each intrusion policy. It typically recommends rule state changes for standard text rules and shared object rules. However, it can also recommend changes for preprocessor and decoder rules.

When you generate rule state recommendations, you can use the default settings or configure advanced settings. Advanced settings allow you to:

- Redefine which hosts on your network the system monitors for vulnerabilities
- Influence which rules the system recommends based on rule overhead
- Specify whether to generate recommendations to disable rules

You can also choose either to use the recommendations immediately or to review the recommendations (and affected rules) before accepting them.

Choosing to use recommended rule states adds a read-only Cisco Recommendations layer to your intrusion policy, and subsequently choosing not to use recommended rule states removes the layer.

You can schedule a task to generate recommendations automatically based on the most recently saved configuration settings in your intrusion policy.

The system does not change rule states that you set manually:

- Manually setting the states of specified rules *before* you generate recommendations prevents the system from modifying the states of those rules in the future.
- Manually setting the states of specified rules *after* you generate recommendations overrides the recommended states of those rules.

Tip The intrusion policy report can include a list of rules with rule states that differ from the recommended state.

While displaying the recommendation-filtered Rules page, or after accessing the Rules page directly from the navigation panel or the Policy Information page, you can manually set rule states, sort rules, and take any of the other actions available on the Rules page, such as suppressing rules, setting rule thresholds, and so on.

Note The Talos Intelligence Group determines the appropriate state of each rule in the system-provided policies. If you use a system-provided policy as your base policy, and you allow the system to set your rules to the Cisco recommended rule state, the rules in your intrusion policy match the settings recommended by Cisco for your network assets.

Default Settings for Cisco Recommendations

When you generate Cisco recommendations, the system searches your base policy for rules that protect against vulnerabilities associated with your network assets, and identifies the current state of rules in your base policy. The system then recommends rule states and, if you choose to, sets the rules to the recommended states.

The system performs the following basic analysis to generate recommendations:

Table 9: Rule State Recommendations Based on Vulnerabilities

Rule Protects Discovered Assets?	Base Policy Rule State	Recommend Rule State
Yes	Disabled	Generate Events
	Generate Events	Generate Events
	Drop and Generate Events	Drop and Generate Events
No	Any	Disabled

Note the following in the table:

- If a rule is disabled in the base policy, or set to Generate Events, the recommended state is always Generate Events.

  For example, if the base policy is No Rules Active, in which all rules are disabled, there will be no recommendations to Drop and Generate Events.

- Recommendations to Drop and Generate Events are made only for rules already set to Drop and Generate Events in the base policy.

  If you want a rule to be set to Drop and Generate events and the rule was disabled or set to Generate Events in the base policy, you must manually reset the rule state.

When you generate recommendations without changing the advanced settings for Cisco recommended rules, the system recommends rule state changes for all hosts in your entire discovered network.

By default, the system generates recommendations only for rules with low or medium overhead, and generates recommendations to disable rules.

The system does not recommend a rule state for an intrusion rule that is based on a vulnerability that you disable using the Impact Qualification feature.

The system always recommends that you enable a local rule associated with a third-party vulnerability mapped to a host.

The system does not make state recommendations for unmapped local rules.

Related Topics

Third-Party Product Mappings

Advanced Settings for Cisco Recommendations

Include all differences between recommendations and rule states in policy reports

By default, an intrusion policy report lists the policy's enabled rules, that is, rules set to either Generate Events or Drop and Generate Events. Enabling the **Include all differences** option also lists the rules whose recommended states differ from their saved states.

Networks to Examine

Specifies the monitored networks or individual hosts to examine for recommendations. You can specify a single IP address or address block, or a comma-separated list comprised of either or both.

Lists of addresses within the hosts that you specify are linked with an OR operation except for negations, which are linked with an AND operation after all OR operations are calculated.

If you want to dynamically adapt active rule processing for specific packets based on host information, you can also enable adaptive profile updates.

Recommendation Threshold (By Rule Overhead)

Prevents the system from recommending or automatically enabling intrusion rules with a higher overhead than the threshold you choose.

Overhead is based on the rule’s potential impact on system performance and the likelihood that the rule may generate false positives. Permitting rules with higher overhead usually results in more recommendations, but can affect system performance. You can view the overhead rating for a rule in the rule detail view on the intrusion Rules page.

Note that the system does not factor rule overhead into recommendations to disable rules. Also, local rules are considered to have no overhead, unless they are mapped to a third-party vulnerability.

Generating recommendations for rules with the overhead rating at a particular setting does not preclude you from generating recommendations with different overhead, then generating recommendations again for the original overhead setting. You get the same rule state recommendations for each overhead setting each time you generate recommendations for the same rule set, regardless of the number of times you generate recommendations or how many different overhead settings you generate with. For example, you can generate recommendations with overhead set to medium, then to high, then finally to medium again; if the hosts and applications on your network have not changed, both sets of recommendations with overhead set to medium are then the same for that rule set.

Accept Recommendations to Disable Rules

Specifies whether the system disables intrusion rules based on Cisco recommendations.

Accepting recommendations to disable rules restricts your rule coverage. Omitting recommendations to disable rules augments your rule coverage.

Related Topics

Adaptive profile updates and recommended rules

Generating and Applying Cisco Recommendations

Starting or stopping use of Cisco recommendations may take several minutes, depending on the size of your network and intrusion rule set.

Before you begin

- Cisco recommendations have the following requirements:
  - Threat Defense License—IPS
  - User roles—Admin or Intrusion Admin
- Configure a network discovery policy before you begin with the steps. Configure the network discovery policy to define internal hosts so that the Cisco recommendations are suitable.

Procedure

Step 1 In the Snort 2 intrusion policy editor's navigation pane, click **Cisco Recommendations**.

Step 2 (Optional) Configure advanced settings; see Advanced Settings for Cisco Recommendations, on page 54.

Step 3 Generate and apply recommendations.

- **Generate and Use Recommendations**—Generates recommendations and changes rule states to match. Only available if you have never generated recommendations.
- **Generate Recommendations**—Regardless of whether you are using recommendations, generates new recommendations but does not change rule states to match.
- **Update Recommendations**—If you are using recommendations, generates recommendations and changes rule states to match. Otherwise, generates new recommendations without changing rule states.
- **Use Recommendations**—Changes rule states to match any unimplemented recommendations.

- **Do Not Use Recommendations**—Stops use of recommendations. If you manually changed a rule's state before you applied recommendations, the rule state returns to the value you gave it. Otherwise, the rule state returns to its default value.

When you generate recommendations, the system displays a summary of the recommended changes. To view a list of rules where the system recommends a state change, click **View** next to the newly proposed rule state.

Step 4 Evaluate and adjust the recommendations you implemented.

Even if you accept most Cisco recommendations, you can override individual recommendations by setting rule states manually; see Setting Intrusion Rule States, on page 42.

Step 5 To save changes you made in this policy since the last policy commit, click **Policy Information**, then click **Commit Changes**.

If you leave the policy without committing changes, changes since the last commit are discarded if you edit a different policy.

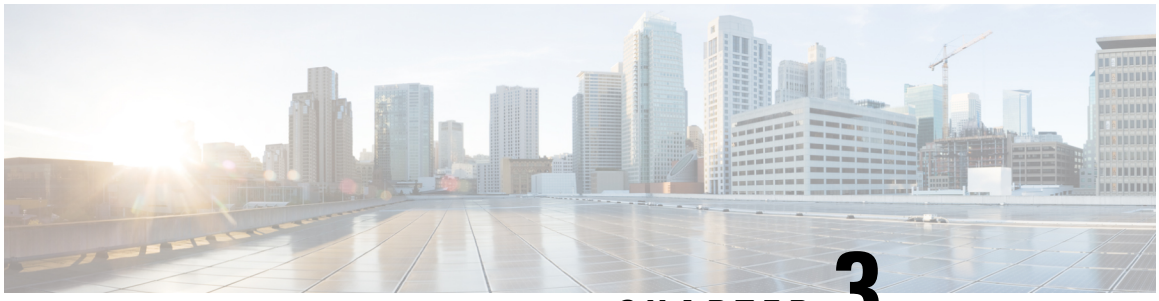
What to do next

- Deploy configuration changes.

Script Detection

The script detection prevents the Snort blocks-too-late intrusion failures with a partial inspection. When HTML files are transferred between a client and a server, these files can contain malicious scripts, such as JavaScript, to initiate an attack. When such malicious scripts are found, the partial inspection allows any IPS rule to match on the malicious script, and the inspector flushes that data segment through inspection and detection. The malicious file never reaches its destination. This feature supports both HTTP/1 and HTTP/2 traffic.

This feature is always enabled by default. To turn it off, set `http_inspect.script_detection=true` to false.



CHAPTER 3

Intrusion Prevention Performance Tuning

The following topics describe how to refine intrusion prevention performance:

About Intrusion Prevention Performance Tuning

Cisco provides several features for improving the performance of your system as it analyzes traffic for attempted intrusions. You can:

- specify the number of packets to allow in the event queue. You can also, before and after stream reassembly, enable or disable inspection of packets that will be rebuilt into larger streams.
- override default match and recursion limits on PCRE that are used in intrusion rules to examine packet payload content.
- elect to have the rules engine log more than one event per packet or packet stream when multiple events are generated, allowing you to collect information beyond the reported event.
- balance security with the need to maintain device latency at an acceptable level with packet and rule latency thresholding.
- configure the basic parameters of how devices monitor and report their own performance. This allows you to specify the intervals at which the system updates performance statistics on your devices.

You configure these performance settings on a per-access-control-policy basis, and they apply to all intrusion policies invoked by that parent access control policy.

License Requirements for Intrusion Prevention Performance Tuning

Threat Defense License

IPS

Requirements and Prerequisites for Intrusion Prevention Performance Tuning

Model support

Any.

Supported domains

Any

User roles

- Admin
- Access Admin
- Network Admin

Limiting Pattern Matching for Intrusions

Procedure

Step 1 In the access control policy editor, click **Advanced** (**Policies** > **Access Control heading** > **Access Control**, click **Edit** and then click **Advanced Settings**).

In the new UI, select **Advanced Settings** from the drop-down arrow at the end of the packet flow line.

Step 2 Click **Edit** (✎) next to **Performance Settings**.

If **View** (👁) appears instead, settings are inherited from an ancestor policy, or you do not have permission to modify the settings. If the configuration is unlocked, uncheck **Inherit from base policy** to enable editing.

Step 3 Click **Pattern Matching Limits** in the **Performance Settings** pop-up window.

Step 4 Enter a value for the maximum number of events to queue in the **Maximum Pattern States to Analyze Per Packet** field.

Step 5 To disable the inspection of packets that will be rebuilt into larger streams of data before and after stream reassembly in Snort 2, check the **Disable Content Checks on Traffic Subject to Future Reassembly** check box. Inspection before and after reassembly requires more processing overhead and may decrease performance.

Important
In Snort 3, the **Disable Content Checks on Traffic Subject to Future Reassembly** check box settings are:

- Checked—Indicates detecting TCP payload before reassembly. It includes inspection of packets before and after stream reassembly. This process requires more processing overhead and may decrease performance.
- Unchecked—Indicates detecting TCP payload after reassembly.

Step 6 Click **OK**.

Step 7 Click **Save** to save the policy.

What to do next

- Deploy configuration changes.

Regular Expression Limits Overrides for Intrusion Rules

The default regular expression limits ensure a minimum level of performance. Overriding these limits could increase security, but could also significantly impact performance by permitting packet evaluation against inefficient regular expressions.

Caution Do not override default PCRE limits unless you are an experienced intrusion rule writer with knowledge of the impact of degenerative patterns.

Table 10: Regular Expression Constraint Options

Option	Description
Match Limit State	Specifies whether to override **Match Limit**. You have the following options: • select **Default** to use the value configured for **Match Limit** • select **Unlimited** to permit an unlimited number of attempts • select **Custom** to specify either a limit of 1 or greater for **Match Limit**, or to specify 0 to completely disable PCRE match evaluations
Match Limit	Specifies the number of times to attempt to match a pattern defined in a PCRE regular expression.

Option	Description
Match Recursion Limit State	Specifies whether to override **Match Recursion Limit**. You have the following options: • select **Default** to use the value configured for **Match Recursion Limit** • select **Unlimited** to permit an unlimited number of recursions • select **Custom** to specify either a limit of 1 or greater for **Match Recursion Limit**, or to specify 0 to completely disable PCRE recursions Note that for **Match Recursion Limit** to be meaningful, it must be smaller than **Match Limit**.
Match Recursion Limit	Specifies the number of recursions when evaluating a PCRE regular expression against the packet payload.

Overriding Regular Expression Limits for Intrusion Rules

Procedure

Step 1 In the access control policy editor, click **Advanced**.

In the new UI, select **Advanced Settings** from the drop-down arrow at the end of the packet flow line.

Step 2 Click **Edit** (✎) next to **Performance Settings**.

If **View** (👁) appears instead, settings are inherited from an ancestor policy, or you do not have permission to modify the settings. If the configuration is unlocked, uncheck **Inherit from base policy** to enable editing.

Step 3 Click **Regular Expression Limits** in the **Performance Settings** pop-up window.

Step 4 You can modify any of the options as described in Regular Expression Limits Overrides for Intrusion Rules, on page 59.

Step 5 Click **OK**.

Step 6 Click **Save** to save the policy.

What to do next

- Deploy configuration changes.

Per Packet Intrusion Event Generation Limits

When the intrusion rules engine evaluates traffic against rules, it places the events generated for a given packet or packet stream in an event queue, then reports the top events in the queue to the user interface. When configuring the intrusion event logging limits, you can specify how many events can be placed in the queue and how many are logged, and select the criteria for determining event order within the queue.

Table 11: Intrusion Event Logging Limits Options

Option	Description
Maximum Events Stored Per Packet	The maximum number of events that can be stored for a given packet or packet stream.
Maximum Events Logged Per Packet	The number of events logged for a given packet or packet stream. This cannot exceed the **Maximum Events Stored Per Packet** value.
Prioritize Event Logging By	The value used to determine event ordering within the event queue. The highest ordered event is reported through the user interface. You can select from: • `priority`, which orders events in the queue by the event priority. • `content_length`, which orders events by the longest identified content match. When events are ordered by content length, rule events always take precedence over decoder and preprocessor events.

Limiting Intrusion Events Generated Per Packet

Procedure

Step 1 In the access control policy editor, click **Advanced**.

In the new UI, select **Advanced Settings** from the drop-down arrow at the end of the packet flow line.

Step 2 Click **Edit** (✎) next to **Performance Settings**.

If **View** (👁) appears instead, settings are inherited from an ancestor policy, or you do not have permission to modify the settings. If the configuration is unlocked, uncheck **Inherit from base policy** to enable editing.

Step 3 Click **Intrusion Event Logging Limits** in the **Performance Settings** pop-up window.

Step 4 You can modify any of the options in Per Packet Intrusion Event Generation Limits, on page 60.

Step 5 Click **OK**.

Step 6 Click **Save** to save the policy.

What to do next

- Deploy configuration changes.

Packet and Intrusion Rule Latency Threshold Configuration

Each access control policy has latency-based settings that use thresholding to manage packet and rule processing performance.

Packet latency thresholding measures the total elapsed time taken to process a packet by applicable decoders, preprocessors, and rules, and ceases inspection of the packet if the processing time exceeds a configurable threshold.

Rule latency thresholding measures the elapsed time each rule takes to process an individual packet, suspends the violating rule along with a group of related rules for a specified time if the processing time exceeds the rule latency threshold a configurable consecutive number of times, and restores the rules when the suspension expires.

Latency-Based Performance Settings

By default, the system takes latency-based performance settings from the latest intrusion rule update deployed on your system.

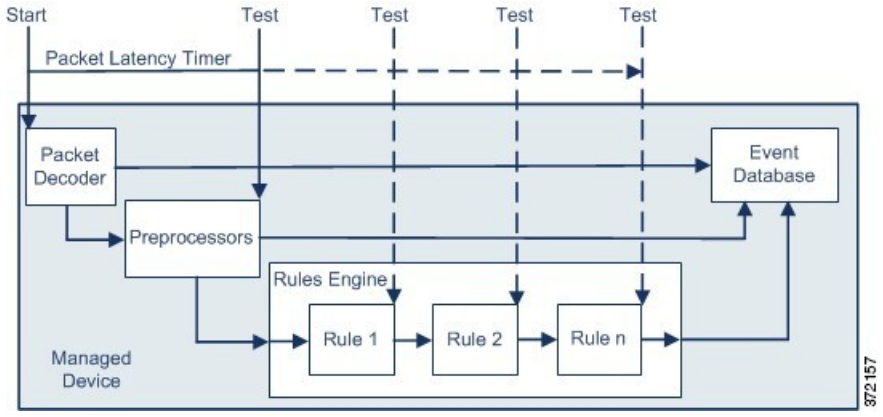
The latency settings that are actually applied depend on the security level of the network analysis policy (NAP) associated with the access control policy. Generally, this is the default NAP policy. However, if custom network analysis rules are configured, and if any of these specify a NAP policy that is more secure than the default NAP policy, then latency settings are based on the most secure NAP policy among the custom rules. If the default NAP policy or any custom rules invoke a custom NAP policy, then the security level used in the evaluation is the system-provided base policy on which each custom NAP policy is based.

The above is true regardless of whether the effective threshold and/or network analysis configurations are inherited or configured directly in the policy.

Packet Latency Thresholding

Packet latency thresholding measures elapsed time, not just processing time, in order to more accurately reflect the actual time required for the rule to process a packet. However, latency thresholding is a software-based latency implementation that does not enforce strict timing.

The trade-off for the performance and latency benefits derived from latency thresholding is that uninspected packets could contain attacks. A timer starts for each packet when decoder processing begins. Timing continues either until all processing ends for the packet or until the processing time exceeds the threshold at a timing test point.



As illustrated in the above figure, packet latency timing is tested at the following test points:

- after the completion of all decoder and preprocessor processing and before rule processing begins
- after processing by each rule

If the processing time exceeds the threshold at any test point, packet inspection ceases.

Tip Total packet processing time does not include routine TCP stream or IP fragment reassembly times.

Packet latency thresholding has no effect on events triggered by a decoder, preprocessor, or rule processing the packet. Any applicable decoder, preprocessor, or rule triggers normally until a packet is fully processed, or until packet processing ends because the latency threshold is exceeded, whichever comes first. If a drop rule detects an intrusion in an inline deployment, the drop rule triggers an event and the packet is dropped.

Note No packets are evaluated against rules after processing for that packet ceases because of a packet latency threshold violation. A rule that would have triggered an event cannot trigger that event, and for drop rules, cannot drop the packet.

Packet latency thresholding can improve system performance in both passive and inline deployments, and can reduce latency in inline deployments, by stopping inspection of packets that require excessive processing time. These performance benefits might occur when, for example:

- for both passive and inline deployments, sequential inspection of a packet by multiple rules requires an excessive amount of time
- for inline deployments, a period of poor network performance, such as when someone downloads an extremely large file, slows packet processing

In a passive deployment, stopping the processing of packets might not contribute to restoring network performance because processing simply moves to the next packet.

Packet Latency Thresholding Notes

By default, the latency-based performance settings for packet handling is disabled. You may choose to enable it. However, Cisco recommends that you do not change the default value for the threshold setting.

The information in this below applies only if you choose to specify custom values.

Table 12: Packet Latency Thresholding Option

Option	Description
Threshold (microseconds)	Specifies the time, in microseconds, when inspection of a packet ceases.

Enabling Packet Latency Thresholding

Procedure

Step 1 In the access control policy editor, click **Advanced**.

In the new UI, select **Advanced Settings** from the drop-down arrow at the end of the packet flow line.

Step 2 Click **Edit** (✎) next to **Latency-Based Performance Settings**.

If **View** ( ) appears instead, settings are inherited from an ancestor policy, or you do not have permission to modify the settings.

Step 3 Click **Packet Handling** in the **Latency-Based Performance Settings** pop-up window.

Step 4 Check the **Enabled** check box.

Step 5 Click **OK**.

Step 6 Click **Save** to save the policy.

What to do next

- Deploy configuration changes.

Configuring Packet Latency Thresholding

By default, the latency-based performance settings for packet handling is disabled. You may choose to enable it. However, Cisco recommends that you do not change the default value for the threshold setting.

Procedure

Step 1 In the access control policy editor, click **Advanced**.

In the new UI, select **Advanced Settings** from the drop-down arrow at the end of the packet flow line.

Step 2 Click **Edit** ( ) next to **Latency-Based Performance Settings**.

System ( ) > **Monitoring** > **Statistics**

Step 3 If the configuration is unlocked, uncheck **Inherit from base policy** to enable editing.

Step 4 Click **Packet Handling** in the **Latency-Based Performance Settings** pop-up window.

By default, **Installed Rule Update** is selected. We recommend using this default.

The values displayed do not reflect the automated settings.

Step 5 If you choose to specify custom values:

- Check the **Enabled** check box, and see Packet Latency Thresholding Notes, on page 63 for recommended minimum **Threshold** settings.
- You must specify custom values in both the packet handling tab and the rule handling tab.

Step 6 Click **OK**.

Step 7 Click **Save** to save the policy.

What to do next

- Deploy configuration changes.

Rule Latency Thresholding

Rule latency thresholding measures elapsed time, not just processing time, in order to more accurately reflect the actual time required for the rule to process a packet. However, latency thresholding is a software-based latency implementation that does not enforce strict timing.

The trade-off for the performance and latency benefits derived from latency thresholding is that uninspected packets could contain attacks. A timer measures the processing time each time a packet is processed against a group of rules. Any time the rule processing time exceeds a specified rule latency threshold, the system increments a counter. If the number of consecutive threshold violations reaches a specified number, the system takes the following actions:

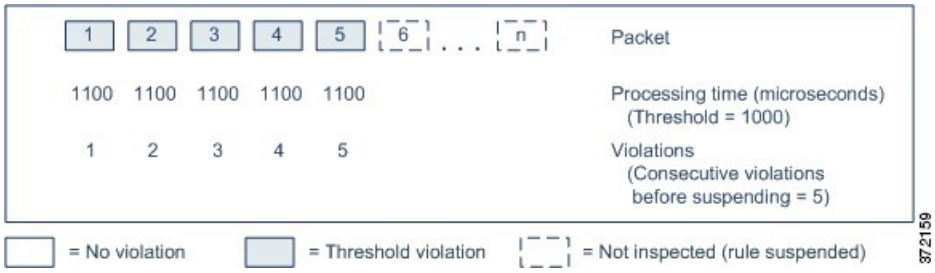
- suspends the rules for the specified period
- triggers an event indicating the rules have been suspended
- re-enables the rules when the suspension expires
- triggers an event indicating the rules have been re-enabled

The system zeroes the counter when the group of rules has been suspended, or when rule violations are not consecutive. Permitting some consecutive violations before suspending rules lets you ignore occasional rule violations that might have negligible impact on performance and focus instead on the more significant impact of rules that repeatedly exceed the rule latency threshold.

The following example shows five consecutive rule processing times that do not result in rule suspension.

In the above example, the time required to process each of the first three packets violates the rule latency threshold of 1000 microseconds, and the violations counter increments with each violation. Processing of the fourth packet does not violate the threshold, and the violations counter resets to zero. The fifth packet violates the threshold and the violations counter restarts at one.

The following example shows five consecutive rule processing times that do result in rule suspension.



In the second example, the time required to process each of the five packets violates the rule latency threshold of 1000 microseconds. The group of rules is suspended because the rule processing time of 1100 microseconds for each packet violates the threshold of 1000 microseconds for the specified five consecutive violations. Any

subsequent packets, represented in the figure as packets 6 through n, are not examined against suspended rules until the suspension expires. If more packets occur after the rules are re-enabled, the violations counter begins again at zero.

Rule latency thresholding has no effect on intrusion events triggered by the rules processing the packet. A rule triggers an event for any intrusion detected in the packet, regardless of whether the rule processing time exceeds the threshold. If the rule detecting the intrusion is a drop rule in an inline deployment, the packet is dropped. When a drop rule detects an intrusion in a packet that results in the rule being suspended, the drop rule triggers an intrusion event, the packet is dropped, and that rule and all related rules are suspended.

Note Packets are not evaluated against suspended rules. A suspended rule that would have triggered an event cannot trigger that event and, for drop rules, cannot drop the packet.

Rule latency thresholding can improve system performance in both passive and inline deployments, and can reduce latency in inline deployments, by suspending rules that take the most time to process packets. Packets are not evaluated again against suspended rules until a configurable time expires, giving the overloaded device time to recover. These performance benefits might occur when, for example:

- hastily written, largely untested rules require an excessive amount of processing time
- a period of poor network performance, such as when someone downloads an extremely large file, causes slow packet inspection

Rule Latency Thresholding Notes

By default, latency-based performance settings for both packet and rule handling are automatically populated by the latest deployed intrusion rule update, and we recommend that you do not change the default.

The information in this topic applies only if you choose to specify custom values.

Rule latency thresholding suspends rules for the time specified by **Suspension Time** when the time rules take to process a packet exceeds **Threshold** for the consecutive number of times specified by **Consecutive Threshold Violations Before Suspending Rule**.

You can enable rule 134:1 to generate an event when rules are suspended, and rule 134:2 to generate an event when suspended rules are enabled.

Table 13: Rule Latency Thresholding Options

Option	Description
Threshold	Specifies the time in microseconds that rules should not exceed when examining a packet.
Consecutive Threshold Violations Before Suspending Rule	Specifies the consecutive number of times rules can take longer than the time set for **Threshold** to inspect packets before rules are suspended.
Suspension Time	Specifies the number of seconds to suspend a group of rules.

Configuring Rule Latency Thresholding

By default, latency-based performance settings for both packet and rule handling are automatically populated by the latest deployed intrusion rule update, and we recommend that you do not change the default.

Procedure

Step 1 In the access control policy editor, click **Advanced**.

In the new UI, select **Advanced Settings** from the drop-down arrow at the end of the packet flow line.

Step 2 Click **Edit** (✎) next to **Latency-Based Performance Settings**.

If **View** (👁) appears instead, settings are inherited from an ancestor policy, or you do not have permission to modify the settings. If the configuration is unlocked, uncheck **Inherit from base policy** to enable editing.

Step 3 Click **Rule Handling** in the **Latency-Based Performance Settings** pop-up window.

By default, **Installed Rule Update** is selected. We recommend using this default.

The values displayed do not reflect the automated settings.

Step 4 If you choose to specify custom values:

- You can configure any of the options in Rule Latency Thresholding Notes, on page 66.
- You must specify custom values in both the packet handling tab and the rule handling tab.

Step 5 Click **OK**.

Step 6 Click **Save** to save the policy.

What to do next

- If you want to generate events, enable latency rules 134:1 and 134:2. For more information, see Intrusion Rule State Options, on page 42.
- Deploy configuration changes.

Intrusion Performance Statistic Logging Configuration

Sample time (seconds) and Minimum number of packets

When the number of seconds specified elapses between performance statistics updates, the system verifies it has analyzed the specified number of packets. If it has, the system updates performance statistics. Otherwise, the system waits until it analyzes the specified number of packets.

Caution Configuring a very low value (for example 1 second) for the sample time can cause a huge impact on the device; the performance statistics logged on the device can cause disk space issues and affect the operation of the device. Hence we recommend you do not configure a very low value.

Troubleshooting Options: Log Session/Protocol Distribution

Support might ask you during a troubleshooting call to log protocol distribution, packet length, and port statistics.

Caution Do not enable **Log Session/Protocol Distribution** unless instructed to by Support. Note that for Classic devices only, enabling or disabling **Log Session/Protocol Distribution** restarts the Snort process when you deploy configuration changes, temporarily interrupting traffic inspection. Whether traffic drops during this interruption or passes without further inspection depends on how the assigned device handles traffic.

Troubleshooting Options: Summary

Support might ask you during a troubleshooting call to configure the system to calculate the performance statistics only when the Snort process is shut down or restarted. To enable this option, you must also enable the **Log Session/Protocol Distribution** troubleshooting option.

Caution Do not enable **Summary** unless instructed to do so by Support.

Configuring Intrusion Performance Statistic Logging

Procedure

Step 1 In the access control policy editor, click **Advanced**, then click **Edit** (✎) next to **Performance Settings**.

In the new UI, select **Advanced Settings** from the drop-down arrow at the end of the packet flow line.

If **View** (👁) appears instead, settings are inherited from an ancestor policy, or you do not have permission to modify the settings. If the configuration is unlocked, uncheck **Inherit from base policy** to enable editing.

Step 2 Click **Performance Statistics** in the pop-up window that appears.

Step 3 Modify the **Sample time** or **Minimum number of packets** as described in Intrusion Performance Statistic Logging Configuration, on page 67.

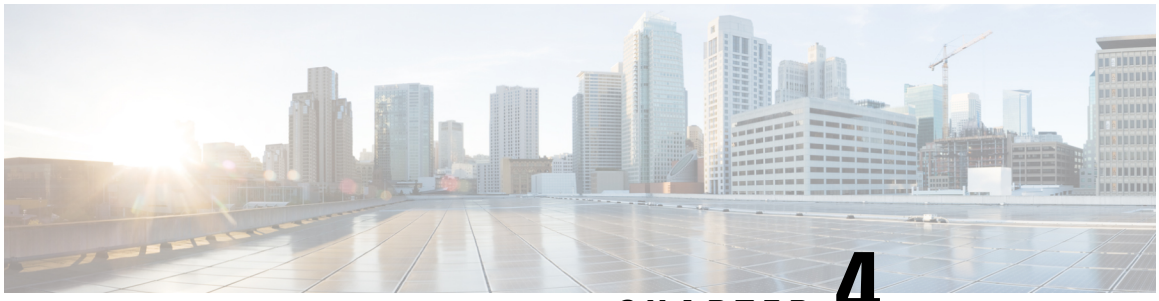
Caution
Configuring a very low value (for example 1 second) for the **Sample time** can cause a huge impact on the device; the performance statistics logged on the device can cause disk space issues and affect the operation of the device. Hence we recommend you do not configure a very low value.

Step 4 Optionally, expand the **Troubleshoot Options** section and modify those options only if asked to do so by Support.

Step 5 Click **OK**.

What to do next

- Deploy configuration changes.



CHAPTER 4

Getting Started with Network Analysis Policies

The following topics describe how to get started with network analysis policies:

Network Analysis Policy Basics

Network analysis policies govern many traffic preprocessing options, and are invoked by advanced settings in your access control policy. Network analysis-related preprocessing occurs after Security Intelligence matching and SSL decryption, but before intrusion or file inspection begins.

By default, the system uses the *Balanced Security and Connectivity* network analysis policy to preprocess all traffic handled by an access control policy. However, you can choose a different default network analysis policy to perform this preprocessing. For your convenience, the system provides a choice of several non-modifiable network analysis policies, which are tuned for a specific balance of security and connectivity by the Talos Intelligence Group. You can also create a custom network analysis policy with custom preprocessing settings.

Tip System-provided intrusion and network analysis policies are similarly named but contain different configurations. For example, the Balanced Security and Connectivity network analysis policy and the Balanced Security and Connectivity intrusion policy work together and can both be updated in intrusion rule updates. However, the network analysis policy governs mostly preprocessing options, whereas the intrusion policy governs mostly intrusion rules. Network analysis and intrusion policies work together to examine your traffic.

You can also tailor traffic preprocessing options to specific security zones, networks, and VLANs by creating multiple custom network analysis policies, then assigning them to preprocess different traffic.

License Requirements for Network Analysis Policies

Threat Defense License

IPS

Requirements and Prerequisites for Network Analysis Policies

Model support

Any.

Supported domains

Any

User roles

- Admin
- Intrusion Admin

Managing Network Analysis Policies

Procedure

Step 1 Choose **Policies** > **Access Control heading** > **Access Control**, and then click **Network Analysis Policy** or **Policies** > **Access Control heading** > **Intrusion**, and then click **Network Analysis Policies**.

Note
If your custom user role limits access to the first path listed here, use the second path to access the policy.

Step 2 Manage your network analysis policy:

- Compare—Click **Compare Policies**; see Comparing policies.
- Create — If you want to create a new network analysis policy, click **Create Policy**.

  Two versions of the network analysis policy are created, a **Snort 2 Version** and a **Snort 3 Version**.
- Delete — If you want to delete a network analysis policy, click **Delete** (), then confirm that you want to delete the policy. You cannot delete a network analysis policy if an access control policy references it.

  If the controls are dimmed, the configuration belongs to an ancestor domain, or you do not have permission to modify the configuration.
- Deploy—Choose **Deploy** > **Deploy**; see Deploy Configuration Changes.
- Edit — If you want to edit an existing network analysis policy, click **Edit** () and proceed as described in Network Analysis Policy Settings and Cached Changes, on page 75.

  If **View** () appears instead, the configuration belongs to an ancestor domain, or you do not have permission to modify the configuration.

- Report—Click **Report** ( ) ; see Generate Current Policy Reports.

Create a Network Analysis Policy

All the existing network analysis policies are available in Firewall Management Center with their corresponding Snort 2 and Snort 3 versions. When you create a new network analysis policy, it is created with both the Snort 2 version and the Snort 3 version.

Procedure

Step 1 Go to **Policies** > **Access Control heading** > **Intrusion** > **Network Analysis Policies**.

Step 2 Click **Create Policy**.

Step 3 Enter the **Name** and **Description**.

Step 4 Select a **Base Policy** and click **Save**.

The new network analysis policy is created with its corresponding **Snort 2 Version** and **Snort 3 Version**.

Modify the Network Analysis Policy

You can modify the network analysis policy to change its name, description, or the base policy.

Procedure

Step 1 Go to **Policies** > **Access Control heading** > **Intrusion** > **Network Analysis Policies**.

Step 2 Click **Edit** to change the name, description, inspection mode, or the base policy.

Attention
Detection mode deprecation: From management center 7.4.0 onwards, for a network analysis policy (NAP), the **Detection** inspection mode is deprecated and will be removed in an upcoming release.

The **Detection** mode was intended to be used as a test mode so that you can enable inspections and see how they behave in your network before setting it to drop traffic, that is, to show traffic that would be dropped.

This behavior is improved where all inspector drops are controlled by the rule state, and you can set each one to generate events. This is done to test them before configuring the rule state to drop traffic. As we now have granular control over traffic drops in Snort 3, the **Detection** mode only adds more complexity to the product and is not needed, so the detection mode is deprecated.

If you change a NAP in **Detection** mode to **Prevention**, the NAP that processes the traffic of intrusion events and have the result "will be dropped" will now be "dropped" and the corresponding traffic will drop the traffic from these events. This is applicable for rules whose GIDs are not 1 or 3. GIDs 1 and 3 are text/compiled rules (typically provided by Talos or from your custom/imported rules) and all other GIDs are inspections for anomalies. These are more uncommon rules to trigger in a network. Changing to **Prevention** mode is unlikely to have any impact on the traffic. You need to just disable the intrusion rule that is applicable for the dropped traffic and set it to just generate or disable.

We recommend you choose **Prevention** as the inspection mode, but if you choose **Prevention**, you cannot revert to **Detection** mode.

Note
If you edit the network analysis policy name, description, base policy, and inspection mode, the edits are applied to both the Snort 2 and Snort 3 versions. If you want to change the inspection mode for a specific version, then you can do that from within the network analysis policy page for that respective version.

Step 3 Click **Save**.

Custom Network Analysis Policy Creation for Snort 2

When you create a new network analysis policy you must give it a unique name, specify a base policy, and choose an *inline mode*.

The base policy defines the network analysis policy's default settings. Modifying a setting in the new policy overrides—but does not change—the settings in the base policy. You can use either a system-provided or custom policy as your base policy.

The network analysis policy's inline mode allows preprocessors to modify (normalize) and drop traffic to minimize the chances of attackers evading detection. Note that in passive deployments, the system cannot affect traffic flow regardless of the inline mode.

Creating a Custom Network Analysis Policy

In a multidomain deployment, the system displays policies created in the current domain, which you can edit. It also displays policies created in ancestor domains, which you cannot edit. To view and edit policies created in a lower domain, switch to that domain.

Procedure

Step 1 Choose **Policies** > **Access Control heading** > **Access Control**, and then click **Network Analysis Policy** or **Policies** > **Access Control heading** > **Intrusion**, and then **Network Analysis Policies**.

Note
If your custom user role limits access to the first path listed here, use the second path to access the policy.

Step 2 Click **Create Policy**. If you have unsaved changes in another policy, click **Cancel** when prompted to return to the **Network Analysis Policy** page.

Step 3 Enter a unique **Name**.

In a multidomain deployment, policy names must be unique within the domain hierarchy. The system may identify a conflict with the name of a policy you cannot view in your current domain.

Step 4 Optionally, enter a **Description**.

Step 5 Choose the initial **Base Policy**. You can use either a system-provided or custom policy as your base policy.

Attention

While configuring your custom NAP, if you select **Maximum Detection** as the **Base Policy**, you might experience performance degrade. It is recommended to review and test this setting before deploying to production environment.

Step 6 If you want to allow preprocessors to affect traffic in an inline deployment, enable **Inline Mode**.

Step 7 To create the policy:

- Click **Create Policy** to create the new policy and return to the **Network Analysis Policy** page. The new policy has the same settings as its base policy.
- Click **Create and Edit Policy** to create the policy and open it for editing in the advanced network analysis policy editor.

Network Analysis Policy Management for Snort 2

On the Network Analysis Policy page (or **Policies** > **Access Control heading** > **Access Control**, and then click **Network Analysis Policy** or **Policies** > **Access Control heading** > **Intrusion**, and then click **Network Analysis Policies**) you can view your current custom network analysis policies, along with the following information:

- the time and date the policy was last modified (in local time) and the user who modified it
- whether the **Inline Mode** setting is enabled, which allows preprocessors to affect traffic
- which access control policies and devices are using the network analysis policy to preprocess traffic
- whether a policy has unsaved changes, as well as information about who (if anyone) is currently editing the policy

In addition to custom policies that you create, the system provides two custom policies: Initial Inline Policy and Initial Passive Policy. These two network analysis policies use the Balanced Security and Connectivity network analysis policy as their base. The only difference between them is their inline mode, which allows preprocessors to affect traffic in the inline policy and disables it in the passive policy. You can edit and use these system-provided custom policies.

Note that you can create and edit network analysis as well as intrusion policies if your system user account's role is restricted to Intrusion Policy or Modify Intrusion Policy.

Network Analysis Policy Settings and Cached Changes

When you create a new network analysis policy, it has the same settings as its base policy.

When tailoring a network analysis policy, especially when disabling preprocessors, keep in mind that some preprocessors and intrusion rules require that traffic first be decoded or preprocessed in a certain way. If you disable a required preprocessor, the system automatically uses it with its current settings, although the preprocessor remains disabled in the network analysis policy web interface.

Note Because preprocessing and intrusion inspection are so closely related, the network analysis and intrusion policies examining a single packet **must** complement each other. Tailoring preprocessing, especially using multiple custom network analysis policies, is an **advanced** task.

The system caches one network analysis policy per user. While editing a network analysis policy, if you select any menu or other path to another page, your changes stay in the system cache even if you leave the page.

Editing Network Analysis Policies

In a multidomain deployment, the system displays policies created in the current domain, which you can edit. It also displays policies created in ancestor domains, which you cannot edit. To view and edit policies created in a lower domain, switch to that domain.

Procedure

Step 1 Choose **Policies** > **Access Control heading** > **Access Control**, and then click **Network Analysis Policy** or **Policies** > **Access Control heading** > **Intrusion**, and then **Network Analysis Policies**.

Note
If your custom user role limits access to the first path listed here, use the second path to access the policy.

Step 2 Click **Snort 2 Version** next to the policy you want to edit.

Step 3 Click **Edit** (✎) next to the network analysis policy you want to configure.

If **View** (👁) appears instead, the configuration belongs to an ancestor domain, or you do not have permission to modify the configuration.

Step 4 Edit your network analysis policy:

- Change the base policy — If you want to change the base policy, choose a base policy from the **Base Policy** drop-down list on the Policy Information page.
- Manage policy layers — If you want to manage policy layers, click **Policy Layers** in the navigation panel.
- Modify a preprocessor — If you want to enable, disable, or edit the settings for a preprocessor, click **Settings** in the navigation panel.
- Modify traffic — If you want to allow preprocessors to modify or drop traffic, check the **Inline Mode** check box on the Policy Information page.
- View settings — If you want to view the settings in the base policy, click **Manage Base Policy** on the Policy Information page.

Step 5 To save changes you made in this policy since the last policy commit, choose **Policy Information**, then click **Commit Changes**. If you leave the policy without committing changes, changes since the last commit are discarded if you edit a different policy.

What to do next

- If you want a preprocessor to generate events and, in an inline deployment, drop offending packets, enable rules for the preprocessor. For more information, see Setting Intrusion Rule States, on page 42.
- Deploy configuration changes.

Preprocessor Configuration in a Network Analysis Policy for Snort 2

Preprocessors prepare traffic to be further inspected by normalizing traffic and identifying protocol anomalies. Preprocessors can generate preprocessor events when packets trigger preprocessor options that you configure. The base policy for your network analysis policy determines which preprocessors are enabled by default and the default configuration for each.

Note In most cases, preprocessors require specific expertise to configure and typically require little or no modification. Tailoring preprocessing, especially using multiple custom network analysis policies, is an **advanced** task. Because preprocessing and intrusion inspection are so closely related, the network analysis and intrusion policies examining a single packet **must** complement each other.

Modifying a preprocessor configuration requires an understanding of the configuration and its potential impact on your network.

Note that some advanced transport and network preprocessor settings apply globally to all networks, zones, and VLANs where you deploy your access control policy. You configure these advanced settings in an access control policy rather than in a network analysis policy.

Note also that you configure the sensitive data preprocessor, which detects sensitive data such as credit card numbers and Social Security numbers in ASCII text, in intrusion policies.

Preprocessor Traffic Modification in Inline Deployments

In an inline deployment (that is, where relevant configurations are deployed to devices using routed, switched, or transparent interfaces, or inline interface pairs), some preprocessors can modify and block traffic. For example:

- The inline normalization preprocessor normalizes packets to prepare them for analysis by other preprocessors and the intrusion rules engine. You can also use the preprocessor's **Allow These TCP Options** and **Block Unresolvable TCP Header Anomalies** options to block certain packets.
- The system can drop packets with invalid checksums.
- The system can drop packets matching rate-based attack prevention settings.

For a preprocessor configured in the network analysis policy to affect traffic, you must enable and correctly configure the preprocessor, as well as correctly deploy managed devices inline. Finally, you must enable the network analysis policy's **Inline Mode** setting.

Preprocessor Configuration in a Network Analysis Policy Notes

When you select **Settings** in the navigation panel of a network analysis policy, the policy lists its preprocessors by type. On the Settings page, you can enable or disable preprocessors in your network analysis policy, as well as access preprocessor configuration pages.

A preprocessor must be enabled for you to configure it. When you enable a preprocessor, a sublink to the configuration page for the preprocessor appears beneath the **Settings** link in the navigation panel, and an **Edit** link to the configuration page appears next to the preprocessor on the Settings page.

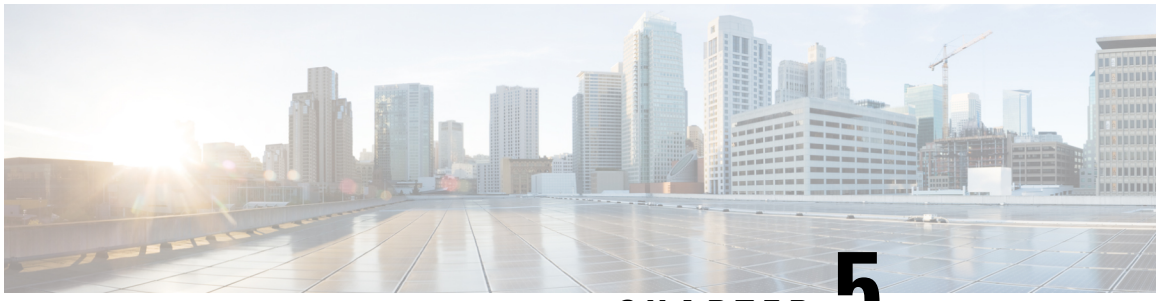
Tip To revert a preprocessor's configuration to the settings in the base policy, click **Revert to Defaults** on a preprocessor configuration page. When prompted, confirm that you want to revert.

When you disable a preprocessor, the sublink and **Edit** link no longer appear, but your configurations are retained. Note that to perform their particular analysis, many preprocessors and intrusion rules require that traffic first be decoded or preprocessed in a certain way. If you disable a required preprocessor, the system automatically uses it with its current settings, although the preprocessor remains disabled in the network analysis policy web interface.

If you want to assess how your configuration would function in an inline deployment without actually modifying traffic, you can disable inline mode. In passive deployments or inline deployments in tap mode, the system cannot affect traffic regardless of the inline mode setting.

Note Disabling inline mode can affect intrusion event performance statistics graphs. With inline mode enabled in an inline deployment, the Intrusion Event Performance page (**Overview** > **Summary** > **Intrusion Event Performance**) displays graphs that represent normalized and blocked packets. If you disable inline mode, or in a passive deployment, many of the graphs display data about the traffic the system would have normalized or dropped.

Note In an inline deployment, we recommend that you enable inline mode and configure the inline normalization preprocessor with the **Normalize TCP Payload** option enabled. In a passive deployment, we recommend that you use adaptive profile updates.



CHAPTER 5

Advanced Access Control Settings for Network Analysis and Intrusion Policies

The following topics describe how to configure advanced settings for network analysis and intrusion policies:

About Advanced Access Control Settings for Network Analysis and Intrusion Policies

Many of the advanced settings in an access control policy govern intrusion detection and prevention configurations that require specific expertise to configure. Advanced settings typically require little or no modification and are not common to every deployment.

Requirements and Prerequisites for Advanced Access Control Settings for Network Analysis and Intrusion Policies

Model support

Any.

Supported domains

Any

User roles

- Admin
- Access Admin

- Network Admin

Inspection of Packets That Pass Before Traffic Is Identified

For some features, including URL filtering, application detection, rate limiting, and Intelligent Application Bypass, a few packets must pass in order for the connection to be established, and to enable the system to identify the traffic and determine which access control rule (if any) will handle that traffic.

You must explicitly configure your access control policy to inspect these packets, prevent them from reaching their destination, and generate any events.

As soon as the system identifies the access control rule or default action that should handle the connection, the remaining packets in the connection are handled and inspected accordingly.

Best Practices for Handling Packets That Pass Before Traffic Identification

- The default action specified for an access control policy is NOT applied to these packets.
- Instead, use the following guidelines to choose a value for the **Intrusion Policy used before Access Control rule is determined** setting in the Advanced settings of the access control policy.
  - You can choose a system-created or custom intrusion policy. For example, you can choose **Balanced Security and Connectivity**.
  - For performance reasons, unless you have good reason to do otherwise, this setting should match the default action set for your access control policy.
  - If your system does not perform intrusion inspection (for example, in a discovery-only deployment), select **No Rules Active**. The system will not inspect these initial packets, and they will be allowed to pass.
  - By default, this setting uses the default variable set. Ensure that this is suitable for your purposes. For information, see Variable Set.
  - The network analysis policy associated with the first matching network analysis rule preprocesses traffic for the policy you select. If there are no network analysis rules, or none match, the default network analysis policy is used.

Specify a Policy to Handle Packets That Pass Before Traffic Identification

Note This setting is sometimes referred to as the *default intrusion policy*. (This is distinct from the default action for an access control policy.)

Caution Changing the total number of intrusion policies used by an access control policy restarts the Snort process when you deploy configuration changes, temporarily interrupting traffic inspection. Whether traffic drops during this interruption or passes without further inspection depends on how the assigned device handles traffic. You change the the total number of intrusion policies by adding an intrusion policy that is not currently used, or by removing the last instance of an intrusion policy. You can use an intrusion policy in an access control rule, as the default action, or as the default intrusion policy.

Before you begin

Review best practices for these settings. See Best Practices for Handling Packets That Pass Before Traffic Identification, on page 80.

Procedure

Step 1 In the access control policy editor, click **Advanced**, then click **Edit** (✎) next to the **Network Analysis** and **Intrusion Policies** section.

If **View** (👁) appears instead, settings are inherited from an ancestor policy, or you do not have permission to modify the settings. If the configuration is unlocked, uncheck **Inherit from base policy** to enable editing.

Step 2 Select an intrusion policy from the **Intrusion Policy used before Access Control rule is determined** drop-down list.

If you choose a user-created policy, you can click **Edit** (✎) to edit the policy in a new window. You cannot edit system-provided policies.

Step 3 Optionally, select a different variable set from the **Intrusion Policy Variable Set** drop-down list. You can also select **Edit** (✎) next to the variable set to create and edit variable sets. If you do not change the variable set, the system uses a default set.

Step 4 Click **OK**.

Step 5 Click **Save** to save the policy.

What to do next

- Deploy configuration changes.

Related Topics

Variable Set

Advanced Settings for Network Analysis Policies

Network analysis policies govern how traffic is decoded and preprocessed so that it can be further evaluated, especially for anomalous traffic that might signal an intrusion attempt. This traffic preprocessing occurs after Security Intelligence matching and traffic decryption, but before intrusion policies inspect packets in detail. By default, the system-provided Balanced Security and Connectivity network analysis policy is the default network analysis policy.

Tip The system-provided Balanced Security and Connectivity network analysis policy and the Balanced Security and Connectivity intrusion policy work together and can both be updated in intrusion rule updates. However, the network analysis policy governs mostly preprocessing options, whereas the intrusion policy governs mostly intrusion rules.

A simple way to tune preprocessing is to create and use a custom network analysis policy as the default. For advanced users with complex deployments, you can create multiple network analysis policies, each tailored to preprocess traffic differently. Then, you can configure the system to use those policies to govern the preprocessing of traffic using different security zones, networks, or VLANs.

To accomplish this, you add custom *network analysis rules* to your access control policy. A network analysis rule is simply a set of configurations and conditions that specifies how you preprocess traffic that matches those qualifications. You create and edit network analysis rules in the advanced options in an existing access control policy. Each rule belongs to only one policy.

Each rule has:

- a set of rule conditions that identifies the specific traffic you want to preprocess
- an associated network analysis policy that you want to use to preprocess traffic that meets all the rules' conditions

When it is time for the system to preprocess traffic, it matches packets to network analysis rules in top-down order by rule number. Traffic that does not match any network analysis rules is preprocessed by the default network analysis policy.

Setting the Default Network Analysis Policy

You can choose a system- or user-created policy.

Note If you disable a preprocessor but the system needs to evaluate preprocessed packets against an enabled intrusion or preprocessor rule, the system automatically enables and uses the preprocessor although it remains disabled in the network analysis policy web interface. Tailoring preprocessing, especially using multiple custom network analysis policies, is an **advanced** task. Because preprocessing and intrusion inspection are so closely related, you **must** be careful that you allow the network analysis and intrusion policies examining a single packet to complement each other.

Procedure

Step 1 In the access control policy editor, click **Advanced**, then click **Edit** (✎) next to the Network Analysis and Intrusion Policies section.

If **View** (👁) appears instead, settings are inherited from an ancestor policy, or you do not have permission to modify the settings. If the configuration is unlocked, uncheck **Inherit from base policy** to enable editing.

Step 2 From the **Default Network Analysis Policy** drop-down list, select a default network analysis policy.

If you choose a user-created policy, you can click **Edit** (✎) to edit the policy in a new window. You cannot edit system-provided policies.

Caution
Changing the total number of network analysis policies used by an access control policy restarts the Snort process when you deploy configuration changes, temporarily interrupting traffic inspection. Whether traffic drops during this interruption or passes without further inspection depends on how the assigned device handles traffic.You change the total number of network analysis policies by adding a policy that is not currently used, or by removing the last instance of a network analysis policy. You can use a network analysis policy with network analysis rules or as the default network analysis policy.

Step 3 Click **OK**.

Step 4 Click **Save** to save the policy.

What to do next

- Deploy configuration changes.

Related Topics

Limitations of Custom Policies, on page 12

Network Analysis Rules

Within your access control policy's advanced settings, you can use network analysis rules to tailor preprocessing configurations to network traffic.

Network analysis rules are numbered, starting at 1. When it is time for the system to preprocess traffic, it matches packets to network analysis rules in top-down order by ascending rule number, and preprocesses traffic according to the first rule where all the rule's conditions match.

You can add zone, network, and VLAN tag conditions to a rule. If you do not configure a particular condition for a rule, the system does not match traffic based on that criterion. For example, a rule with a network condition but no zone condition evaluates traffic based on its source or destination IP address, regardless of its ingress or egress interface. Traffic that does not match any network analysis rules is preprocessed by the default network analysis policy.

Network Analysis Policy Rule Conditions

Rule conditions enable you to fine-tune your network analysis policy to target the users and networks you want to control. See one of the following sections for more information.

Related Topics

Security zone rule conditions, on page 83
Network rule conditions, on page 84
VLAN tags rule conditions, on page 84

Security zone rule conditions

Security zones segment your network to help you manage, classify, and decrypt traffic flow by grouping interfaces across multiple devices.

Security zones control or decrypt traffic by its source and destination security zones. If you add both source and destination zones to a zone condition, matching traffic must originate from an interface in one of the source zones and leave through an interface in one of the destination zones.

Just as all interfaces in a zone must be of the same type (all inline, passive, switched, or routed), all zones used in a zone condition must be of the same type. Because devices deployed passively do not transmit traffic, you cannot use a zone with passive interfaces as a destination zone.

Minimize the number of matching criteria whenever possible, especially those for security zones, network objects, and port objects. When you specify multiple criteria, the system must match against *every* combination of the contents of the criteria you specify.

Tip Constraining rules by zone is one of the best ways to improve system performance. If a rule does not apply to traffic through any of device's interfaces, that rule does not affect that device's performance.

Security zone conditions and multitenancy

In a multidomain deployment, a zone created in an ancestor domain can contain interfaces that reside on devices in different domains. When you configure a zone condition in an descendant domain, your configurations apply to only the interfaces you can see.

Network rule conditions

Networks control or decrypt traffic by its source and destination IP address, using inner headers. Tunnel rules, which use outer headers, have tunnel endpoint conditions instead of network conditions.

You can use predefined objects to build network conditions, or manually specify individual IP addresses or address blocks.

Minimize the number of matching criteria whenever possible, especially those for security zones, network objects, and port objects. When you specify multiple criteria, the system must match against *every* combination of the contents of the criteria you specify.

Note You *cannot* use FDQN network objects in identity rules.

VLAN tags rule conditions

Note VLAN tags in access rules only apply to inline sets. Access rules with VLAN tags do not match traffic on firewall interfaces.

VLAN rule conditions control VLAN-tagged traffic, including Q-in-Q (stacked VLAN) traffic. The system uses the innermost VLAN tag to filter VLAN traffic, with the exception of the prefilter policy, which uses the outermost VLAN tag in its rules.

Note the following Q-in-Q support:

- Firewall Threat Defense on Firepower 4100/9300—Does not support Q-in-Q (supports only one VLAN tag).

- Firewall Threat Defense on all other models:
  - Inline sets and passive interfaces—Supports Q-in-Q, up to 2 VLAN tags.
  - Firewall interfaces—Does not support Q-in-Q (supports only one VLAN tag).

You can use predefined objects to build VLAN conditions, or manually enter any VLAN tag from 1 to 4094. Use a hyphen to specify a range of VLAN tags.

In a cluster, if you encounter problems with VLAN matching, edit the access control policy advanced options, Transport/Network Preprocessor Settings, and select the **Ignore the VLAN header when tracking connections** option.

Configuring Network Analysis Rules

Procedure

Step 1 In the access control policy editor, click **Advanced**, then click **Edit** (✎) next to the Network Analysis and Intrusion Policies section.

If **View** (👁) appears instead, settings are inherited from an ancestor policy, or you do not have permission to modify the settings.If the configuration is unlocked, uncheck **Inherit from base policy** to enable editing.

Tip
Click **Network Analysis Policy List** to view and edit existing custom network analysis policies.

Step 2 Next to **Network Analysis Rules**, click the statement that indicates how many custom rules you have.

Step 3 Click **Add Rule**.

Step 4 Configure the rule's conditions by clicking the conditions you want to add.

Step 5 Click **Network Analysis** and choose the **Network Analysis Policy** you want to use to preprocess the traffic matching this rule.

Click **Edit** (✎) to edit a custom policy in a new window. You cannot edit system-provided policies.

Caution
Changing the total number of network analysis policies used by an access control policy restarts the Snort process when you deploy configuration changes, temporarily interrupting traffic inspection. Whether traffic drops during this interruption or passes without further inspection depends on how the assigned device handles traffic.You change the total number of network analysis policies by adding a policy that is not currently used, or by removing the last instance of a network analysis policy. You can use a network analysis policy with network analysis rules or as the default network analysis policy.

Step 6 Click **Add**.

What to do next

- Deploy configuration changes.

Managing Network Analysis Rules

A network analysis rule is simply a set of configurations and conditions that specifies how you preprocess traffic that matches those qualifications. You create and edit network analysis rules in the advanced options in an existing access control policy. Each rule belongs to only one policy.

Procedure

Step 1 In the access control policy editor, click **Advanced**, then click **Edit** (✎) next to the Intrusion and Network Analysis Policies section.

If **View** (👁) appears instead, settings are inherited from an ancestor policy, or you do not have permission to modify the settings.If the configuration is unlocked, uncheck **Inherit from base policy** to enable editing.

Step 2 Next to **Network Analysis Rules**, click the statement that indicates how many custom rules you have.

Step 3 Edit your custom rules. You have the following options:

- To edit a rule's conditions, or change the network analysis policy invoked by the rule, click **Edit** (✎) next to the rule.
- To change a rule's order of evaluation, click and drag the rule to the correct location. To select multiple rules, use the Shift and Ctrl keys.
- To delete a rule, click **Delete** (🗑) next to the rule.

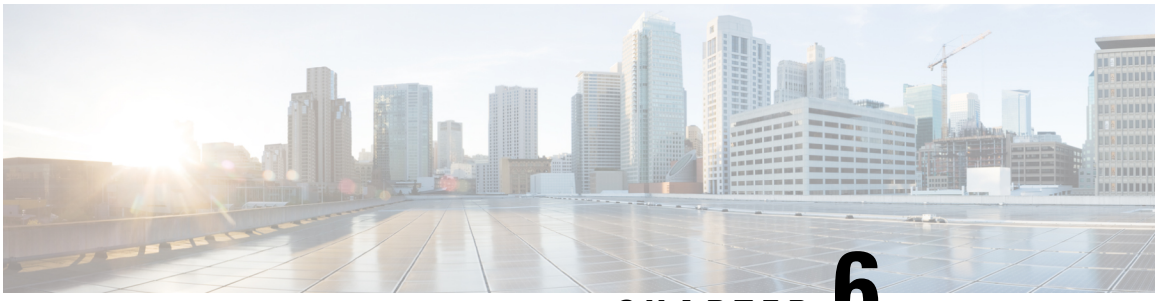
Tip
Right-clicking a rule displays a context menu that allows you to cut, copy, paste, edit, delete, and add new network analysis rules.

Step 4 Click **OK**.

Step 5 Click **Save** to save the policy.

What to do next

- Deploy configuration changes.



CHAPTER 6

Custom Intrusion Rules

The following topics describe how to use the intrusion rules editor:

Custom Intrusion Rules Overview

An *intrusion rule* is a set of keywords and arguments that the system uses to detect attempts to exploit vulnerabilities on your network. As the system analyzes network traffic, it compares packets against the conditions specified in each rule. If the packet data matches all the conditions specified in a rule, the rule triggers. If a rule is an *alert rule*, it generates an intrusion event. If it is a *pass rule*, it ignores the traffic. For a *drop* rule in an inline deployment, the system drops the packet and generates an event. You can view and evaluate intrusion events from the Secure Firewall Management Center web interface.

The system provides two types of intrusion rules: shared object rules and standard text rules. The Talos Intelligence Group can use shared object rules to detect attacks against vulnerabilities in ways that traditional standard text rules cannot. You cannot create shared object rules. When you write your own intrusion rule, you create a standard text rule.

You can write custom standard text rules to tune the types of events you are likely to see. Note that while this documentation sometimes discusses rules targeted to detect specific exploits, the most successful rules target traffic that may attempt to exploit known vulnerabilities rather than specific known exploits. By writing rules and specifying the rule's event message, you can more easily identify traffic that indicates attacks and policy evasions.

When you enable a custom standard text rule in a custom intrusion policy, keep in mind that some rule keywords and arguments require that traffic first be decoded or preprocessed in a certain way. This chapter explains the options you must configure in your network analysis policy, which governs preprocessing. Note that if you disable a required preprocessor, the system automatically uses it with its current settings, although the preprocessor remains disabled in the network analysis policy web interface.

Caution Make sure you use a controlled network environment to test any intrusion rules that you write before you use the rules in a production environment. Poorly written intrusion rules may seriously affect the performance of the system.

Note You can create custom intrusion rules using Snort. However, support for tuning and troubleshooting these rules is not available currently.

License Requirements for the Intrusion Rule Editor

Threat Defense License

IPS

Requirements and Prerequisites for the Intrusion Rule Editor

Model support

Any.

Supported domains

Any

User roles

- Admin
- Intrusion Admin

Rule Anatomy

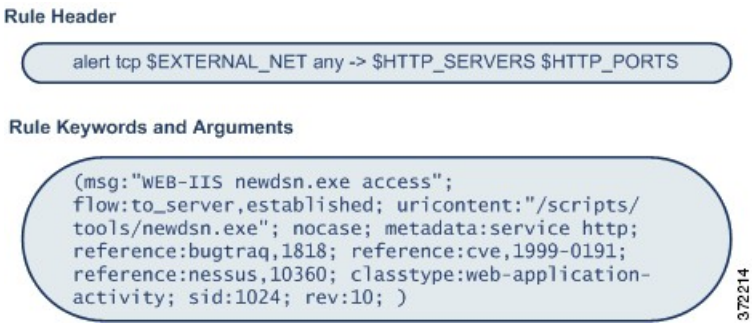
All standard text rules contain two logical sections: the rule header and the rule options. The rule header contains:

- the rule's action or type
- the protocol
- the source and destination IP addresses and netmasks
- direction indicators showing the flow of traffic from source to destination
- the source and destination ports

The rule options section contains:

- event messages
- keywords and their parameters and arguments
- patterns that a packet's payload must match to trigger the rule
- specifications of which parts of the packet the rules engine should inspect

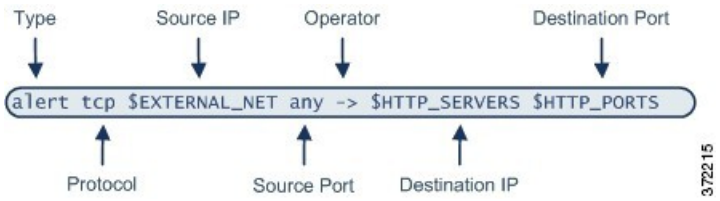
The following diagram illustrates the parts of a rule:



Note that the options section of a rule is the section enclosed in parentheses. The intrusion rules editor provides an easy-to-use interface to help you build standard text rules.

The Intrusion Rule Header

Every standard text rule and shared object rule has a rule header containing parameters and arguments. The following illustrates parts of a rule header:



The following table describes each part of the rule header shown above.

Table 14: Rule Header Values

Rule Header Component	Example Value	This Value...
Action	`alert`	Generates an intrusion event when triggered.
Protocol	`tcp`	Tests TCP traffic only.
Source IP Address	`$EXTERNAL_NET`	Tests traffic coming from any host that is not on your internal network.
Source Ports	`any`	Tests traffic coming from any port on the originating host.

Rule Header Component	Example Value	This Value...
Operator	`->`	Tests external traffic (destined for the web servers on your network).
Destination IP Address	`$HTTP_SERVERS`	Tests traffic to be delivered to any host specified as a web server on your internal network.
Destination Ports	`$HTTP_PORTS`	Tests traffic delivered to an HTTP port on your internal network.

Note The previous example uses default variables, as do most intrusion rules.

Intrusion Rule Header Action

Each rule header includes a parameter that specifies the action the system takes when a packet triggers a rule. Rules with the action set to *alert* generate an intrusion event against the packet that triggered the rule and log the details of that packet. Rules with the action set to *pass* do not generate an event against, or log the details of, the packet that triggered the rule.

Note In an inline deployment, rules with the rule state set to *Drop and Generate Events* generate an intrusion event against the packet that triggered the rule. Also, if you apply a drop rule in a passive deployment, the rule acts as an alert rule.

By default, pass rules override alert rules. You can create pass rules to prevent packets that meet criteria defined in the pass rule from triggering the alert rule in specific situations, rather than disabling the alert rule. For example, you might want a rule that looks for attempts to log into an FTP server as the user "anonymous" to remain active. However, if your network has one or more legitimate anonymous FTP servers, you could write and activate a pass rule that specifies that, for those specific servers, anonymous users do not trigger the original rule.

Within the intrusion rules editor, you select the rule type from the **Action** list.

Intrusion Rule Header Protocol

In each rule header, you must specify the protocol of the traffic the rule inspects. You can specify the following network protocols for analysis:

- ICMP (Internet Control Message Protocol)
- IP (Internet Protocol)

Note The system ignores port definitions in an intrusion rule header when the protocol is set to `ip`.

- TCP (Transmission Control Protocol)

- UDP (User Datagram Protocol)

Use **IP** as the protocol type to examine all protocols assigned by IANA, including TCP, UDP, ICMP, IGMP, and many more.

Note You cannot currently write rules that match patterns in the next header (for example, the TCP header) in an IP payload. Instead, content matches begin with the last decoded protocol. As a workaround, you can match patterns in TCP headers by using rule options.

Within the Intrusion Rules editor, you select the protocol type from the **Protocol** list.

Intrusion Rule Header Direction

Within the rule header, you can specify the direction that the packet must travel for the rule to inspect it. The following table describes these options.

Table 15: Directional Options in Rule Headers

Use...	To Test...
Directional	only traffic from the specified source IP address to the specified destination IP address
Bidirectional	all traffic traveling between the specified source and destination IP addresses

Intrusion Rule Header Source and Destination IP Addresses

Restricting packet inspection to the packets originating from specific IP addresses or destined to a specific IP address reduces the amount of packet inspection the system must perform. This also reduces false positives by making the rule more specific and removing the possibility of the rule triggering against packets whose source and destination IP addresses do not indicate suspicious behavior.

Tip The system recognizes only IP addresses and does not accept host names for source or destination IP addresses.

Within the intrusion rules editor, you specify source and destination IP addresses in the **Source IPs** and **Destination IPs** fields.

When writing standard text rules, you can specify IPv4 and IPv6 addresses in a variety of ways, depending on your needs. You can specify a single IP address, `any`, IP address lists, CIDR notation, prefix lengths, or a network variable. Additionally, you can indicate that you want to exclude a specific IP address or set of IP addresses. When specifying IPv6 addresses, you can use any addressing convention defined in RFC 4291.

IP Address Syntax in Intrusion Rules

The following table summarizes the various ways you can specify source and destination IP addresses.

Table 16: Source/Destination IP Address Syntax

To Specify...	Use...	Example
any IP address	`any`	`any`

To Specify...	Use...	Example
a specific IP address	the IP address Note that you would not mix IPv4 and IPv6 source and destination addresses in the same rule.	`192.168.1.1` `2001:db8::abcd`
a list of IP addresses	brackets (`[]`) to enclose the IP addresses and commas to separate them	`[192.168.1.1,192.168.1.15]` `[2001:db8::b3ff, 2001:db8::0202]`
a block of IP addresses	IPv4 CIDR block or IPv6 address prefix notation	`192.168.1.0/24` `2001:db8::/32`
anything except a specific IP address or set of addresses	the `!` character before the IP address or addresses you want to negate	`!192.168.1.15` `!2001:db8::0202:b3ff:fe1e`
anything in a block of IP addresses except one or more specific IP addresses	a block of addresses followed by a list of negated addresses or blocks	`[10.0.0/8, !10.2.3.4, !10.1.0.0/16]` `[2001:db8::/32, !2001:db8::8329, !2001:db8::0202]`
IP addresses defined by a network variable	the variable name, in uppercase letters, preceded by `$` Note that preprocessor rules can trigger events regardless of the hosts defined by network variables used in intrusion rules.	`$HOME_NET`
all IP addresses except addresses defined by an IP address variable	the variable name, in uppercase letters, preceded by `!$`	`!$HOME_NET`

The following descritptions provide additional information on some of the IP address entry methods.

Any IP Address

You can specify the word `any` as a rule source or destination IP address to indicate any IPv4 or IPv6 address.

For example, the following rule uses the argument **`any`** in the **Source IPs** and **Destination IPs** fields and evaluates packets with any IPv4 or IPv6 source or destination address:

```
alert tcp any any -> any any
```

You can also specify `::` to indicate any IPv6 address.

Multiple IP Addresses

You can list individual IP addresses by separating the IP addresses with commas and, optionally, by surrounding non-negated lists with brackets, as shown in the following example:

```
[192.168.1.100,192.168.1.103,192.168.1.105]
```

You can list IPv4 and IPv6 addresses alone or in any combination, as shown in the following example:

```
[192.168.1.100,2001:db8::1234,192.168.1.105]
```

Note that surrounding an IP address list with brackets, which was required in earlier software releases, is not required. Note also that, optionally, you can enter lists with a space before or after each comma.

Note You must surround negated lists with brackets.

You can also use IPv4 Classless Inter-Domain Routing (CIDR) notation or IPv6 prefix lengths to specify address blocks. For example:

- 192.168.1.0/24 specifies the IPv4 addresses in the 192.168.1.0 network with a subnet mask of 255.255.255.0, that is, 192.168.1.0 through 192.168.1.255.
- 2001:db8::/32 specifies the IPv6 addresses in the 2001:db8:: network with a prefix length of 32 bits, that is, 2001:db8:: through 2001:db8:ffff:ffff:ffff:ffff:ffff:ffff.

Tip If you need to specify a block of IP addresses but cannot express it using CIDR or prefix length notation alone, you can use CIDR blocks and prefix lengths in an IP address list.

IP Addresses Negation

You can use an exclamation point (`!`) to negate a specified IP address. That is, you can match any IP address with the exception of the specified IP address or addresses. For example, `!192.168.1.1` specifies any IP address other than 192.168.1.1, and `!2001:db8:ca2e::fa4c` specifies any IP address other than 2001:db8:ca2e::fa4c.

To negate a list of IP addresses, place `!` before a bracketed list of IP addresses. For example, `![192.168.1.1,192.168.1.5]` would define any IP address other than 192.168.1.1 or 192.168.1.5.

Note You must use brackets to negate a list of IP addresses.

Be careful when using the negation character with IP address lists. For example, if you use `[!192.168.1.1,!192.168.1.5]` to match any address that is not 192.168.1.1 or 192.168.1.5, the system interprets this syntax as "anything that is not 192.168.1.1, **or** anything that is not 192.168.1.5."

Because 192.168.1.5 is not 192.168.1.1, and 192.168.1.1 is not 192.168.1.5, both IP addresses match the IP address value of `[!192.168.1.1,!192.168.1.5]`, and it is essentially the same as using "`any`."

Instead, use `![192.168.1.1,192.168.1.5]`. The system interprets this as "**not** 192.168.1.1 **and not** 192.168.1.5," which matches any IP address other than those listed between brackets.

Note that you cannot logically use negation with `any` which, if negated, would indicate no address.

Intrusion Rule Header Source and Destination Ports

Within the intrusion rules editor, you specify source and destination ports in the **Source Port** and **Destination Port** fields.

Port Syntax in Intrusion Rules

The system uses a specific type of syntax to define the port numbers used in rule headers.

Note The system ignores port definitions in an intrusion rule header when the protocol is set to `ip`.

You can list ports by separating the ports with commas, as shown in the following example:

```
80, 8080, 8138, 8600-9000, !8650-8675
```

Optionally, the following example shows how you can surround a port list with brackets, which was required in previous software versions but is no longer required:

```
[80, 8080, 8138, 8600-9000, !8650-8675]
```

Note that you **must** surround negated port lists in brackets, as shown in the following example:

```
![20, 22, 23]
```

The following table summarizes the syntax you can use:

Table 17: Source/Destination Port Syntax

To Specify...	Use	Example
any port	`any`	`any`
a specific port	the port number	`80`
a range of ports	a dash between the first and last port number in the range	`80-443`
all ports less than or equal to a specific port	a dash before the port number	`-21`
all ports greater than or equal to a specific port	a dash after the port number	`80-`
all ports except a specific port or range of ports	the `!` character before the port, port list, or range of ports you want to negate Note that you can logically use negation with all port designations except any, which if negated would indicate *no port.*	`!20`
all ports defined by a port variable	the variable name, in uppercase letter, preceded by `$`	`$HTTP_PORTS`
all ports except ports defined by a port variable	the variable name, in uppercase letter, preceded by `!$`	`!$HTTP_PORTS`

Intrusion Event Details

As you construct a standard text rule, you can include contextual information that describes the vulnerability that the rule detects in exploit attempts. You can also include external references to vulnerability databases

and define the priority that the event holds in your organization. When analysts see the event, they then have information about the priority, exploit, and known mitigation readily available.

Message

You can specify meaningful text that appears as a message when the rule triggers. The message gives immediate insight into the nature of the vulnerability that the rule detects attempts to exploit. You can use any printable standard ASCII characters except curly braces ({}). The system strips quotes that completely surround the message.

Tip You must specify a rule message. Also, the message cannot consist of white space only, one or more quotation marks only, one or more apostrophes only, or any combination of just white space, quotation marks, or apostrophes.

To define the event message in the intrusion rules editor, you enter the event message in the **Message** field.

Classification

For each rule, you can specify an attack classification that appears in the packet display of the event. The following table lists the name and number for each classification.

Table 18: Rule Classifications

Number	Classification Name	Description
1	not-suspicious	Not Suspicious Traffic
2	unknown	Unknown Traffic
3	bad-unknown	Potentially Bad Traffic
4	attempted-recon	Attempted Information Leak
5	successful-recon-limited	Information Leak
6	successful-recon-largescale	Large Scale Information Leak
7	attempted-dos	Attempted Denial of Service
8	successful-dos	Denial of Service
9	attempted-user	Attempted User Privilege Gain
10	unsuccessful-user	Unsuccessful User Privilege Gain
11	successful-user	Successful User Privilege Gain
12	attempted-admin	Attempted Administrator Privilege Gain
13	successful-admin	Successful Administrator Privilege Gain
14	rpc-portmap-decode	Decode of an RPC Query
15	shellcode-detect	Executable Code was Detected

Number	Classification Name	Description
16	string-detect	A Suspicious String was Detected
17	suspicious-filename-detect	A Suspicious Filename was Detected
18	suspicious-login	An Attempted Login Using a Suspicious Username was Detected
19	system-call-detect	A System Call was Detected
20	tcp-connection	A TCP Connection was Detected
21	trojan-activity	A Network Trojan was Detected
22	unusual-client-port-connection	A Client was Using an Unusual Port
23	network-scan	Detection of a Network Scan
24	denial-of-service	Detection of a Denial of Service Attack
25	non-standard-protocol	Detection of a Non-Standard Protocol or Event
26	protocol-command-decode	Generic Protocol Command Decode
27	web-application-activity	Access to a Potentially Vulnerable Web Application
28	web-application-attack	Web Application Attack
29	misc-activity	Misc Activity
30	misc-attack	Misc Attack
31	icmp-event	Generic ICMP Event
32	inappropriate-content	Inappropriate Content was Detected
33	policy-violation	Potential Corporate Privacy Violation
34	default-login-attempt	Attempt to Login By a Default Username and Password
35	sdf	Sensitive Data
36	malware-cnc	Known malware command and control traffic
37	client-side-exploit	Known client side exploit attempt
38	file-format	Known malicious file or file based exploit

Custom Classification

If you want more customized content for the packet display description of the events generated by a rule you define, you can create a custom classification.

Argument	Description
Classification Name	The name of the classification. The page is difficult to read if you use more than 40 characters. The following characters are not supported: `<>()\'"&$;` and the space character.
Classification Description	A description of the classification. You can use alphanumeric characters and spaces. The following characters are not supported: `<>()\'"&$;`
Priority	High, medium, or low.

Custom Priority

By default, the priority of a rule derives from the event classification for the rule. However, you can override the classification priority for a rule by adding the `priority` keyword to the rule and selecting a high, medium, or low priority. For example, to assign a high priority for a rule that detects web application attacks, add the `priority` keyword to the rule and select **high** as the priority.

Custom Reference

You can use the `reference` keyword to add references to external web sites and additional information about the event. Adding a reference provides analysts with an immediately available resource to help them identify why the packet triggered a rule. The following table lists some of the external systems that can provide data on known exploits and attacks.

Table 19: External Attack Identification Systems

System ID	Description	Example ID
`bugtraq`	Bugtraq page	`8550`
`cve`	Common Vulnerabilities and Exposure ID	`2020-9607`
`mcafee`	McAfee page	`98574`
`url`	Website reference	`www.example.com?exploit=14`
`msb`	Microsoft security bulletin	`MS11-082`
`nessus`	Nessus page	`10039`
`secure-url`	Secure Website Reference (https://...)	`intranet/exploits/exploit=14` Note that you can use `secure-url` with any secure website.

You specify a reference by entering a reference value, as follows:

```
id_system,id
```

where id_system is the system being used as a prefix, and id is the CVE ID number, Arachnids ID, or URL (without `http://`).

For example, to specify the Adobe Acrobat and Reader issue documented in CVE-2020-9607, enter the value:

```
cve,2020-9607
```

Note the following when adding references to a rule:

- Do not use a space after the comma.
- Do not use uppercase letters in the system ID.

Adding a Custom Classification

Procedure

Step 1 While creating or editing a rule, choose **Edit Classifications** from the **Classification** drop-down list (**Objects** > **Intrusion Rules** > **Create Rules** > **Edit Classifications**).

If **View Classifications** displays instead, the configuration belongs to an ancestor domain, or you do not have permission to modify the configuration.

Step 2 Enter a **Classification Name** and **Classification Description** as described in Intrusion Event Details, on page 94.

Step 3 Choose a priority for the classification from the **Priority** drop-down list.

Step 4 Click **Add**.

Step 5 Click **Done**.

What to do next

- Continue with creating or editing the rule. See Writing New Rules, on page 100 or Modifying Existing Rules, on page 101 for more information.

Defining an Event Priority

Procedure

Step 1 While creating or editing a rule, choose `priority` from the **Detection Options** drop-down list.

Step 2 Click **Add Option**.

Step 3 Choose a value from the **priority** drop-down list.

Step 4 Click **Save**.

What to do next

- Continue with creating or editing the rule. See Writing New Rules, on page 100 or Modifying Existing Rules, on page 101 for more information.

Defining an Event Reference

Procedure

Step 1 While creating or editing a rule, choose `reference` from the **Detection Options** drop-down list.

Step 2 Click **Add Option**.

Step 3 Enter a value in the **reference** field as described in Intrusion Event Details, on page 94.

Step 4 Click **Save**.

What to do next

- Continue with creating or editing the rule. See Writing New Rules, on page 100 or Modifying Existing Rules, on page 101 for more information.

Custom Rule Creation

You can create a custom intrusion rule by:

- creating your own standard text rules
- saving existing standard text rules as new
- saving system-provided shared object rules as new
- importing a local rule file

The system saves the custom rule in the local rule category, regardless of the method you used to create it.

When you create a custom intrusion rule, the system assigns it a unique rule number, which has the format `GID:SID:Rev`. The elements of this number are:

GID

Generator ID. For all standard text rules, this value is 1 (Global domain or legacy GID) or 1000 - 2000 (descendant domains). For all shared object rules you save as new, this value is 1.

SID

Snort ID. Indicates whether the rule is a local rule of a system rule. When you create a new rule, the system assigns the next available SID for a local rule.

SID numbers for local rules start at 1000000, and the SID for each new local rule is incremented by one.

Rev

The revision number. For a new rule, the revision number is one. Each time you modify a custom rule the revision number increments by one.

In a custom standard text rule, you set the rule header settings and the rule keywords and arguments. You can use the rule header settings to focus the rule to only match traffic using a specific protocol and traveling to or from specific IP addresses or ports.

In a custom system-provided standard text rule or shared object rule, you are limited to modifying rule header information such as the source and destination ports and IP addresses. You cannot modify the rule keywords or arguments.

Modifying header information for a shared object rule and saving your changes creates a new instance of the rule with a generator ID (GID) of 1 (Global domain) or 1000 - 2000 (descendant domains) and the next available SID for a custom rule. The system links the new instance of the shared object rule to the reserved `soid` keyword, which maps the rule you create to the rule created by the Talos Intelligence Group. You can delete instances of a shared object rule that you create, but you cannot delete shared object rules created by Talos.

Writing New Rules

Procedure

Step 1 Choose **Objects** > **Intrusion Rules**.

Step 2 Click **Create Rule**.

Step 3 Enter a value in the **Message** field.

Step 4 Choose a value from each of the following drop-down lists:

- **Classification**
- **Action**
- **Protocol**
- **Direction**

Step 5 Enter values in the following fields:

- **Source IPs**
- **Destination IPs**
- **Source Port**
- **Destination Port**

The system uses the value `any` if you do not specify a value for these fields.

Step 6 Choose a value from the **Detection Options** drop-down list.

Step 7 Click **Add Option**.

Step 8 Enter any arguments for the keyword you added.

Step 9 Optionally, repeat steps 6 to 8.

Step 10 If you added multiple keywords, you can:

- Reorder keywords — Click the up or down arrow next to the keyword you want to move.

- Delete a keyword — Click the **X** next to that keyword.

Step 11 Click **Save As New**.

What to do next

- Enable your new or changed rules within the appropriate intrusion policy.
- Deploy configuration changes.

Modifying Existing Rules

You can save system-provided rules and rules belonging to ancestor domains as new custom rules in the local rule category, which you can then modify.

Procedure

Step 1 Access the intrusion rules using either of the following methods:

- Choose **Policies** > **Access Control heading** > **Intrusion**.

  Click **Snort 2 Version** next to the policy you want to edit and click **Rules**.

- Choose **Objects** > **Intrusion Rules**.

Step 2 Locate the rule you want to modify. You have the following choices:

- Navigate through the folders to the rule.
- Search for the rule; see Searching for Rules, on page 104.
- Filter for the group to which the rule belongs; see Filtering Rules, on page 108.

Step 3 Click **Edit** (✎) next to the rule or, in the case of search results, click the rule message.

If **View** (👁) appears instead, the configuration belongs to an ancestor domain, or you do not have permission to modify the configuration.

Step 4 Modify the rule as appropriate for the rule type.

Note
Do not modify the protocol for a shared object rule; doing so would render the rule ineffective.

Step 5 You have the following choices:

- Click **Save** if you are editing a custom rule and want to overwrite the current version of that rule.
- Click **Save As New** if you are editing a system-provided rule or any rule belonging to an ancestor domain, or if you are editing a custom rule and want to save the changes as a new rule.

What to do next

- If you want to use the local modification of the rule instead of the system-provided rule, deactivate the system-provided rule by using the procedures at Intrusion Rule States, on page 42 and activate the local rule.
- Deploy configuration changes.

Viewing Rule Documentation

From the Rule Edit page, you can view rule documentation supplied by the Talos Intelligence Group. While viewing, you can click **Rule Documentation** and other external references to view additional information provided by Talos. You can also click **Context Explorer** to view contextual information for events generated by the rule.

Procedure

Step 1 Access an intrusion rule using either of the following methods:

- Choose **Policies** > **Access Control heading** > **Intrusion**.

  Click **Snort 2 Version** next to the policy you want to edit and click **Rules**.

- Choose **Objects** > **Intrusion Rules**.

Step 2 Locate the rule you want to view. You have the following choices:

- Navigate through the folders to the rule.
- Search for the rule; see Searching for Rules, on page 104.
- Filter for the group to which the rule belongs; see Filtering Rules, on page 108.

Step 3 Click **Edit** (✎) next to the rule or, in the case of search results, click the rule message.

If **View** (👁) appears instead, the configuration belongs to an ancestor domain, or you do not have permission to modify the configuration.

Step 4 Click **View Documentation**.

Step 5 Optionally, click any of the following links:

- **Rule Documentation** —to view detailed rule specifics.
- Other external references—see Keyword Filtering, on page 106 and *Custom Reference* in Intrusion Event Details, on page 94 for information on available external references.
- **Context Explorer**—see for information on viewing contextual data for the rule in the context explorer.

Tip
Selecting an external link closes the documentation pop-up window; to exit the rule edit page without modifying the rule, select any menu path.

Adding Comments to Intrusion Rules

You can add comments to any intrusion rule. Such comments can be helpful to provide context and additional information about the rule and the exploit or policy violation it identifies.

Procedure

Step 1 Access the intrusion rules using either of the following methods:

- Choose **Policies** > **Access Control heading** > **Intrusion**.

  Click **Snort 2 Version** next to the policy you want to edit and click **Rules**.

- Choose **Objects** > **Intrusion Rules**.

Step 2 Locate the rule you want to annotate. You have the following choices:

- Navigate through the folders to the rule.
- Search for the rule; see Searching for Rules, on page 104.
- Filter for the group where the rule belongs; see Filtering Rules, on page 108.

Step 3 Click **Edit** (✎) next to the rule or, in the case of search results, click the rule message.

If **View** (👁) appears next to a rule instead, the rule belongs to an ancestor policy, or you do not have permission to modify the rule.

Step 4 Click **Rule Comment**.

Step 5 Enter your comment in the text box.

Step 6 Click **Add Comment**.

Tip
You can also add and view rule comments in an intrusion event's packet view.

What to do next

- Continue with creating or editing the rule. See Writing New Rules, on page 100 or Modifying Existing Rules, on page 101 for more information.

Deleting Custom Rules

You can delete custom rules if the rules are not currently enabled in an intrusion policy. You cannot delete either standard text rules or shared object rules provided by the system.

The system stores deleted rules in the deleted category, and you can use a deleted rule as the basis for a new rule. The Rules page in an intrusion policy does not display the deleted category, so you cannot enable deleted custom rules.

Tip Custom rules include shared object rules that you save with modified header information. The system also saves these in the local rule category and lists them with a GID of 1 (Global domain or legacy GID) or 1000 - 2000 (descendant domains). You can delete your modified version of a shared object rule, but you cannot delete the original shared object rule.

Procedure

Step 1 Access the intrusion rules using either of the following methods:

- Choose **Policies** > **Access Control heading** > **Intrusion**.

  Click **Snort 2 Version** next to the policy you want to edit and click **Rules**.

- Choose **Objects** > **Intrusion Rules**.

Step 2 You have two choices:

- Delete all local rules — Click **Delete Local Rules**, then click **OK**.
- Delete a single rule — Choose `Local Rules` from the **Group Rules By** drop-down, click **Delete** ( ) next to a rule you want to delete, and click **OK** to confirm the deletion.

Searching for Rules

The system provides thousands of standard text rules, and the Talos Intelligence Group continues to add rules as new vulnerabilities and exploits are discovered. You can easily search for specific rules so that you can activate, deactivate, or edit them.

Procedure

Step 1 Access the intrusion rules using either of the following methods:

- Choose **Policies** > **Access Control heading** > **Intrusion**.

  Click **Snort 2 Version** next to the policy you want to edit and click **Rules**.

- Choose **Objects** > **Intrusion Rules**.

Step 2 Click **Search** on the toolbar.

Step 3 Add search criteria.

Step 4 Click **Search**.

What to do next

- If you want to view or edit a located rule (or a copy of the rule, if it is a system rule), click the hyperlinked rule message. See Writing New Rules, on page 100 or Modifying Existing Rules, on page 101 for more information.

Search Criteria for Intrusion Rules

The following table describes the available search options:

Table 20: Rule Search Criteria

Option	Description
Signature ID	To search for a single rule based on Snort ID (SID), enter an SID number. To search for multiple rules, enter a comma-separated list of SID numbers. This field has an 80-character limit.
Generator ID	To search for standard text rules, select **1**. To search for shared object rules, select **3**.
Message	To search for a rule with a specific message, enter a single word from the rule message in the **Message** field. For example, to search for DNS exploits, you would enter `DNS`, or to search for buffer overflow exploits, enter `overflow`.
Protocol	To search rules that evaluate traffic of a specific protocol, select the protocol. If you do not select a protocol, search results contain rules for all protocols.
Source Port	To search for rules that inspect packets originating from a specified port, enter a source port number or a port-related variable.
Destination Port	To search for rules that inspect packets destined for a specific port, enter a destination port number or a port-related variable.
Source IP	To search for rules that inspect packets originating from a specified IP address, enter a source IP address or an IP address-related variable.
Destination IP	To search for rules that inspect packets destined for a specified IP address, enter a destination IP address or an IP address-related variable.
Keyword	To search for specific keywords, you can use the keyword search options. You select a keyword and enter a keyword value for which to search. You can also precede the keyword value with an exclamation point (`!`) to match any value other than the specified value.
Category	To search for rules in a specific category, select the category from the **Category** list.
Classification	To search for rules that have a specific classification, select the classification name from the **Classification** list.
Rule State	To search for rules within a specific policy and a specific rule state, select the policy from the first **Rule State** list, and choose a state from the second list to search for rules set to **Generate Events**, **Drop and Generate Events**, or **Disabled**.

Rule Filtering on the Intrusion Rules Editor Page

You can filter the rules on the intrusion rules editor page to display a subset of rules. This can be useful, for example, when you want to modify a rule or change its state but have difficulty finding it among the thousands of rules available.

When you enter a filter, the page displays any folder that includes at least one matching rule, or a message when no rule matches.

Filtering Guidelines

Your filter can include special keywords and their arguments, character strings, and literal character strings in quotes, with spaces separating multiple filter conditions. A filter cannot include regular expressions, wild card characters, or any special operator such as a negation character (!), a greater than symbol (>), less than symbol (<), and so on.

All keywords, keyword arguments, and character strings are case-insensitive. Except for the `gid` and `sid` keywords, all arguments and strings are treated as partial strings. Arguments for `gid` and `sid` return only exact matches.

You can expand a folder on the original, unfiltered page and the folder remains expanded when the subsequent filter returns matches in that folder. This can be useful when the rule you want to find is in a folder that contains a large number of rules.

You cannot constrain a filter with a subsequent filter. Any filter you enter searches the entire rules database and returns all matching rules. When you enter a filter while the page still displays the result of a previous filter, the page clears and returns the result of the new filter instead.

You can use the same features with rules in a filtered or unfiltered list. For example, you can edit rules in a filtered or unfiltered list on the intrusion rules editor page. You can also use any of the options in the context menu for the page.

Tip Filtering may take significantly longer when the combined total of rules in all sub-groups is large because rules appear in multiple categories, even when the total number of unique rules is much smaller.

Keyword Filtering

Each rule filter can include one or more keywords in the format:

```
keyword:argument
```

where keyword is one of the keywords in the following table and argument is a single, case-insensitive, alphanumeric string to search for in the specific field or fields relevant to the keyword.

Arguments for all keywords except `gid` and `sid` are treated as partial strings. For example, the argument `123` returns `"12345"`, `"41235"`, `"45123"`, and so on. The arguments for `gid` and `sid` return only exact matches; for example, `sid:3080` returns only SID 3080.

Tip You can search for a partial SID by filtering with one or more character strings.

The following table describes the specific filtering keywords and arguments you can use to filter rules.

Table 21: Rule Filter Keywords

Keyword	Description	Example
`arachnids`	Returns one or more rules based on all or part of the Arachnids ID in a rule reference.	`arachnids:181`
`bugtraq`	Returns one or more rules based on all or part of the Bugtraq ID in a rule reference.	`bugtraq:2120`
`cve`	Returns one or more rules based on all or part of the CVE number in a rule reference.	`cve:2003-0109`
`gid`	The argument `1` returns standard text rules. The argument `3` returns shared object rules.	`gid:3`
`mcafee`	Returns one or more rules based on all or part of the McAfee ID in a rule reference.	`mcafee:10566`
`msg`	Returns one or more rules based on all or part of the rule Message field, also known as the event message.	`msg:chat`
`nessus`	Returns one or more rules based on all or part of the Nessus ID in a rule reference.	`nessus:10737`
`ref`	Returns one or more rules based on all or part of a single alphanumeric string in a rule reference or in the rule Message field.	`ref:MS03-039`
`sid`	Returns the rule with the exact Snort ID.	`sid:235`
`url`	Returns one or more rules based on all or part of the URL in a rule reference.	`url:faqs.org`

Character String Filtering

Each rule filter can include one or more alphanumeric character strings. Character strings search the rule **Message** field, Snort ID (SID), and Generator ID (GID). For example, the string `123` returns the strings `"Lotus123"`, `"123mania"`, and so on in the rule message, and also returns SID 6123, SID 12375, and so on.

All character strings are case-insensitive and are treated as partial strings. For example, any of the strings `ADMIN`, `admin`, or `Admin` return `"admin"`, `"CFADMIN"`, `"Administrator"` and so on.

You can enclose character strings in quotes to return exact matches. For example, the literal string `"overflow attempt"` in quotes returns only that exact string, whereas a filter comprised of the two strings `overflow` and `attempt` without quotes returns `"overflow attempt"`, `"overflow multipacket attempt"`, `"overflow with evasion attempt"`, and so on.

Combination Keyword and Character String Filtering

You can narrow filter results by entering any combination of keywords, character strings, or both, separated by spaces. The result includes any rule that matches all the filter conditions.

You can enter multiple filter conditions in any order. For example, each of the following filters returns the same rules:

- `url:at login attempt cve:200`
- `login attempt cve:200 url:at`
- `login cve:200 attempt url:at`

Filtering Rules

On the Intrusion Rules page, you can filter rules into subsets so you can more easily find specific rules. You can then use any of the page features, including choosing any of the features available in the context menu.

Rule filtering can be particularly useful to locate a specific rule to edit.

Procedure

Step 1 Access the intrusion rules using either of the following methods:

- Choose **Policies** > **Access Control heading** > **Intrusion**.

  Click **Snort 2 Version** next to the policy you want to edit and click **Rules**.
- Choose **Objects** > **Intrusion Rules**.

Step 2 Prior to filtering, you have the following choices:

- Expand any rule group you want to expand. Some rule groups also have sub-groups that you can expand.

  Expanding a group on the original, unfiltered page can be useful when you expect that a rule might be in that group. The group remains expanded when the subsequent filter results in a match in that folder, and when you return to the original, unfiltered page by clicking filter **Clear** ( ).
- Choose a different grouping method from the **Group Rules By** drop-down list.

Step 3 Enter filter constraints in the text box next to **Filter** ( ) under the **Group Rules By** list.

Step 4 Press Enter.

Note
Clear the current filtered list by clicking filter **Clear** ( ).

Keywords and Arguments in Intrusion Rules

Using the rules language, you can specify the behavior of a rule by combining keywords. Keywords and their associated values (called *arguments*) dictate how the system evaluates packets and packet-related values that the rules engine tests. The system currently supports keywords that allow you to perform inspection functions, such as content matching, protocol-specific pattern matching, and state-specific matching. You can define up to 100 arguments per keyword, and combine any number of compatible keywords to create highly specific rules. This helps decrease the chance of false positives and false negatives and focus the intrusion information you receive.

Note that you can also use adaptive profile updates in passive deployments to dynamically adapt active rule processing for specific packets based on rule metadata and host information.

Keywords described in this section are listed under Detection Options in the rules editor.

The content and protected_content Keywords

Use the `content` keyword or the `protected_content` keyword to specify content that you want to detect in a packet.

You should almost always follow a `content` or `protected_content` keyword by modifiers that indicate where the content should be searched for, whether the search is case sensitive, and other options.

Note that all content matches must be true for the rule to trigger an event, that is, each content match has an AND relationship with the others.

Note also that, in an inline deployment, you can set up rules that match malicious content and then replace it with your own text string of equal length.

content

When you use the `content` keyword, the rules engine searches the packet payload or stream for that string. For example, if you enter `/bin/sh` as the value for one of the `content` keywords, the rules engine searches the packet payload for the string `/bin/sh`.

Match content using either an ASCII string, hexadecimal content (binary byte code), or a combination of both. Surround hexadecimal content with pipe characters (|) in the keyword value. For example, you can mix hexadecimal content and ASCII content using something that looks like `|90C8 C0FF FFFF|/bin/sh`.

You can specify multiple content matches in a single rule. To do this, use additional instances of the `content` keyword. For each content match, you can indicate that content matches must be found in the packet payload or stream for the rule to trigger.

Caution You may invalidate your intrusion policy if you create a rule that includes only one `content` keyword and that keyword has the **Not** option selected.

protected_content

The `protected_content` keyword allows you to encode your search content string before configuring the rule argument. The original rule author uses a hash function (SHA-512, SHA-256, or MD5) to encode the string before configuring the keyword.

When you use the `protected_content` keyword instead of the `content` keyword, there is no change to how the rules engine searches the packet payload or stream for that string and most of the keyword options function as expected. The following table summarizes the exceptions, where the `protected_content` keyword options differ from the `content` keyword options.

Table 22: protected_content Option Exceptions

Option	Description
Hash Type	New option for the `protected_content` rule keyword.
Case Insensitive	Not supported
Within	Not supported
Depth	Not supported
Length	New option for the `protected_content` rule keyword.
Use Fast Pattern Matcher	Not supported
Fast Pattern Matcher Only	Not supported
Fast Pattern Matcher Offset and Length	Not supported

Cisco recommends that you include at least one `content` keyword in rules that include a `protected_content` keyword to ensure that the rules engine uses the fast pattern matcher, which increases processing speed and improves performance. Position the content `keyword` before the `protected_content` keyword in the rule. Note that the rules engine uses the fast pattern matcher when a rule includes at least one `content` keyword, regardless of whether you enable the `content` keyword Use Fast Pattern Matcher argument.

Caution You may invalidate your intrusion policy if you create a rule that includes only one `protected_content` keyword and that keyword has the **Not** option selected.

Basic content and protected_content Keyword Arguments

You can constrain the location and case-sensitivity of content searches with parameters that modify the `content` or `protected_content` keyword. Configure options that modify the `content` or `protected_content` keyword to specify the content for which you want to search.

Case Insensitive

Note This option is **not** supported when configuring the `protected_content` keyword.

You can instruct the rules engine to ignore case when searching for content matches in ASCII strings. To make your search case-insensitive, check **Case Insensitive** when specifying a content search.

Hash Type

Note This option is **only** configurable with the `protected_content` keyword.

Use the **Hash Type** drop-down to identify the hash function you used to encode your search string. The system supports SHA-512, SHA-256, and MD5 hashing for `protected_content` search strings. If the length of your hashed content does not match the selected hash type, the system does **not** save the rule.

The system automatically selects the Cisco-set default value. When **Default** is selected, no specific hash function is written into the rule and the system assumes SHA-512 for the hash function.

Raw Data

The **Raw Data** option instructs the rules engine to analyze the original packet payload before analyzing the normalized payload data (decoded by a network analysis policy) and does not use an argument value. You can use this keyword when analyzing telnet traffic to check the telnet negotiation options in the payload before normalization.

You cannot use the **Raw Data** option together in the same `content` or `protected_content` keyword with any HTTP content option.

Tip You can configure the HTTP Inspect preprocessor **Client Flow Depth** and **Server Flow Depth** options to determine whether raw data is inspected in HTTP traffic, and how much raw data is inspected.

Not

Select the **Not** option to search for content that does not match the specified content. If you create a rule that includes a `content` or `protected_content` keyword with the **Not** option selected, you must also include in the rule at least one other `content` or `protected_content` keyword without the **Not** option selected.

Caution Do not create a rule that includes only one `content` or `protected_content` keyword if that keyword has the **Not** option selected. You may invalidate your intrusion policy.

For example, SMTP rule 1:2541:9 includes three `content` keywords, one of which has the **Not** option selected. A custom rule based on this rule would be invalid if you removed all of the `content` keywords except the one with the **Not** option selected. Adding such a rule to your intrusion policy could invalidate the policy.

Tip You cannot select the **Not** check box and the **Use Fast Pattern Matcher** check box with the same `content` keyword.

content and protected_content Keyword Search Locations

You can use search location options to specify where to begin searching for the specified content and how far to continue searching.

Permitted Combinations: content Search Location Arguments

You can use either of two `content` location pairs to specify where to begin searching for the specified content and how far to continue searching, as follows:

- Use **Offset** and **Depth** together to search relative to the beginning of the packet payload.
- Use **Distance** and **Within** together to search relative to the current search location.

When you specify only one of a pair, the default for the other option in the pair is assumed.

You cannot mix the **Offset** and **Depth** options with the **Distance** and **Within** options. For example, you cannot pair **Offset** and **Within**. You can use any number of location options in a rule.

When no location is specified, the defaults for **Offset** and **Depth** are assumed; that is, the content search starts at the beginning of the packet payload and continues to the end of the packet.

You can also use an existing `byte_extract` variable to specify the value for a location option.

Tip You can use any number of location options in a rule.

Permitted Combinations: protected_content Search Location Arguments

Use the required **Length** `protected_content` location option in combination with either the **Offset** or **Distance** location option to specify where to begin searching for the specified content and how far to continue searching, as follows:

- Use **Length** and **Offset** together to search for the protected string relative to the beginning of the packet payload.
- Use **Length** and **Distance** together to search for the protected string relative to the current search location.

Tip You cannot mix the **Offset** and **Distance** options within a single keyword configuration, but you can use any number of location options in a rule.

When no location is specified, the defaults are assumed; that is, the content search starts at the beginning of the packet payload and continues to the end of the packet.

You can also use an existing `byte_extract` variable to specify the value for a location option.

content and protected_content Search Location Arguments

Depth

Note This option is **only** supported when configuring the `content` keyword.

Specifies the maximum content search depth, in bytes, from the beginning of the offset value, or if no offset is configured, from the beginning of the packet payload.

For example, in a rule with a content value of `cgi-bin/phf`, and `offset` value of `3`, and a `depth` value of `22`, the rule starts searching for a match to the `cgi-bin/phf` string at byte 3, and stops after processing 22 bytes (byte 25) in packets that meet the parameters specified by the rule header.

You must specify a value that is greater than or equal to the length of the specified content, up to a maximum of 65535 bytes. You cannot specify a value of 0.

The default depth is to search to the end of the packet.

Distance

Instructs the rules engine to identify subsequent content matches that occur a specified number of bytes after the previous successful content match.

Because the distance counter starts at byte 0, specify one less than the number of bytes you want to move forward from the last successful content match. For example, if you specify 4, the search begins at the fifth byte.

You can specify a value of -65535 to 65535 bytes. If you specify a negative `Distance` value, the byte you start searching on may fall outside the beginning of a packet. Any calculations will take into account the bytes outside the packet, even though the search actually starts on the first byte in the packet. For example, if the current location in the packet is the fifth byte, and the next content rule option specifies a `Distance` value of -10 and a `Within` value of 20, the search starts at the beginning of the payload and the `Within` option is adjusted to 15.

The default distance is 0, meaning the current location in the packet subsequent to the last content match.

Length

Note This option is **only** supported when configuring the `protected_content` keyword.

The **Length** `protected_content` keyword option indicates the length, in bytes, of the unlashed search string.

For example, if you used the content `Sample1` to generate a secure hash, use `7` for the **Length** value. You **must** enter a value in this field.

Offset

Specifies in bytes where in the packet payload to start searching for content relative to the beginning of the packet payload. You can specify a value of 65535 to 65535 bytes.

Because the offset counter starts at byte 0, specify one less than the number of bytes you want to move forward from the beginning of the packet payload. For example, if you specify 7, the search begins at the eighth byte.

The default offset is 0, meaning the beginning of the packet.

Within

Note This option is **only** supported when configuring the `content` keyword.

The **Within** option indicates that, to trigger the rule, the next content match must occur within the specified number of bytes after the end of the last successful content match. For example, if you specify a **Within** value

of `8`, the next content match must occur within the next eight bytes of the packet payload or it does not meet the criteria that triggers the rule.

You can specify a value that is greater than or equal to the length of the specified content, up to a maximum of 65535 bytes.

The default for **Within** is to search to the end of the packet.

Overview: HTTP content and protected_content Keyword Arguments

HTTP `content` or `protected_content` keyword options let you specify where to search for content matches within an HTTP message decoded by the HTTP Inspect preprocessor.

Two options search status fields in HTTP responses:

- **HTTP Status Code**
- **HTTP Status Message**

Note that although the rules engine searches the raw, unnormalized status fields, these options are listed here separately to simplify explanation below of the restrictions to consider when combining other raw HTTP fields and normalized HTTP fields.

Five options search normalized fields in HTTP requests, responses, or both, as appropriate :

- **HTTP URI**
- **HTTP Method**
- **HTTP Header**
- **HTTP Cookie**
- **HTTP Client Body**

Three options search raw (unnormalized) non-status fields in HTTP requests, responses, or both, as appropriate:

- **HTTP Raw URI**
- **HTTP Raw Header**
- **HTTP Raw Cookie**

Use the following guidelines when selecting HTTP `content` options:

- HTTP `content` options apply only to TCP traffic.
- To avoid a negative impact on performance, select only those parts of the message where the specified content might appear.

  For example, when traffic is likely to include large cookies such as those in shopping cart messages, you might search for the specified content in the HTTP header but not in HTTP cookies.

- To take advantage of HTTP Inspect preprocessor normalization, and to improve performance, any HTTP-related rule you create should at a minimum include at least one `content` or `protected_content` keyword with an **HTTP URI**, **HTTP Method**, **HTTP Header**, or **HTTP Client Body** option selected.

- You cannot use the `replace` keyword in conjunction with HTTP `content` or `protected_content` keyword options.

You can specify a single normalized HTTP option or status field, or use normalized HTTP options and status fields in any combination to target a content area to match. However, note the following restrictions when using HTTP field options:

- You cannot use the **Raw Data** option together in the same `content` or `protected_content` keyword with any HTTP option.
- You cannot use a raw HTTP field option (**HTTP Raw URI**, **HTTP Raw Header**, or **HTTP Raw Cookie**) together in the same `content` or `protected_content` keyword with its normalized counterpart (**HTTP URI**, **HTTP Header**, or **HTTP Cookie**, respectively).
- You cannot select **Use Fast Pattern Matcher** in combination with one or more of the following HTTP field options:

  HTTP Raw URI, **HTTP Raw Header**, **HTTP Raw Cookie**, **HTTP Cookie**, **HTTP Method**, **HTTP Status Message**, or **HTTP Status Code**

  However, you can include the options above in a `content` or `protected_content` keyword that also uses the fast pattern matcher to search one of the following normalized fields:

  HTTP URI, **HTTP Header**, or **HTTP Client Body**

  For example, if you select **HTTP Cookie**, **HTTP Header**, and **Use Fast Pattern Matcher**, the rules engine searches for content in both the HTTP cookie and the HTTP header, but the fast pattern matcher is applied only to the HTTP header, not to the HTTP cookie.

- When you combine restricted and unrestricted options, the fast pattern matcher searches only the unrestricted fields you specify to test whether to pass the rule to the intrusion rules editor for complete evaluation, including evaluation of the restricted fields.

HTTP content and protected_content Keyword Arguments

HTTP URI

Select this option to search for content matches in the normalized request URI field.

Note that you cannot use this option in combination with the `pcre` keyword HTTP URI (U) option to search the same content.

Note A pipelined HTTP request packet contains multiple URIs. When **HTTP URI** is selected and the rules engine detects a pipelined HTTP request packet, the rules engine searches all URIs in the packet for a content match.

HTTP Raw URI

Select this option to search for content matches in the normalized request URI field.

Note that you cannot use this option in combination with the `pcre` keyword HTTP URI (U) option to search the same content.

Note A pipelined HTTP request packet contains multiple URIs. When **HTTP URI** is selected and the rules engine detects a pipelined HTTP request packet, the rules engine searches all URIs in the packet for a content match.

HTTP Method

Select this option to search for content matches in the request method field, which identifies the action such as GET and POST to take on the resource identified in the URI.

HTTP Header

Select this option to search for content matches in the normalized header field, except for cookies, in HTTP requests; also in responses when the HTTP Inspect preprocessor **Inspect HTTP Responses** option is enabled.

Note that you cannot use this option in combination with the `pcre` keyword HTTP header (H) option to search the same content.

HTTP Raw Header

Select this option to search for content matches in the raw header field, except for cookies, in HTTP requests; also in responses when the HTTP Inspect preprocessor I**nspect HTTP Responses** option is enabled.

Note that you cannot use this option in combination with the `pcre` keyword HTTP raw header (D) option to search the same content.

HTTP Cookie

Select this option to search for content matches in any cookie identified in a normalized HTTP client request header; also in response set-cookie data when the HTTP Inspect preprocessor **Inspect HTTP Responses** option is enabled. Note that the system treats cookies included in the message body as body content.

You must enable the HTTP Inspect preprocessor **Inspect HTTP Cookies** option to search only the cookie for a match; otherwise, the rules engine searches the entire header, including the cookie.

Note the following:

- You cannot use this option in combination with the `pcre` keyword HTTP cookie (C) option to search the same content.
- The `Cookie:` and `Set-Cookie:` header names, leading spaces on the header line, and the `CRLF` that terminates the header line are inspected as part of the header and not as part of the cookie.

HTTP Raw Cookie

Select this option to search for content matches in any cookie identified in a raw HTTP client request header; also in response set-cookie data when the HTTP Inspect preprocessor **Inspect HTTP Responses** option is enabled; note that the system treats cookies included in the message body as body content.

You must enable the HTTP Inspect preprocessor **Inspect HTTP Cookies** option to search only the cookie for a match; otherwise, the rules engine searches the entire header, including the cookie.

Note the following:

- You cannot use this option in combination with the `pcre` keyword HTTP raw cookie (K) option to search the same content.
- The `Cookie:` and `Set-Cookie:` header names, leading spaces on the header line, and the `CRLF` that terminates the header line are inspected as part of the header and not as part of the cookie.

HTTP Client Body

Select this option to search for content matches in the message body in an HTTP client request.

Note that for this option to function, you must specify a value of 0 to 65535 for the HTTP Inspect preprocessor **HTTP Client Body Extraction Depth** option.

HTTP Status Code

Select this option to search for content matches in the 3-digit status code in an HTTP response.

You must enable the HTTP Inspect preprocessor **Inspect HTTP Responses** option for this option to return a match.

HTTP Status Message

Select this option to search for content matches in the textual description that accompanies the status code in an HTTP response.

You must enable the HTTP Inspect preprocessor **Inspect HTTP Responses** option for this option to return a match.

Overview: content Keyword Fast Pattern Matcher

Note These options are **not** supported when configuring the `protected_content` keyword.

The fast pattern matcher quickly determines which rules to evaluate before passing a packet to the rules engine. This initial determination improves performance by significantly reducing the number of rules used in packet evaluation.

By default, the fast pattern matcher searches packets for the longest content specified in a rule; this is to eliminate as much as possible needless evaluation of a rule. Consider the following example rule fragment:

```
alert tcp any any -> any 80 (msg:"Exploit"; content:"GET";
http_method; nocase; content:"/exploit.cgi"; http_uri;
nocase;)
```

Almost all HTTP client requests contain the content `GET`, but few will contain the content `/exploit.cgi`. Using `GET` as the fast pattern content would cause the rules engine to evaluate this rule in most cases and would rarely result in a match. However, most client `GET` requests would not be evaluated using `/exploit.cgi`, thus increasing performance.

The rules engine evaluates the packet against the rule only when the fast pattern matcher detects the specified content. For example, if one `content` keyword in a rule specifies the content `short`, another specifies `longer`, and a third specifies `longest`, the fast pattern matcher will use the content `longest` and the rule will be evaluated only if the rules engine finds `longest` in the payload.

content Keyword Fast Pattern Matcher Arguments

Use Fast Pattern Matcher

Use this option to specify a shorter search pattern for the fast pattern matcher to use. Ideally, the pattern you specify is less likely to be found in the packet than the longest pattern and, therefore, more specifically identifies the targeted exploit.

Note the following restrictions when selecting **Use Fast Pattern Matcher** and other options in the same `content` keyword:

- You can specify **Use Fast Pattern Matcher** only one time per rule.
- You cannot use **Distance**, **Within**, **Offset**, or **Depth** when you select **Use Fast Pattern Matcher** in combination with **Not**.
- You cannot select Use Fast Pattern Matcher in combination with any of the following HTTP field options:

  HTTP Raw URI, **HTTP Raw Header**, **HTTP Raw Cookie**, **HTTP Cookie**, **HTTP Method**, **HTTP Status Message**, or **HTTP Status Code**

  However, you can include the options above in a `content` keyword that also uses the fast pattern matcher to search one of the following normalized fields:

  HTTP URI, **HTTP Header**, or **HTTP Client Body**

  For example, if you select **HTTP Cookie**, **HTTP Header**, and **Use Fast Pattern Matcher**, the rules engine searches for content in both the HTTP cookie and the HTTP header, but the fast pattern matcher is applied only to the HTTP header, not to the HTTP cookie.

  Note that you cannot use a raw HTTP field option (**HTTP Raw URI**, **HTTP Raw Header**, or **HTTP Raw Cookie**) together in the same `content` keyword with its normalized counterpart (**HTTP URI**, **HTTP Header**, or **HTTP Cookie**, respectively).

  When you combine restricted and unrestricted options, the fast pattern matcher searches only the unrestricted fields you specify to test whether to pass the packet to the rules engine for complete evaluation, including evaluation of the restricted fields.

- Optionally, when you select **Use Fast Pattern Matcher** you can also select **Fast Pattern Matcher Only** or **Fast Pattern Matcher Offset and Length**, but not both.
- You cannot use the fast pattern matcher when inspecting Base64 data.

Fast Pattern Matcher Only

This option allows you to use the `content` keyword only as a fast pattern matcher option and not as a rule option. You can use this option to conserve resources when rules engine evaluation of the specified content is not necessary. For example, consider a case where a rule requires only that the content `12345` be anywhere in the payload. When the fast pattern matcher detects the pattern, the packet can be evaluated against additional keywords in the rule. There is no need for the rules engine to reevaluate the packet to determine if it includes the pattern `12345`.

You would not use this option when the rule contains other conditions relative to the specified content. For example, you would not use this option to search for the content `1234` if another rule condition sought to determine if `abcd` occurs before `1234`. In this case, the rules engine could not determine the relative location because specifying **Fast Pattern Matcher Only** instructs the rules engine not to search for the specified content.

Note the following conditions when using this option:

- The specified content is location-independent; that is, it may occur anywhere in the payload; thus, you cannot use positional options (**Distance**, **Within**, **Offset**, **Depth**, or **Fast Pattern Matcher Offset and Length**).
- You cannot use this option in combination with **Not**.

- You cannot use this option in combination with **Fast Pattern Matcher Offset and Length**.
- The specified content will be treated as case-insensitive, because all patterns are inserted into the fast pattern matcher in a case-insensitive manner; this is handled automatically, so it is not necessary to select **Case Insensitive** when you select this option.
- You should not immediately follow a `content` keyword that uses the **Fast Pattern Matcher Only** option with the following keywords, which set the search location relative to the current search location:
  - `isdataat`
  - `pcre`
  - `content` when **Distance** or **Within** is selected
  - `content` when **HTTP URI** is selected
  - `asn1`
  - `byte_jump`
  - `byte_test`
  - `byte_math`
  - `byte_extract`
  - `base64_decode`

Fast Pattern Matcher Offset and Length

The **Fast Pattern Matcher Offset and Length** option allows you to specify a portion of the content to search. This can reduce memory consumption in cases where the pattern is very long and only a portion of the pattern is sufficient to identify the rule as a likely match. When a rule is selected by the fast pattern matcher, the entire pattern is evaluated against the rule.

You determine the portion for the fast pattern matcher to use by specifying in bytes where to begin the search (offset) and how far into the content (length) to search, using the syntax:

```
offset,length
```

For example, for the content:

```
1234567
```

if you specify the number of offset and length bytes as:

```
1,5
```

the fast pattern matcher searches only for the content `23456`.

Note that you cannot use this option together with **Fast Pattern Matcher Only**.

The replace Keyword

You can use the `replace` keyword in an inline deployment to replace specified content or to replace content in SSL traffic detected by the Cisco SSL Appliance.

To use the `replace` keyword, construct a custom standard text rule that uses the `content` keyword to look for a specific string. Then use the `replace` keyword to specify a string to replace the content. The replace value and content value must be the same length.

Note You **cannot** use the `replace` keyword to replace hashed content in a `protected_content` keyword.

Optionally, you can enclose the replacement string in quotation marks for backward compatibility with previous software versions. If you do not include quotation marks, they are added to the rule automatically so the rule is syntactically correct. To include a leading or trailing quotation mark as part of the replacement text, you must use a backslash to escape it, as shown in the following example:

```
"replacement text plus \"quotation\" marks""
```

A rule can contain multiple `replace` keywords, but only one per `content` keyword. Only the first instance of the content found by the rule is replaced.

The following are example uses of the `replace` keyword:

- If the system detects an incoming packet that contains an exploit, you can replace the malicious string with a harmless one. Sometimes this technique is more successful than simply dropping the offending packet. In some attack scenarios, the attacker simply resends the dropped packet until it bypasses your network defenses or floods your network. By substituting one string for another rather than dropping the packet, you may trick the attacker into believing that the attack was launched against a target that was not vulnerable.
- If you are concerned about reconnaissance attacks that try to learn whether you are running a vulnerable version of, for example, a web server, then you can detect the outgoing packet and replace the banner with your own text.

Note Make sure that you set the rule state to Generate Events in the inline intrusion policy where you want to use the replace rule; setting the rule to Drop and Generate events would cause the packet to drop, which would prevent replacing the content.

As part of the string replacement process, the system automatically updates the packet checksums so that the destination host can receive the packet without error.

Note that you cannot use the `replace` keyword in combination with HTTP request message `content` keyword options.

The byte_jump Keyword

The `byte_jump` keyword calculates the number of bytes defined in a specified byte segment, and then skips that number of bytes within the packet, either forward from the end of the specified byte segment, or from the beginning or end of the packet payload, or from a point relative to the last content match, depending on the options you specify. This is useful in packets where a specific segment of bytes describe the number of bytes included in variable data within the packet.

The following table describes the arguments required by the `byte_jump` keyword.

Table 23: Required byte_jump Arguments

Argument	Description
Bytes	The number of bytes to pick up from the packet. If used without DCE/RPC, the allowed values are 0 to 10, with the following restrictions: • If used with the `From End` argument, bytes can be 0. If Bytes is 0, the extracted value is 0. • If you specify a number of bytes other than 1, 2, or 4, you must specify a Number Type (hexadecimal, octal, or decimal.) If used with DCE/RPC, allowed values are 1, 2, and 4.
Offset	The number of bytes into the payload to start processing. The `offset` counter starts at byte 0, so calculate the `offset` value by subtracting 1 from the number of bytes you want to jump forward from the beginning of the packet payload or the last successful content match. You can specify -65535 to 65535 bytes. You can also use an existing `byte_extract` variable or `byte_math` result to specify the value for this argument.

The following table describes options you can use to define how the system interprets the values you specified for the required arguments.

Table 24: Additional Optional byte_jump Arguments

Argument	Description
Relative	Makes the offset relative to the last pattern found in the last successful content match.
Align	Rounds the number of converted bytes up to the next 32-bit boundary.
Multiplier	Indicates the value by which the rules engine should multiply the `byte_jump` value obtained from the packet to get the final `byte_jump` value. That is, instead of skipping the number of bytes defined in a specified byte segment, the rules engine skips that number of bytes multiplied by an integer you specify with the Multiplier argument.
Post Jump Offset	The number of bytes -65535 through 65535 to skip forward or backward after applying other `byte_jump` arguments. A positive value skips forward and a negative value skips backward. Leave the field blank or enter `0` to disable. Note that some `byte_jump` arguments do not apply when you select the **DCE/RPC** argument.
From Beginning	Indicates that the rules engine should skip the specified number of bytes in the payload starting from the beginning of the packet payload, instead of from the current position in the packet.
From End	The jump will originate from the byte that follows the last byte of the buffer.

Argument	Description
Bitmask	Applies the specified hexadecimal bitmask using the AND operator to the bytes extracted from the Bytes argument. A bitmask can be 1 to 4 bytes. The result will be right-shifted by the number of bits equal to the number of trailing zeros in the mask.

You can specify only one of **DCE/RPC**, **Endian**, or **Number Type**.

If you want to define how the `byte_jump` keyword calculates the bytes, you can choose from the arguments described in the following table. If you do not select a byte-ordering argument, the rules engine uses big endian byte order.

Table 25: Byte-Ordering byte_jump Arguments

Argument	Description
Big Endian	Processes data in big endian byte order, which is the default network byte order.
Little Endian	Processes data in little endian byte order.
DCE/RPC	Specifies a `byte_jump` keyword for traffic processed by the DCE/RPC preprocessor. The DCE/RPC preprocessor determines big endian or little endian byte order, and the **Number Type** and **Endian** arguments do not apply. When you enable this argument, you can also use `byte_jump` in conjunction with other specific DCE/RPC keywords.

Define how the system views string data in a packet by using one of the arguments in the following table.

Table 26: Number Type Arguments

Argument	Description
Hexadecimal String	Represents converted string data in hexadecimal format.
Decimal String	Represents converted string data in decimal format.
Octal String	Represents converted string data in octal format.

For example, if the values you set for `byte_jump` are as follows:

- Bytes = 4
- Offset = 12
- Relative enabled
- Align enabled

the rules engine calculates the number described in the four bytes that appear 13 bytes after the last successful content match, and skips ahead that number of bytes in the packet. For instance, if the four calculated bytes in a specific packet were `00 00 00 1F`, the rules engine would convert this to 31. Because `align` is specified

(which instructs the engine to move to the next 32-bit boundary), the rules engine skips ahead 32 bytes in the packet.

Alternately, if the values you set for `byte_jump` are as follows:

- Bytes = 4
- Offset = 12
- From Beginning enabled
- Multiplier = 2

the rules engine calculates the number described in the four bytes that appear 13 bytes after the beginning of the packet. Then, the engine multiplies that number by two to obtain the total number of bytes to skip. For instance, if the four calculated bytes in a specific packet were `00 00 00 1F`, the rules engine would convert this to 31, then multiply it by two to get 62. Because From Beginning is enabled, the rules engine skips the first 63 bytes in the packet.

The byte_test Keyword

The `byte_test` keyword tests the specified byte segment against the Value argument and its operator.

The following table describes the required arguments for the `byte_test` keyword.

Table 27: Required byte_test Arguments

Argument	Description
Bytes	The number of bytes to calculate from the packet. If used without DCE/RPC, the allowed values are 1 to 10. However, if you specify a number of bytes other than 1, 2, or 4, you must specify a Number Type (hexadecimal, octal, or decimal.). If used with DCE/RPC, allowed values are 1, 2, and 4.
Value	Value to test, including its operator. Supported operators: <, >, =, !, &, ^, !>, !<, !=, !&, or !^. For example, if you specify `!1024`, `byte_test` would convert the specified number, and if it did not equal 1024, it would generate an event (if all other keyword parameters matched). Note that ! and != are equivalent. You can also use an existing `byte_extract` variable or `byte_math` result to specify the value for this argument.
Offset	The number of bytes into the payload to start processing. The `offset` counter starts at byte 0, so calculate the `offset` value by subtracting 1 from the number of bytes you want to count forward from the beginning of the packet payload or the last successful content match. You can use an existing `byte_extract` variable or `byte_math` result to specify the value for this argument.

You can further define how the system uses `byte_test` arguments with the arguments described in the following table.

Table 28: Additional Optional byte_test Arguments

Argument	Description
Bitmask	Applies the specified hexadecimal bitmask using the AND operator to the bytes extracted from the Bytes argument. A bitmask can be 1 to 4 bytes. The result will be right-shifted by the number of bits equal to the number of trailing zeros in the mask.
Relative	Makes the offset relative to the last successful pattern match.

You can specify only one of **DCE/RPC**, **Endian**, or **Number Type**.

To define how the `byte_test` keyword calculates the bytes it tests, choose from the arguments in the following table. If you do not select a byte-ordering argument, the rules engine uses big endian byte order.

Table 29: Byte-Ordering byte_test Arguments

Argument	Description
Big Endian	Processes data in big endian byte order, which is the default network byte order.
Little Endian	Processes data in little endian byte order.
DCE/RPC	Specifies a `byte_test` keyword for traffic processed by the DCE/RPC preprocessor. The DCE/RPC preprocessor determines big endian or little endian byte order, and the **Number Type** and **Endian** arguments do not apply. When you enable this argument, you can also use `byte_test` in conjunction with other specific DCE/RPC keywords.

You can define how the system views string data in a packet by using one of the arguments in the following table.

Table 30: Number Type byte-test Arguments

Argument	Description
Hexadecimal String	Represents converted string data in hexadecimal format.
Decimal String	Represents converted string data in decimal format.
Octal String	Represents converted string data in octal format.

For example, if the value for `byte_test` is specified as the following:

- Bytes = 4
- Operator and Value > 128
- Offset = 8

- Relative enabled

The rules engine calculates the number described in the four bytes that appear 9 bytes away from (relative to) the last successful content match, and, if the calculated number is larger than 128 bytes, the rule is triggered.

The byte_extract Keyword

You can use the `byte_extract` keyword to read a specified number of bytes from a packet into a variable. You can then use the variable later in the same rule as the value for specific arguments in certain other detection keywords.

This is useful, for example, for extracting data size from packets where a specific segment of bytes describes the number of bytes included in data within the packet. For example, a specific segment of bytes might say that subsequent data is comprised of four bytes; you can extract the data size of four bytes to use as your variable value.

You can use `byte_extract` to create up to two separate variables in a rule concurrently. You can redefine a `byte_extract` variable any number of times; entering a new `byte_extract` keyword with the same variable name and a different variable definition overwrites the previous definition of that variable.

The following table describes the arguments required by the `byte_extract` keyword.

Table 31: Required byte_extract Arguments

Argument	Description
Bytes to Extract	The number of bytes to pick up from the packet. If you specify a number of bytes other than 1, 2, or 4, you must specify a Number Type (hexadecimal, octal, or decimal.)
Offset	The number of bytes into the payload to begin extracting data. You can specify -65535 to 65535 bytes. The offset counter starts at byte 0, so calculate the offset value by subtracting 1 from the number of bytes you want to count forward. For example, specify `7` to count forward 8 bytes. The rules engine counts forward from the beginning of the packet payload or, if you also specify **Relative**, after the last successful content match. Note that you can specify negative numbers only when you also specify **Relative**. You can use an existing `byte_math` result to specify the value for this argument.
Variable Name	The variable name to use in arguments for other detection keywords. You can specify an alphanumeric string that must begin with a letter.

To further define how the system locates the data to extract, you can use the arguments described in the following table.

Table 32: Additional Optional byte_extract Arguments

Argument	Description
Multiplier	A multiplier for the value extracted from the packet. You can specify 0 to 65535. If you do not specify a multiplier, the default value is 1.
Align	Rounds the extracted value to the nearest 2-byte or 4-byte boundary. When you also select **Multiplier**, the system applies the multiplier before the alignment.

Argument	Description
Relative	Makes **Offset** relative to the end of the last successful content match instead of the beginning of the payload.
Bitmask	Applies the specified hexadecimal bitmask using the AND operator to the bytes extracted from the Bytes to Extract argument. A bitmask can be 1 to 4 bytes. The result will be right-shifted by the number of bits equal to the number of trailing zeros in the mask.

You can specify only one of **DCE/RPC**, **Endian**, or **Number Type**.

To define how the `byte_extract` keyword calculates the bytes it tests, you can choose from the arguments in the following table. If you do not select a byte-ordering argument, the rules engine uses big endian byte order.

Table 33: Byte-Ordering byte_extract Arguments

Argument	Description
Big Endian	Processes data in big endian byte order, which is the default network byte order.
Little Endian	Processes data in little endian byte order.
DCE/RPC	Specifies a `byte_extract` keyword for traffic processed by the DCE/RPC preprocessor. The DCE/RPC preprocessor determines big endian or little endian byte order, and the **Number Type** and **Endian** arguments do not apply. When you enable this argument, you can also use `byte_extract` in conjunction with other specific DCE/RPC keywords.

You can specify a number type to read data as an ASCII string. To define how the system views string data in a packet, you can select one of the arguments in the following table.

Table 34: Number Type byte_extract arguments

Argument	Description
Hexadecimal String	Reads extracted string data in hexadecimal format.
Decimal String	Reads extracted string data in decimal format.
Octal String	Reads extracted string data in octal format.

For example, if the value for `byte_extract` is specified as the following:

- Bytes to Extract = 4
- Variable Name = var
- Offset = 8
- Relative = enabled

the rules engine reads the number described in the four bytes that appear 9 bytes away from (relative to) the last successful content match into a variable named `var`, which you can specify later in the rule as the value for certain keyword arguments.

The following table lists the keyword arguments where you can specify a variable defined in the `byte_extract` keyword.

Table 35: Arguments Accepting a byte_extract Variable

Keyword	Argument
content	Depth, Offset, Distance, Within
byte_jump	Offset
byte_test	Offset, Value
byte_math	RValue, Offset
isdataat	Offset

The byte_math Keyword

The `byte_math` keyword performs a mathematical operation on an extracted value and a specified value or existing variable, and stores the outcome in a new resulting variable. You can then use the resulting variable as an argument in other keywords.

You can use multiple `byte_math` keywords in a rule to perform multiple byte_math operations.

The following table describes the arguments required by the `byte_math` keyword.

Table 36: Required byte_math Arguments

Argument	Description
Bytes	The number of bytes to calculate from the packet. If used without DCE/RPC, the allowed values are 1 to 10: • Bytes can be 1 to 10 when the operator is +, -. *, or /. • Bytes can be 1 to 4 when the operator is << or >>. • If you specify a number of bytes other than 1, 2, or 4, you must specify a Number Type (hexadecimal, octal, or decimal.) If used with DCE/RPC, allowed values are 1, 2, and 4.
Offset	The number of bytes into the payload to start processing. The `offset` counter starts at byte 0, so calculate the `offset` value by subtracting 1 from the number of bytes you want to jump forward from the beginning of the packet payload or (if you specified Relative) from the last successful content match. You can specify -65535 to 65535 bytes. You can also specify the `byte_extract` variable here.

Argument	Description
Operator	+, -, *, /, <<, or >>
RValue	The value following the operator. This can be an unsigned integer or a variable passed from `byte_extract`.
Result Variable	The name of the variable into which the result of the `byte_math` calculation will be stored. You can use this variable as an argument in other keywords. This value is stored as an unsigned integer. This variable name: • Must use alphanumeric characters • Must not begin with a number • May include special characters supported by the Microsoft filename/variable name convention • Cannot consist entirely of special characters

The following table describes options you can use to define how the system interprets the values you specified for the required arguments.

Table 37: Additional Optional byte_math Arguments

Argument	Description
Relative	Makes the offset relative to the last pattern found in the last successful content match instead of the beginning of the payload.
Bitmask	Applies the specified hexadecimal bitmask using the AND operator to the bytes extracted from the Bytes argument. A bitmask can be 1 to 4 bytes. The result will be right-shifted by the number of bits equal to the number of trailing zeros in the mask.

You can specify only one of **DCE/RPC**, **Endian**, or **Number Type**.

If you want to define how the `byte_math` keyword calculates the bytes, you can choose from the arguments described in the following table. If you do not select a byte-ordering argument, the rules engine uses big endian byte order.

Table 38: Byte-Ordering byte_math Arguments

Argument	Description
Big Endian	Processes data in big endian byte order, which is the default network byte order.
Little Endian	Processes data in little endian byte order.

Argument	Description
DCE/RPC	Specifies a `byte_math` keyword for traffic processed by the DCE/RPC preprocessor. The DCE/RPC preprocessor determines big endian or little endian byte order, and the **Number Type** and **Endian** arguments do not apply. When you enable this argument, you can also use `byte_math` in conjunction with other specific DCE/RPC keywords.

Define how the system views string data in a packet by using one of the arguments in the following table.

Table 39: Number Type Arguments

Argument	Description
Hexadecimal String	Represents string data in hexadecimal format.
Decimal String	Represents string data in decimal format.
Octal String	Represents string data in octal format.

For example, if the values you set for `byte_math` are as follows:

- Bytes = 2
- Offset = 0
- Operator = *
- RValue = height
- Result Variable = area

the rules engine extracts the number described in the first two bytes in the packet and multiplies it by the RValue (which uses the existing variable, `height`) to create the new variable, `area`.

Table 40: Arguments Accepting a byte_math Variable

Keyword	Argument
byte_jump	Offset
byte_test	Offset, Value
byte_extract	Offset
isdataat	Offset

Overview: The pcre Keyword

The `pcre` keyword allows you to use Perl-compatible regular expressions (PCRE) to inspect packet payloads for specified content. You can use PCRE to avoid writing multiple rules to match slight variations of the same content.

Regular expressions are useful when searching for content that could be displayed in a variety of ways. The content may have different attributes that you want to account for in your attempt to locate it within a packet's payload.

Note that the regular expression syntax used in intrusion rules is a subset of the full regular expression library and varies in some ways from the syntax used in commands in the full library. When adding a `pcre` keyword using the intrusion rules editor, enter the full value in the following format:

```
!/pcre/ ismxAEGRBUIPHDMCKSY
```

where:

- ! is an optional negation (use this if you want to match patterns that **do not** match the regular expression).
- /pcre/ is a Perl-compatible regular expression.
- `ismxAEGRBUIPHDMCKSY` is any combination of modifier options.

Also note that you must escape the characters listed in the following table for the rules engine to interpret them correctly when you use them in a PCRE to search for specific content in a packet payload.

Table 41: Escaped PCRE Characters

You must escape...	with a backslash...	or Hex code...
# (hash mark)	\#	\x23
; (semicolon)	\;	\x3B
| (vertical bar)	|	\x7C
: (colon)	\:	\x3A

You can also use `m?regex?`, where `?` is a delimiter other than /. You may want to use this in situations where you need to match a forward slash within a regular expression and do not want to escape it with a backslash. For example, you might use `m?`regex`? ismxAEGRBUIPHDMCKSY` where regex is your Perl-compatible regular expression and `ismxAEGRBUIPHDMCKSY` is any combination of modifier options.

Tip Optionally, you can surround your Perl-compatible regular expression with quote characters, for example, pcre_expression or "pcre_expression". The option of using quotes accommodates experienced users accustomed to previous versions when quotes were required instead of optional. The intrusion rules editor does not display quotation marks when you display a rule after saving it.

pcre Syntax

The `pcre` keyword accepts standard Perl-compatible regular expression (PCRE) syntax. The following sections describe that syntax.

Tip While this section describes the basic syntax you may use for PCRE, you may want to consult an online reference or book dedicated to Perl and PCRE for more advanced information.

Metacharacters

Metacharacters are literal characters that have special meaning within regular expressions. When you use them within a regular expression, you must ��escape" them by preceding them with a backslash.

The following table describes the metacharacters you can use with PCRE and gives examples of each.

Table 42: PCRE Metacharacters

Metacharacter	Description	Example
`.`	Matches any character except newlines. If `s` is used as a modifying option, it also includes newline characters.	`abc.` matches `abcd`, `abc1`, `abc#`, and so on.
`*`	Matches zero or more occurrences of a character or expression.	`abc*` matches `abc`, `abcc`, `abccc`, `abcccccc`, and so on.
`?`	Matches zero or one occurrence of a character or expression.	`abc?` matches `abc`.
`+`	Matches one or more occurrences of a character or expression.	`abc+` matches `abc`, `abcc`, `abccc`, `abcccccc`, and so on.
`()`	Groups expressions.	`(abc)+` matches `abc`, `abcabc`, `abcabcabc` and so on.
`{}`	Specifies a limit for the number of matches for a character or expression. If you want to set a lower and upper limit, separate the lower limit and upper limit with a comma.	`a{4,6}` matches `aaaa`, `aaaaa`, or `aaaaaa`. `(ab){2}` matches `abab`.
`[]`	Allows you to define character classes, and matches any character or combination of characters described in the set.	`[abc123]` matches `a` or `b` or `c`, and so on.
`^`	Matches content at the beginning of a string. Also used for negation, if used within a character class.	`^in` matches the "in" in `info`, but not in `bin`. `[^a]` matches anything that does not contain `a`.
`$`	Matches content at the end of a string.	`ce$` matches the "`ce`" in `announce`, but not `cent`.
`|`	Indicates an OR expression.	`(MAILTO|HELP)` matches `MAILTO` or `HELP`.
`\`	Allows you to use metacharacters as actual characters and is also used to specify a predefined character class.	`\.` matches a period, `\*` matches an asterisk, `\\` matches a backslash and so on. `\d` matches the numeric characters, `\w` matches alphanumeric characters, and so on.

Character Classes

Character classes include alphabetic characters, numeric characters, alphanumeric characters, and white space characters. While you can create your own character classes within brackets, you can use the predefined classes as shortcuts for different types of character types. When used without additional qualifiers, a character class matches a single digit or character.

The following table describes and provides examples of the predefined character classes accepted by PCRE.

Table 43: PCRE Character Classes

Character Class	Description	Character Class Definition
\d	Matches a numeric character (“digit”).	[0-9]
\D	Matches anything that is not an numeric character.	[^0-9]
\w	Matches an alphanumeric character (“word”).	[a-zA-Z0-9_]
\W	Matches anything that is not an alphanumeric character.	[^a-zA-Z0-9_]
\s	Matches white space characters, including spaces, carriage returns, tabs, newlines, and form feeds.	[\r\t\n\f]
\S	Matches anything that is not a white space character.	[^ \r\t\n\f]

pcre Modifier Options

You can use modifying options after you specify regular expression syntax in the `pcre` keyword’s value. These modifiers perform Perl, PCRE, and Snort-specific processing functions. Modifiers always appear at the end of the PCRE value, and appear in the following format:

```
/pcre/ismxAEGRBUIPHDMCKSY
```

where `ismxAEGRBUPHMC` can include any of the modifying options that appear in the following tables.

Tip Optionally, you can surround the regular expression and any modifying options with quotes, for example, `“/pcre/ismxAEGRBUIPHDMCKSY”`. The option of using quotes accommodates experienced users accustomed to previous versions when quotes were required instead of optional. The intrusion rules editor does not display quotation marks when you display a rule after saving it.

The following table describes options you can use to perform Perl processing functions.

Table 44: Perl-Related Post Regular Expression Options

Option	Description
i	Makes the regular expression case-insensitive.
s	The dot character (.) describes all characters except the newline or `\n` character. You can use `"s"` as an option to override this and have the dot character match all characters, including the newline character.
m	By default, a string is treated as a single line of characters, and `^` and `$` match the beginning and ending of a specific string. When you use `"m"` as an option, `^` and `$` match content immediately before or after any newline character in the buffer, as well as at the beginning or end of the buffer.
x	Ignores white space data characters that may appear within the pattern, except when escaped (preceded by a backslash) or included inside a character class.

The following table describes the PCRE modifiers you can use after the regular expression.

Table 45: PCRE-Related Post Regular Expression Options

Option	Description
A	The pattern must match at the beginning of the string (same as using `^` in a regular expression).
E	Sets `$` to match only at the end of the subject string. (Without `E`, `$` also matches immediately before the final character if it is a newline, but not before any other newline characters).
G	By default, `*` `+` and `?` are "greedy," which means that if two or more matches are found, they will choose the longest match. Use the G character to change this so that these characters always choose the first match unless followed by a question mark character (`?`). For example, `*?` `+?` and `??` would be greedy in a construct using the G modifier, and any incidences of `*`, `+`, or `?` without the additional question mark will not be greedy.

The following table describes the Snort-specific modifiers that you can use after the regular expression.

.

Table 46: Snort-Specific Post Regular Expression Modifiers

Option	Description
R	Searches for matching content relative to the end of the last match found by the rules engine.
B	Searches for the content within data before it is decoded by a preprocessor (this option is similar to using the `Raw Data` argument with the `content` or `protected_content` keyword).
U	Searches for the content within the URI of a normalized HTTP request message decoded by the HTTP Inspect preprocessor. Note that you cannot use this option in combination with the `content` or `protected_content` keyword **HTTP URI** option to search the same content. Note that a pipelined HTTP request packet contains multiple URIs. A PCRE expression that includes the U option causes the rules engine to search for a content match only in the first URI in a pipelined HTTP request packet. To search all URIs in the packet, use the `content` or `protected_content` keyword with **HTTP URI** selected, either with or without an accompanying PCRE expression that uses the U option.
I	Searches for the content within the URI of a raw HTTP request message decoded by the HTTP Inspect preprocessor. Note that you cannot use this option in combination with the `content` or `protected_content` keyword **HTTP Raw URI** option to search the same content
P	Searches for the content within the body of a normalized HTTP request message decoded by the HTTP Inspect preprocessor.

Option	Description
H	Searches for the content within the header, excluding cookies, of an HTTP request or response message decoded by the HTTP Inspect preprocessor. Note that you cannot use this option in combination with the `content` or `protected_content` keyword **HTTP Header** option to search the same content.
D	Searches for the content within the header, excluding cookies, of a raw HTTP request or response message decoded by the HTTP Inspect preprocessor. Note that you cannot use this option in combination with the `content` or `protected_content` keyword **HTTP Raw Header** option to search the same content.
M	Searches for the content within the method field of a normalized HTTP request message decoded by the HTTP Inspect preprocessor; the method field identifies the action such as GET, PUT, CONNECT, and so on to take on the resource identified in the URI.
C	When the HTTP Inspect preprocessor **Inspect HTTP Cookies** option is enabled, searches for the normalized content within any cookie in an HTTP request header, and also within any set-cookie in an HTTP response header when the preprocessor **Inspect HTTP Responses** option is enabled. When **Inspect HTTP Cookies** is not enabled, searches the entire header, including the cookie or set-cookie data. Note the following: • Cookies included in the message body are treated as body content. • You cannot use this option in combination with the `content` or `protected_content` keyword **HTTP Cookie** option to search the same content. • The `Cookie:` and `Set-Cookie:` header names, leading spaces on the header line, and the `CRLF` that terminates the header line are inspected as part of the header and not as part of the cookie.
K	When the HTTP Inspect preprocessor **Inspect HTTP Cookies** option is enabled, searches for the raw content within any cookie in an HTTP request header, and also within any set-cookie in an HTTP response header when the preprocessor **Inspect HTTP Responses** option is enabled. When **Inspect HTTP Cookies** is not enabled, searches the entire header, including the cookie or set-cookie data. Note the following: • Cookies included in the message body are treated as body content. • You cannot use this option in combination with the `content` or `protected_content` keyword **HTTP Raw Cookie** option to search the same content. • The `Cookie:` and `Set-Cookie:` header names, leading spaces on the header line, and the `CRLF` that terminates the header line are inspected as part of the header and not as part of the cookie.
S	Searches the 3-digit status code in an HTTP response.
Y	Searches the textual description that accompanies the status code in an HTTP response.

Note Do not use the U option in combination with the R option. This could cause performance problems. Also, do not use the U option in combination with any other HTTP content option (I, P, H, D, M, C, K, S, or Y).

pcre Example Keyword Values

The following examples show values that you could enter for `pcre`, with descriptions of what each example would match.

- **`/feedback[(\d{0,1})]?\.cgi/U`**

This example searches packet payload for `feedback`, followed by zero or one numeric character, followed by `.cgi`, and located only in URI data.

This example would match:

- `feedback.cgi`
- `feedback1.cgi`
- `feedback2.cgi`
- `feedback3.cgi`

This example would **not** match:

- `feedbacka.cgi`
- `feedback11.cgi`
- `feedback21.cgi`
- `feedbackzb.cgi`

- **`/^ez(\w{3,5})\.cgi/iU`**

This example searches packet payload for `ez` at the beginning of a string, followed by a word of 3 to 5 letters, followed by `.cgi`. The search is case-insensitive and only searches URI data.

This example would match:

- `EZBoard.cgi`
- `ezman.cgi`
- `ezadmin.cgi`
- `EZAdmin.cgi`

This example would **not** match:

- `ezez.cgi`
- `fez.cgi`
- `abcezboard.cgi`
- `ezboardman.cgi`

- **/mail(file|seek)\.cgi/U**

This example searches packet payload for `mail`, followed by either `file` or `seek`, in URI data.

This example would match:

- `mailfile.cgi`
- `mailseek.cgi`

This example would **not** match:

- `MailFile.cgi`
- `mailfilefile.cgi`

- **m?http\\x3a\x2f\x2f.*(\n|\t)+?U**

This example searches packet payload for URI content for a tab or newline character in an HTTP request, after any number of characters. This example uses `m?`regex`?` to avoid using `http\:\/\/` in the expression. Note that the colon is preceded by a backslash.

This example would match:

- `http://www.example.com?scriptvar=x&othervar=\n\..\..`
- `http://www.example.com?scriptvar=\t`

This example would **not** match:

- `ftp://ftp.example.com?scriptvar=&othervar=\n\..\..`
- `http://www.example.com?scriptvar=|/bin/sh -i|`

- **m?http\\x3a\x2f\x2f.*=\|.*\|+?sU**

This example searches packet payload for a URL with any number of characters, including newlines, followed by an equal sign, and pipe characters that contain any number of characters or white space. This example uses `m?`regex`?` to avoid using `http\:\/\/` in the expression.

This example would match:

- `http://www.example.com?value=|/bin/sh/ -i|`
- `http://www.example.com?input=|cat /etc/passwd|`

This example would **not** match:

- `ftp://ftp.example.com?value=|/bin/sh/ -i|`
- `http://www.example.com?value=x&input?|cat /etc/passwd|`

- **/[0-9a-f]{2}\:[0-9a-f]{2}\:[0-9a-f]{2}\:[0-9a-f]{2}\:[0-9a-f]{2}\:[0-9a-f]{2}/i**

This example searches packet payload for any MAC address. Note that it escapes the colon characters with backslashes.

The metadata Keyword

You can use the `metadata` keyword to add your own descriptive information to a rule. You can also use the `metadata` keyword with `service` arguments to identify applications and ports in network traffic. You can use the information you add to organize or identify rules in ways that suit your needs, and you can search rules for information you add and for `service` arguments.

The system validates metadata based on the argument format:

key value

where *key* and *value* provide a combined description separated by a space. This is the format used by the Talos Intelligence Group for adding metadata to rules provided by Cisco.

Alternatively, you can also use the format:

key = *value*

For example, you could use the *key value* format to identify rules by author and date, using a category and sub-category as follows:

```
author SnortGuru_20050406
```

You can use multiple `metadata` keywords in a rule. You can also use commas to separate multiple *key value* arguments in a single `metadata` keyword, as seen in the following example:

```
author SnortGuru_20050406, revised_by SnortUser1_20050707,
revised_by SnortUser2_20061003,
revised_by SnortUser1_20070123
```

You are not limited to using a *key value* or *key=value* format; however, you should be aware of limitations resulting from validation based on these formats.

Restricted Characters to Avoid

Note the following character restrictions:

- Do not use a semicolon (;) or colon (:).
- The system interprets a comma as a separator for multiple *key value* or *key=value* arguments. For example:

  key value,*key value*,*key value*
- The system interprets the equal to (=) character or space character as separators between *key* and *value*. For example:

  key value

  key=value

All other characters are permitted.

Reserved Metadata to Avoid

Avoid using the following words in a `metadata` keyword, either as a single argument or as the *key* in a *key value* argument; these are reserved for use by Talos:

```
application
```

```
engine
impact_flag
os
policy
rule-type
rule-flushing
soid
```

Note Contact Support for assistance in adding restricted metadata to local rules that might not otherwise function as expected.

Impact Level 1

You can use the following reserved *key value* argument in a `metadata` keyword:

```
impact_flag red
```

This *key value* argument sets the impact flag to red (level 1) for a local rule you import or a custom rule you create using the intrusion rules editor.

Note that when Talos includes the `impact_flag red` argument in a rule provided by Cisco, Talos has determined that a packet triggering the rule indicates that the source or destination host is potentially compromised by a virus, trojan, or other piece of malicious software.

Service Metadata

The system detects applications running on the hosts in your network and inserts application protocol information into your network traffic; it does this regardless of the configuration of your discovery policy. You can use `metadata` keyword `service` arguments in a TCP or UDP rule to match application protocols and ports in your network traffic. You can combine one or more `service` application arguments in a rule with a single port argument.

Service Applications

You can use the `metadata` keyword with `service` as the *key* and an application as the *value* to match packets with the identified application protocol. For example, the following *key value* argument in a `metadata` keyword associates the rule with HTTP traffic:

```
service http
```

You can identify multiple applications separated by commas. For example:

```
service http, service smtp, service ftp
```

Caution Adaptive profiling **must** be enabled (its default state) for intrusion rules to use service metadata.

The following table describes the most common application values used with the `service` keyword.

Note Contact Support for assistance if you have difficulty identifying applications not in the table.

Table 47: service Values

Value	Description
`cvs`	Concurrent Versions System
`dcerpc`	Distributed Computing Environment/Remote Procedure Calls System
`dns`	Domain Name System
`finger`	Finger user information protocol
`ftp`	File Transfer Protocol
`ftp-data`	File Transfer Protocol (Data Channel)
`http`	Hypertext Transfer Protocol
`imap`	Internet Message Access Protocol
`isakmp`	Internet Security Association and Key Management Protocol
`mysql`	My Structured Query Language
`netbios-dgm`	NETBIOS Datagram Service
`netbios-ns`	NETBIOS Name Service
`netbios-ssn`	NETBIOS Session Service
`nntp`	Network News Transfer Protocol
`oracle`	Oracle Net Services
`shell`	OS Shell
`pop2`	Post Office Protocol, version 2
`pop3`	Post Office Protocol, version 3
`smtp`	Simple Mail Transfer Protocol
`snmp`	Simple Network Management Protocol
`ssh`	Secure Shell network protocol
`sunrpc`	Sun Remote Procedure Call Protocol
`telnet`	Telnet network protocol
`tftp`	Trivial File Transfer Protocol
`x11`	X Window System

Service Ports

You can use the `metadata` keyword with `service` as the *key* and a specified port argument as the *value* to define how the rule matches ports in combination with applications.

You can specify any of the port values in the table below, one value per rule.

Table 48: service Port Values

Value	Description
`else-ports` or `unknown`	The system applies the rule if either of the following conditions is met: • The packet application is known and matches the rule application. • The packet application is unknown and packet ports match the rule ports. The `else-ports` and `unknown` values produce the default behavior that the system uses when `service` specifies an application protocol with no port modifier.
`and-ports`	The system applies the rule if the packet application is known and matches the rule application, and the packet port matches the ports in the rule header. You cannot use `and-ports` in a rule that does not specify an application.
`or-ports`	The system applies the rule if any of the following conditions are met: • The packet application is known and matches the rule application. • The packet application is unknown and packet port matches the rule ports. • The packet application does not match the rule application and packet ports match the rule ports. • The rule does not specify an application and packet ports match the rule ports.

Note the following:

- You must include a `service` application argument with the `service and-ports` argument.
- If a rule specifies more than one of the values in the table above, the system applies the last one that appears in the rule.
- Port and application arguments can be in any order.

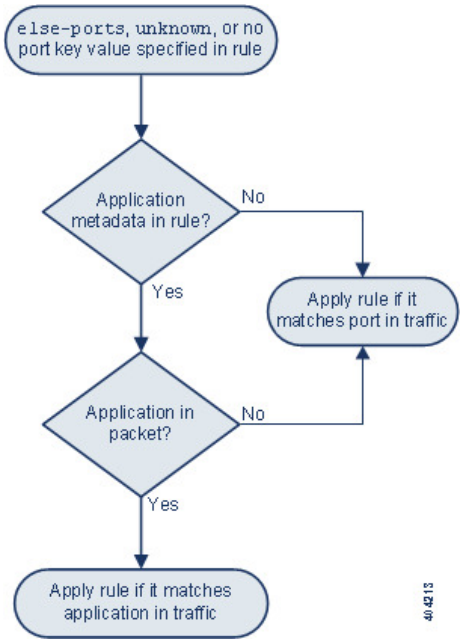
Except for the `and-ports` value, you can include a `service` port argument with or without one or more `service` application arguments. For example:

```
service or-ports, service http, service smtp
```

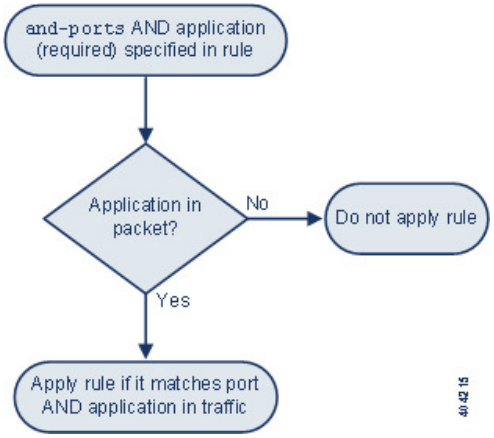
Applications and Ports in Traffic

The diagrams below illustrate the application and port combinations that intrusion rules support, and the results of applying these rule constraints to packet data.

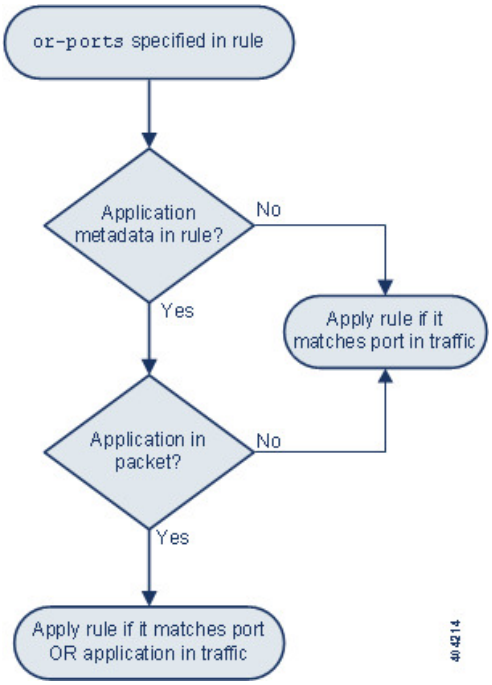
Host application protocol else source/destination ports:



Host application protocol and source/destination ports:



Host application protocol or source/destination ports:



Example Matches

The following sample rules using the `metadata` keyword with `service` arguments are shown with examples of data they match and do not match:

- `alert tcp any any -> any [80,8080] (metadata:service and-ports, service http, service smtp;)`

Example Matches	Example Non-Matches
• HTTP traffic over TCP port 80 • HTTP traffic over TCP port 8080 • SMTP traffic over TCP port 80 • SMTP traffic over TCP port 8080	• POP3 traffic on ports 80 or 8080 • Traffic of unknown application on ports 80 or 8080 • HTTP traffic on port 9999

- `alert tcp any any -> any [80,8080] (metadata:service or-ports, service http;)`

Example Matches	Example Non-Matches
• HTTP traffic on any port • SMTP traffic on port 80 • SMTP traffic on port 8080 • Traffic of unknown application on port 80 and 8080	• Non-HTTP and non-SMTP traffic on ports other than 80 or 8080

- Any of the following rules:
  - `alert tcp any any -> any [80,8080] metadata:service else-ports, service http;)`
  - `alert tcp any any -> any [80,8080] metadata:service unknown, service http;)`
  - `alert tcp any any -> any [80,8080] metadata:service http;)`

Example Matches	Example Non-Matches
• HTTP traffic on any port • port 80 if packet application is unknown • port 8080 if packet application is unknown	• SMTP traffic on ports 80 or 8080 • POP3 traffic on ports 80 or 8080

Metadata Search Guidelines

To search for rules that use the `metadata` keyword, select the `metadata` keyword on the rules Search page and, optionally, type any portion of the metadata. For example, you can type:

- `search` to display all rules where you have used `search` for *key*.
- `search http` to display all rules where you have used `search` for *key* and `http` for *value*.
- `author snortguru` to display all rules where you have used `author` for *key* and `SnortGuru` for *value*.
- `author s` to display all rules where you have used `author` for *key* and any terms such as `SnortGuru` or `SnortUser1` or `SnortUser2` for *value*.

Tip When you search for both *key* and *value* , use the same connecting operator (equal to [=] or a space character) in searches that is used in the *key value* argument in the rule; searches return different results depending on whether you follow *key* with equal to (=) or a space character.

Note that regardless of the format you use to add metadata, the system interprets your metadata search term as all or part of a *key value* or *key*=*value* argument. For example, the following would be valid metadata that does not follow a *key value* or *key*=*value* format:

```
ab cd ef gh
```

However, the system would interpret each space in the example as a separator between a *key* and *value* . Thus, you could successfully locate a rule containing the example metadata using any of the following searches for juxtaposed and single terms:

```
cd ef
ef gh
ef
```

but you would not locate the rule using the following search, which the system would interpret as a single *key value* argument:

```
ab ef
```

IP Header Values

You can use keywords to identify possible attacks or security policy violations in the IP headers of packets.

fragbits

The `fragbits` keyword inspects the fragment and reserved bits in the IP header. You can check each packet for the Reserved Bit, the More Fragments bit, and the Don't Fragment bit in any combination.

Table 49: Fragbits Argument Values

Argument	Description
R	Reserved bit
M	More Fragments bit
D	Don’t Fragment bit

To further refine a rule using the `fragbits` keyword, you can specify any operator described in the following table after the argument value in the rule.

Table 50: Fragbit Operators

Operator	Description
plus sign (**+**)	The packet must match against all specified bits.
asterisk (*****)	The packet can match against any of the specified bits.
exclamation point (**!**)	The packet meets the criteria if none of the specified bits are set.

For example, to generate an event against packets that have the Reserved Bit set (and possibly any other bits), use `R+` as the `fragbits` value.

id

The `id` keyword tests the IP header fragment identification field against the value you specify in the keyword’s argument. Some denial-of-service tools and scanners set this field to a specific number that is easy to detect. For example, in SID 630, which detects a Synscan portscan, the `id` value is set to `39426`, the static value used as the ID number in packets transmitted by the scanner.

Note `id` argument values must be numeric.

ipopts

The `IPopts` keyword allows you to search packets for specified IP header options. The following table lists the available argument values.

Table 51: IPoption Arguments

Argument	Description
rr	record route
eol	end of list
nop	no operation
ts	time stamp
sec	IP security option
lsrr	loose source routing
ssrr	strict source routing
satid	stream identifier

Analysts most frequently watch for strict and loose source routing because these options may be an indication of a spoofed source IP address.

ip_proto

The `ip_proto` keyword allows you to identify packets with the IP protocol specified as the keyword's value. You can specify the IP protocols as a number, 0 through 255. You can combine these numbers with the following operators: `<`, `>`, or `!`. For example, to inspect traffic with any protocol that is not ICMP, use `!1` as a value to the `ip_proto` keyword. You can also use the `ip_proto` keyword multiple times in a single rule; note, however, that the rules engine interprets multiple instances of the keyword as having a Boolean AND relationship. For example, if you create a rule containing `ip_proto:!3; ip_proto:!6`, the rule ignores traffic using the GGP protocol AND the TCP protocol.

tos

Some networks use the type of service (ToS) value to set precedence for packets traveling on that network. The `tos` keyword allows you to test the packet's IP header ToS value against the value you specify as the keyword's argument. Rules using the `tos` keyword will trigger on packets whose ToS is set to the specified value and that meet the rest of the criteria set forth in the rule.

Note Argument values for `tos` must be numeric.

The ToS field has been deprecated in the IP header protocol and replaced with the Differentiated Services Code Point (DSCP) field.

ttl

A packet's time-to-live (ttl) value indicates how many hops it can make before it is dropped. You can use the `ttl` keyword to test the packet's IP header ttl value against the value, or range of values, you specify as the keyword's argument. It may be helpful to set the `ttl` keyword parameter to a low value such as 0 or 1, as low time-to-live values are sometimes indicative of a traceroute or intrusion evasion attempt. (Note, though, that

the appropriate value for this keyword depends on your managed device placement and network topology.) Use syntax as follows:

- Use an integer from 0 to 255 to set a specific value for the TTL value. You can also precede the value with an equal (=) sign (for example, you can specify `5` or `=5`).
- Use a hyphen (-) to specify a range of TTL values (for example, `0-2` specifies all values 0 through 2, `-5` specifies all values 0 through 5, and `5-` specifies all values 5 through 255).
- Use the greater than (>) sign to specify TTL values greater than a specific value (for example, `>3` specifies all values greater than 3).
- Use the greater than and equal to signs (>=) to specify TTL values greater than or equal to a specific value (for example, `>=3` specifies all values greater than or equal to 3).
- Use the less than (<) sign to specify TTL values less than a specific value (for example, `<3` specifies all values less than 3).
- Use the less than and equal to signs (<=) to specify TTL values less than or equal to a specific value (for example, `<=3` specifies all values less than or equal to 3).

ICMP Header Values

The system supports keywords that you can use to identify attacks and security policy violations in the headers of ICMP packets. Note, however, that predefined rules exist that detect most ICMP types and codes. Consider enabling an existing rule or creating a local rule based on an existing rule; you may be able to find a rule that meets your needs more quickly than if you build an ICMP rule from scratch.

icmp_id and icmp_seq

The ICMP identification and sequence numbers help associate ICMP replies with ICMP requests. In normal traffic, these values are dynamically assigned to packets. Some covert channel and Distributed Denial of Server (DDoS) programs use static ICMP ID and sequence values. The following keywords allow you to identify ICMP packets with static values.

Keyword	Definition
icmp_id	Inspects an ICMP echo request or reply packet's ICMP ID number. Use a numeric value that corresponds with the ICMP ID number as the argument for the `icmp_id` keyword.
icmp_seq	The `icmp_seq` keyword inspects an ICMP echo request or reply packet's ICMP sequence. Use a numeric value that corresponds with the ICMP sequence number as the argument for the `icmp_seq` keyword.

itype

Use the `itype` keyword to look for packets with specific ICMP message type values. You can specify either a valid ICMP type value or an invalid ICMP type value to test for different types of traffic. For example, attackers may set ICMP type values out of range to cause denial of service and flooding attacks.

You can specify a range for the `itype` argument value using less than (<) and greater than (>).

For example:

- <35
- >36
- 3<>55

icode

ICMP messages sometimes include a code value that provides details when a destination is unreachable.

You can use the `icode` keyword to identify packets with specific ICMP code values. You can choose to specify either a valid ICMP code value or an invalid ICMP code value to test for different types of traffic.

You can specify a range for the `icode` argument value using less than (<) and greater than (>).

For example:

- to find values less than 35, specify `<35`.
- to find values greater than 36, specify `>36`.
- to find values between 3 and 55, specify `3<>55`.

Tip You can use the `icode` and `itype` keywords together to identify traffic that matches both. For example, to identify ICMP traffic that contains an ICMP Destination Unreachable code type with an ICMP Port Unreachable code type, specify an `itype` keyword with a value of 3 (for Destination Unreachable) and an `icode` keyword with a value of 3 (for Port Unreachable).

TCP Header Values and Stream Size

The system supports keywords that are designed to identify attacks attempted using TCP headers of packets and TCP stream size.

ack

You can use the `ack` keyword to compare a value against a packet's TCP acknowledgment number. The rule triggers if a packet's TCP acknowledgment number matches the value specified for the `ack` keyword.

Argument values for `ack` must be numeric.

flags

You can use the `flags` keyword to specify any combination of TCP flags that, when set in an inspected packet, cause the rule to trigger.

Note In situations where you would traditionally use `A+` as the value for `flags`, you should instead use the `flow` keyword with a value of `established`. Generally, you should use the `flow` keyword with a value of `stateless` when using flags to ensure that all combinations of flags are detected.

You can either check for or ignore the values described in the following table for the `flag` keyword.

Table 52: flag Arguments

Argument	TCP Flag
Ack	Acknowledges data.
Psh	Data should be sent in this packet.
Syn	A new connection.
Urg	Packet contains urgent data.
Fin	A closed connection.
Rst	An aborted connection.
CWR	An ECN congestion window has been reduced. This was formerly the R1 argument, which is still supported for backward compatibility.
ECE	ECN echo. This was formerly the R2 argument, which is still supported for backward compatibility.

When using the `flags` keyword, you can use an operator to indicate how the system performs matches against multiple flags. The following table describes these operators.

Table 53: Operators Used with flags

Operator	Description	Example
all	The packet must contain all specified flags.	Select `Urg` and `all` to specify that a packet must contain the Urgent flag and may contain any other flags.
any	The packet can contain any of the specified flags.	Select `Ack`, `Psh`, and `any` to specify that either or both the `Ack` and `Psh` flags must be set to trigger the rule, and that other flags may also be set on a packet.
not	The packet must **not** contain the specified flag set.	Select `Urg` and `not` to specify that the Urgent flag is **not** set on packets that trigger this rule.

flow

You can use the `flow` keyword to select packets for inspection by a rule based on session characteristics. The `flow` keyword allows you to specify the direction of the traffic flow to which a rule applies, applying rules to either the client flow or server flow. To specify how the `flow` keyword inspects your packets, you can set the direction of traffic you want analyzed, the state of packets inspected, and whether the packets are part of a rebuilt stream.

Stateful inspection of packets occurs when rules are processed. If you want a TCP rule to ignore stateless traffic (traffic without an established session context), you must add the `flow` keyword to the rule and select the **Established** argument for the keyword. If you want a UDP rule to ignore stateless traffic, you must add the `flow` keyword to the rule and select either the **Established** argument or a directional argument, or both. This causes the TCP or UDP rule to perform stateful inspection of a packet.

When you add a directional argument, the rules engine inspects only those packets that have an established state with a flow that matches the direction specified. For example, if you add the `flow` keyword with the `established` argument and the `From Client` argument to a rule that triggers when a TCP or UDP connection is detected, the rules engine only inspects packets that are sent from the client.

Tip For maximum performance, always include a `flow` keyword in a TCP rule or a UDP session rule.

The following table describes the stream-related arguments you can specify for the `flow` keyword:

Table 54: State-Related flow Arguments

Argument	Description
Established	Triggers on established connections.
Stateless	Triggers regardless of the state of the stream processor.

The following table describes the directional options you can specify for the `flow` keyword:

Table 55: flow Directional Arguments

Argument	Description
To Client	Triggers on server responses.
To Server	Triggers on client responses.
From Client	Triggers on client responses.
From Server	Triggers on server responses.

Notice that `From Server` and `To Client` perform the same function, as do `To Server` and `From Client`. These options exist to add context and readability to the rule. For example, if you create a rule designed to detect an attack from a server to a client, use `From Server`. But, if you create a rule designed to detect an attack from the client to the server, use `From Client`.

The following table describes the stream-related arguments you can specify for the `flow` keyword:

Table 56: Stream-Related flow Arguments

Argument	Description
Ignore Stream Traffic	Does not trigger on rebuilt stream packets.
Only Stream Traffic	Triggers only on rebuilt stream packets.

For example, you can use `To Server, Established, Only Stream Traffic` as the value for the `flow` keyword to detect traffic, traveling from a client to the server in an established session, that has been reassembled by the stream preprocessor.

seq

The `seq` keyword allows you to specify a static sequence number value. Packets whose sequence number matches the specified argument trigger the rule containing the keyword. While this keyword is used rarely, it is helpful in identifying attacks and network scans that use generated packets with static sequence numbers.

window

You can use the `window` keyword to specify the TCP window size you are interested in. A rule containing this keyword triggers whenever it encounters a packet with the specified TCP window size. While this keyword is used rarely, it is helpful in identifying attacks and network scans that use generated packets with static TCP window sizes.

stream_size

You can use the `stream_size` keyword in conjunction with the stream preprocessor to determine the size in bytes of a TCP stream, using the format:

```
direction,operator,bytes
```

where bytes is number of bytes. You must separate each option in the argument with a comma (,).

The following table describes the case-insensitive directional options you can specify for the `stream_size` keyword:

Table 57: stream_size Keyword Directional Arguments

Argument	Description
client	triggers on a stream from the client matching the specified stream size.
server	triggers on a stream from the server matching the specified stream size.
both	triggers on traffic from the client and traffic from the server both matching the specified stream size. For example, the argument `both, >, 200` would trigger when traffic from the client is greater than 200 bytes AND traffic from the server is greater than 200 bytes.
either	triggers on traffic from either the client or the server matching the specified stream size, whichever occurs first. For example, the argument `either, >, 200` would trigger when traffic from the client is greater than 200 bytes OR traffic from the server is greater than 200 bytes.

The following table describes the operators you can use with the `stream_size` keyword:

Table 58: stream_size Keyword Argument Operators

Operator	Description
=	equal to
!=	not equal to
>	greater than

Operator	Description
<	less than
>=	greater than or equal to
<=	less than or equal to

For example, you could use `client, >=, 5001216` as the argument for the `stream_size` keyword to detect a TCP stream traveling from a client to a server and greater than or equal to 5001216 bytes.

The stream_reassembly Keyword

You can use the `stream_reassemble` keyword to enable or disable TCP stream reassembly for a single connection when inspected traffic on the connection matches the conditions of the rule. Optionally, you can use this keyword multiple times in a rule.

Use the following syntax to enable or disable stream reassembly:

```
enable|disable, server|client|both, option, option
```

The following table describes the optional arguments you can use with the `stream_reassemble` keyword.

Table 59: stream_reassemble Optional Arguments

Argument	Description
noalert	Generate no events regardless of any other detection options specified in the rule.
fastpath	Ignore the rest of the connection traffic when there is a match.

For example, the following rule disables TCP client-side stream reassembly without generating an event on the connection where a 200 OK status code is detected in an HTTP response:

```
alert tcp any 80 -> any any (flow:to_client, established; content: “200 OK”;
stream_reassemble:disable, client, noalert
```

SSL Keywords

You can use SSL rule keywords to invoke the Secure Sockets Layer (SSL) preprocessor and extract information about SSL version and session state from packets in an encrypted session.

When a client and server communicate to establish an encrypted session using SSL or Transport Layer Security (TLS), they exchange handshake messages. Although the data transmitted in the session is encrypted, the handshake messages are not.

The SSL preprocessor extracts state and version information from specific handshake fields. Two fields within the handshake indicate the version of SSL or TLS used to encrypt the session and the stage of the handshake.

ssl_state

The `ssl_state` keyword can be used to match against state information for an encrypted session. To check for two or more SSL versions used simultaneously, use multiple `ssl_version` keywords in a rule.

When a rule uses the `ssl_state` keyword, the rules engine invokes the SSL preprocessor to check traffic for SSL state information.

For example, to detect an attacker's attempt to cause a buffer overflow on a server by sending a `ClientHello` message with an overly long challenge length and too much data, you could use the `ssl_state` keyword with `client_hello` as an argument then check for abnormally large packets.

Use a comma-separated list to specify multiple arguments for the SSL state. When you list multiple arguments, the system evaluates them using the OR operator. For example, if you specify `client_hello` and `server_hello` as arguments, the system evaluates the rule against traffic that has a `client_hello` OR a `server_hello`.

You can also negate any argument; for example:

```
!client_hello, !unknown
```

To ensure the connection has reached each of a set of states, multiple rules using the ssl_state rule option should be used. The `ssl_state` keyword takes the following identifiers as arguments:

Table 60: `ssl_state` Arguments

Argument	Purpose
`client_hello`	Matches against a handshake message with `ClientHello` as the message type, where the client requests an encrypted session.
`server_hello`	Matches against a handshake message with `ServerHello` as the message type, where the server responds to the client's request for an encrypted session.
`client_keyx`	Matches against a handshake message with `ClientKeyExchange` as the message type, where the client transmits a key to the server to confirm receipt of a key from the server.
`server_keyx`	Matches against a handshake message with `ServerKeyExchange` as the message type, where the client transmits a key to the server to confirm receipt of a key from the server.
`unknown`	Matches against any handshake message type.

ssl_version

The `ssl_version` keyword can be used to match against version information for an encrypted session. When a rule uses the `ssl_version` keyword, the rules engine invokes the SSL preprocessor to check traffic for SSL version information.

For example, if you know there is a buffer overflow vulnerability in SSL version 2, you could use the `ssl_version` keyword with the `sslv2` argument to identify traffic using that version of SSL.

Use a comma-separated list to specify multiple arguments for the SSL version. When you list multiple arguments, the system evaluates them using the OR operator. For example, if you wanted to identify any encrypted traffic that was not using SSLv2, you could add `ssl_version:ssl_v3,tls1.0,tls1.1,tls1.2` to a rule. The rule would evaluate any traffic using SSL Version 3, TLS Version 1.0, TLS Version 1.1, or TLS Version 1.2.

The `ssl_version` keyword takes the following SSL/TLS version identifiers as arguments:

Table 61: `ssl_version` Arguments

Argument	Purpose
`sslv2`	Matches against traffic encoded using Secure Sockets Layer (SSL) Version 2.
`sslv3`	Matches against traffic encoded using Secure Sockets Layer (SSL) Version 3.
`tls1.0`	Matches against traffic encoded using Transport Layer Security (TLS) Version 1.0.
`tls1.1`	Matches against traffic encoded using Transport Layer Security (TLS) Version 1.1.
`tls1.2`	Matches against traffic encoded using Transport Layer Security (TLS) Version 1.2.

The appid Keyword

You can use the `appid` keyword to identify the application protocol, client application, or web application in a packet. For example, you could target a specific application that you know is susceptible to a specific vulnerability.

Within the `appid` keyword of an intrusion rule, click **Configure AppID** to select one or more applications that you want to detect.

Browsing the Available Applications

When you first start to build the condition, the **Available Applications** list is unconstrained and displays every application the system detects, 100 per page:

- To page through the applications, click the arrows underneath the list.
- To display a pop-up window with summary information about the application's characteristics, as well as Internet search links that you can follow, click **Information** ( ) next to an application.

Using Application Filters

To help you find the applications you want to match, you can constrain the **Available Applications** list in the following ways:

- To search for applications, click the **Search by name** prompt above the list, then type a name. The list updates as you type to display matching applications.
- To constrain the applications by applying a filter, use the **Application Filters** list. The **Available Applications** list updates as you apply filters. For your convenience, the system uses an **Unlock icon** to mark applications that the system can identify only in decrypted traffic—not encrypted or unencrypted.

Note If you select one or more filters in the Application Filters list and also search the **Available Applications** list, your selections and the search-filtered **Available Applications** list are combined using an AND operation.

Selecting Applications

To select a single application, select it and click **Add to Rule**. To select all applications in the current constrained view, right-click and select **Select All**.

Application Layer Protocol Values

Although preprocessors perform most of the normalization and inspection of application layer protocol values, you can continue to inspect application layer values using various preprocessor options.

The RPC Keyword

The `rpc` keyword identifies Open Network Computing Remote Procedure Call (ONC RPC) services in TCP or UDP packets. This allows you to detect attempts to identify the RPC programs on a host. Intruders can use an RPC portmapper to determine if any of the RPC services running on your network can be exploited. They can also attempt to access other ports running RPC without using portmapper. The following table lists the arguments that the `rpc` keyword accepts.

Table 62: rpc Keyword Arguments

Argument	Description
application	The RPC application number
procedure	The RPC procedure invoked
version	The RPC version

To specify the arguments for the `rpc` keyword, use the following syntax:

```
application,procedure,version
```

where application is the RPC application number, procedure is the RPC procedure number, and version is the RPC version number. You must specify all arguments for the `rpc` keyword — if you are not able to specify one of the arguments, replace it with an asterisk (`*`).

For example, to search for RPC portmapper (which is the RPC application indicated by the number 100000), with any procedure or version, use `100000,*,*` as the arguments.

The ASN.1 Keyword

The `asn1` keyword allows you to decode a packet or a portion of a packet, looking for various malicious encodings.

The following table describes the arguments for the `asn1` keyword.

Table 63: asn.1 Keyword Arguments

Argument	Description
Bitstring Overflow	Detects invalid, remotely exploitable bitstring encodings.
Double Overflow	Detects a double ASCII encoding that is larger than a standard buffer. This is known to be an exploitable function in Microsoft Windows, but it is unknown at this time which services may be exploitable.

Argument	Description
Oversize Length	Detects ASN.1 type lengths greater than the supplied argument. For example, if you set the Oversize Length to 500, any ASN.1 type greater than 500 triggers the rule.
Absolute Offset	Sets an absolute offset from the beginning of the packet payload. (Remember that the offset counter starts at byte 0.) For example, if you want to decode SNMP packets, set Absolute Offset to 0 and do not set a Relative Offset. Absolute Offset may be positive or negative.
Relative Offset	This is the relative offset from the last successful content match, `pcre`, or `byte_jump`. To decode an ASN.1 sequence right after the content "foo", set Relative Offset to 0, and do not set an Absolute Offset. Relative Offset may be positive or negative. (Remember that the offset counter starts at 0.)

For example, there is a known vulnerability in the Microsoft ASN.1 Library that creates a buffer overflow, allowing an attacker to exploit the condition with a specially crafted authentication packet. When the system decodes the asn.1 data, exploit code in the packet could execute on the host with system-level privileges or could cause a DoS condition. The following rule uses the `asn1` keyword to detect attempts to exploit this vulnerability:

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 445
(flow:to_server, established; content:"|FF|SMB|73|";
nocase; offset:4; depth:5;
asn1:bitstring_overflow,double_overflow,oversize_length 100,
relative_offset 54;)
```

The above rule generates an event against TCP traffic traveling from any IP address defined in the $EXTERNAL_NET variable, from any port, to any IP address defined in the $HOME_NET variable using port 445. In addition, it only executes the rule on established TCP connections to servers. The rule then tests for specific content in specific locations. Finally, the rule uses the `asn1` keyword to detect bitstring encodings and double ASCII encodings and to identify asn.1 type lengths over 100 bytes in length starting 55 bytes from the end of the last successful content match. (Remember that the `offset` counter starts at byte 0.)

The urilen Keyword

You can use the `urilen` keyword in conjunction with the HTTP Inspect preprocessor to inspect HTTP traffic for URIs of a specific length, less than a maximum length, greater than a minimum length, or within a specified range.

After the HTTP Inspect preprocessor normalizes and inspects the packet, the rules engine evaluates the packet against the rule and determines whether the URI matches the length condition specified by the `urilen` keyword. You can use this keyword to detect exploits that attempt to take advantage of URI length vulnerabilities, for example, by creating a buffer overflow that allows the attacker to cause a DoS condition or execute code on the host with system-level privileges.

Note the following when using the `urilen` keyword in a rule:

- In practice, you always use the `urilen` keyword in combination with the `flow:established` keyword and one or more other keywords.
- The rule protocol is always TCP.
- Target ports are always HTTP ports.

You specify the URI length using a decimal number of bytes, less than (<) and greater than (>).

For example:

- specify `5` to detect a URI 5 bytes long.
- specify `< 5` (separated by one space character) to detect a URI less than 5 bytes long.
- specify `> 5` (separated by one space character) to detect a URI greater than 5 bytes long.
- specify `3 <> 5` (with one space character before and after `<>`) to detect a URI between 3 and 5 bytes long inclusive.

For example, there is a known vulnerability in Novell's server monitoring and diagnostics utility iMonitor version 2.4, which comes with eDirectory version 8.8. A packet containing an excessively long URI creates a buffer overflow, allowing an attacker to exploit the condition with a specially crafted packet that could execute on the host with system-level privileges or could cause a DoS condition. The following rule uses the `urilen` keyword to detect attempts to exploit this vulnerability:

```
alert tcp $EXTERNAL_NET any -> $HOME_NET $HTTP_PORTS
(msg:"EXPLOIT eDirectory 8.8 Long URI iMonitor buffer
overflow attempt"; flow:to_server,established;
urilen:> 8192; uricontent:"/nds/"; nocase;
classtype:attempted-admin; sid:x; rev:1;)
```

The above rule generates an event against TCP traffic traveling from any IP address defined in the $EXTERNAL_NET variable, from any port, to any IP address defined in the $HOME_NET variable using the ports defined in the $HTTP_PORTS variable. In addition, packets are evaluated against the rule only on established TCP connections to servers. The rule uses the `urilen` keyword to detect any URI over 8192 bytes in length. Finally, the rule searches the URI for the specific case-insensitive content `/nds/`.

DCE/RPC Keywords

The three DCE/RPC keywords described in the following table allow you to monitor DCE/RPC session traffic for exploits. When the system processes rules with these keywords, it invokes the DCE/RPC preprocessor.

Table 64: DCE/RPC Keywords

Use...	In this way...	To detect...
`dce_iface`	alone	packets identifying a specific DCE/RPC service
`dce_opnum`	preceded by `dce_iface`	packets identifying specific DCE/RPC service operations
`dce_stub_data`	preceded by `dce_iface` + `dce_opnum`	stub data defining a specific operation request or response

Note in the table that you should always precede `dce_opnum` with `dce_iface`, and you should always precede `dce_stub_data` with `dce_iface` + `dce_opnum`.

You can also use these DCE/RPC keywords in combination with other rule keywords. Note that for DCE/RPC rules, you use the `byte_jump`, `byte_test`, `and byte_extract` keywords with their **DCE/RPC** arguments selected.

Cisco recommends that you include at least one `content` keyword in rules that include DCE/RPC keywords to ensure that the rules engine uses the fast pattern matcher, which increases processing speed and improves

performance. Note that the rules engine uses the fast pattern matcher when a rule includes at least one `content` keyword, regardless of whether you enable the `content` keyword **Use Fast Pattern Matcher** argument.

You can use the DCE/RPC version and adjoining header information as the matching content in the following cases:

- the rule does not include another `content` keyword
- the rule contains another `content` keyword, but the DCE/RPC version and adjoining information represent a more unique pattern than the other content

  For example, the DCE/RPC version and adjoining information are more likely to be unique than a single byte of content.

You should end qualifying rules with one of the following version and adjoining information content matches:

- For connection-oriented DCE/RPC rules, use the content `|05 00 00|` (for major version 05, minor version 00, and the request PDU (protocol data unit) type 00).
- For connectionless DCE/RPC rules, use the content `|04 00|` (for version 04, and the request PDU type 00).

In either case, position the `content` keyword for version and adjoining information as the last keyword in the rule to invoke the fast pattern matcher without repeating processing already completed by the DCE/RPC preprocessor. Note that placing the `content` keyword at the end of the rule applies to version content used as a device to invoke the fast pattern matcher, and not necessarily to other content matches in the rule.

dce_iface

You can use the `dce_iface` keyword to identify a specific DCE/RPC service.

Optionally, you can also use `dce_iface` in combination with the `dce_opnum` and `dce_stub_data` keywords to further limit the DCE/RPC traffic to inspect.

A fixed, sixteen-byte Universally Unique Identifier (UUID) identifies the application interface assigned to each DCE/RPC service. For example, the UUID 4b324fc8-670-01d3-1278-5a47bf6ee188 identifies the DCE/RPC lanmanserver service, also known as the srvsvc service, which provides numerous management functions for sharing peer-to-peer printers, files, and SMB named pipes. The DCE/RPC preprocessor uses the UUID and associated header values to track DCE/RPC sessions.

The interface UUID is comprised of five hexadecimal strings separated by hyphens:

```
<4hexbytes>-<2hexbytes>-<2hexbytes>-<2hexbytes>-<6hexbytes>
```

You specify the interface by entering the entire UUID including hyphens, as seen in the following UUID for the netlogon interface:

```
12345678-1234-abcd-ef00-01234567cffb
```

Note that you must specify the first three strings in the UUID in big endian byte order. Although published interface listings and protocol analyzers typically display UUIDs in the correct byte order, you might encounter a need to rearrange the UUID byte order before entering it. Consider the following messenger service UUID shown as it might sometimes be displayed in raw ASCII text with the first three strings in little endian byte order:

```
f8 91 7b 5a 00 ff d0 11 a9 b2 00 c0 4f b6 e6 fc
```

You would specify the same UUID for the `dce_iface` keyword by inserting hyphens and putting the first three strings in big endian byte order as follows:

```
5a7b91f8-ff00-11d0-a9b2-00c04fb6e6fc
```

Although a DCE/RPC session can include requests to multiple interfaces, you should include only one `dce_iface` keyword in a rule. Create additional rules to detect additional interfaces.

DCE/RPC application interfaces also have interface version numbers. You can optionally specify an interface version with an operator indicating that the version equals, does not equal, is less than, or greater than the specified value.

Both connection-oriented and connectionless DCE/RPC can be fragmented in addition to any TCP segmentation or IP fragmentation. Typically, it is not useful to associate any DCE/RPC fragment other than the first with the specified interface, and doing so may result in a large number of false positives. However, for flexibility you can optionally evaluate all fragments against the specified interface.

The following table summarizes the `dce_iface` keyword arguments.

Table 65: `dce_iface` Arguments

Argument	Description
Interface UUID	The UUID, including hyphens, that identifies the application interface of the specific service that you want to detect in DCE/RPC traffic. Any request associated with the specified interface would match the interface UUID.
Version	Optionally, the application interface version number 0 to 65535 and an operator indicating whether to detect a version greater than (>), less than (<), equal to (=), or not equal to (!) the specified value.
All Fragments	Optionally, enable to match against the interface in all associated DCE/RPC fragments and, if specified, on the interface version. This argument is disabled by default, indicating that the keyword matches only if the first fragment or the entire unfragmented packet is associated with the specified interface. Note that enabling this argument may result in false positives.

The dce_opnum Keyword

You can use the `dce_opnum` keyword in conjunction with the DCE/RPC preprocessor to detect packets that identify one or more specific operations that a DCE/RPC service provides.

Client function calls request specific service functions, which are referred to in DCE/RPC specifications as *operations*. An operation number (opnum) identifies a specific operation in the DCE/RPC header. It is likely that an exploit would target a specific operation.

For example, the UUID 12345678-1234-abcd-ef00-01234567cffb identifies the interface for the netlogon service, which provides several dozen different operations. One of these is operation 6, the NetrServerPasswordSet operation.

You should precede a `dce_opnum` keyword with a `dce_iface` keyword to identify the service for the operation.

You can specify a single decimal value 0 to 65535 for a specific operation, a range of operations separated by a hyphen, or a comma-separated list of operations and ranges in any order.

Any of the following examples would specify valid netlogon operation numbers:

```
15
15-18
15, 18-20
15, 20-22, 17
15, 18-20, 22, 24-26
```

The dce_stub_data Keyword

You can use the `dce_stub_data` keyword in conjunction with the DCE/RPC preprocessor to specify that the rules engine should start inspection at the beginning of the stub data, regardless of any other rule options. Packet payload rule options that follow the `dce_stub_data` keyword are applied relative to the stub data buffer.

DCE/RPC stub data provides the interface between a client procedure call and the DCE/RPC run-time system, the mechanism that provides the routines and services central to DCE/RPC. DCE/RPC exploits are identified in the stub data portion of the DCE/RPC packet. Because stub data is associated with a specific operation or function call, you should always precede `dce_stub_data` with `dce_iface` and `dce_opnum` to identify the related service and operation.

The `dce_stub_data` keyword has no arguments.

SIP Keywords

Four SIP keywords allow you to monitor SIP session traffic for exploits.

Note that the SIP protocol is vulnerable to denial of service (DoS) attacks. Rules addressing these attacks can benefit from rate-based attack prevention.

The sip_header Keyword

You can use the `sip_header` keyword to start inspection at the beginning of the extracted SIP request or response header and restrict inspection to header fields.

The `sip_header` keyword has no arguments.

The following example rule fragment points to the SIP header and matches the CSeq header field:

```
alert udp any any -> any 5060 ( sip_header; content:"CSeq"; )
```

The sip_body Keyword

You can use the `sip_body` keyword to start inspection at the beginning of the extracted SIP request or response message body and restrict inspection to the message body.

The `sip_body` keyword has no arguments.

The following example rule fragment points to the SIP message body and matches a specific IP address in the c (connection information) field in extracted SDP data:

```
alert udp any any -> any 5060 ( sip_body; content:"c=IN 192.168.12.14"; )
```

Note that rules are not limited to searching for SDP content. The SIP preprocessor extracts the entire message body and makes it available to the rules engine.

The sip_method Keyword

A *method* field in each SIP request identifies the purpose of the request. You can use the `sip_method` keyword to test SIP requests for specific methods. Separate multiple methods with commas.

You can specify any of the following currently defined SIP methods:

```
ack, benotify, bye, cancel, do, info, invite, join, message, notify, options, prack,
publish, quath, refer, register, service, sprack, subscribe, unsubscribe, update
```

Methods are case-insensitive. You can separate multiple methods with commas.

Because new SIP methods might be defined in the future, you can also specify a custom method, that is, a method that is not a currently defined SIP method. Accepted field values are defined in RFC 2616, which allows all characters except control characters and separators such as `=`, `(`, and `)`. See RFC 2616 for the complete list of excluded separators. When the system encounters a specified custom method in traffic, it will inspect the packet header but not the message.

The system supports up to 32 methods, including the 21 currently defined methods and an additional 11 methods. The system ignores any undefined methods that you might configure. Note that the 32 total methods includes methods specified using the **Methods to Check** SIP preprocessor option.

You can specify only one method when you use negation. For example:

```
!invite
```

Note, however, that multiple `sip_method` keywords in a rule are linked with an **AND** operation. For example, to test for all extracted methods except `invite` and `cancel`, you would use two negated `sip_method` keywords:

```
sip_method: !invite
sip_method: !cancel
```

Cisco recommends that you include at least one `content` keyword in rules that include the `sip_method` keyword to ensure that the rules engine uses the fast pattern matcher, which increases processing speed and improves performance. Note that the rules engine uses the fast pattern matcher when a rule includes at least one `content` keyword, regardless of whether you enable the `content` keyword **Use Fast Pattern Matcher** argument.

The sip_stat_code Keyword

A three-digit status code in each SIP response indicates the outcome of the requested action. You can use the `sip_stat_code` keyword to test SIP responses for specific status codes.

You can specify a one-digit response-type number 1-9, a specific three-digit number 100-999, or a comma-separated list of any combination of either. A list matches if any single number in the list matches the code in the SIP response.

The following table describes the SIP status code values you can specify.

Table 66: `sip_stat_code` Values

To detect...	Specify...	For example...	Detects...
a specific status code	the three-digit status code	189	189
any three-digit code that begins with a specified single digit	the single digit	1	1xx; that is, 100, 101, 102, and so on

To detect...	Specify...	For example...	Detects...
a list of values	any comma-separated combination of specific codes and single digits	222, 3	222 plus 300, 301, 302, and so on

Note also that the rules engine does not use the fast pattern matcher to search for the value specify using the `sip_stat_code` keyword, regardless of whether your rule includes a `content` keyword.

GTP Keywords

Three GSRP Tunneling Protocol (GTP) keywords allow you to inspect the GTP command channel for GTP version, message type, and information elements. You cannot use GTP keywords in combination with other intrusion rule keywords such as `content` or `byte_jump`. You **must** use the `gtp_version` keyword in each rule that uses the `gtp_info` or `gtp_type` keyword.

The gtp_version Keyword

You can use the `gtp_version` keyword to inspect GTP control messages for GTP version 0, 1, or 2.

Because different GTP versions define different message types and information elements, you must use `gtp_version` when you use the `gtp_type` or `gtp_info` keyword. You can specify the value 0, 1, or 2.

The gtp_type Keyword

Each GTP message is identified by a message type, which is comprised of both a numeric value and a string. You can use the `gtp_type` keyword to inspect traffic for specific GTP message types. Because different GTP versions define different message types and information elements, you must also use `gtp_version` when you use the `gtp_type` or `gtp_info` keyword.

You can specify a defined decimal value for a message type, a defined string, or a comma-separated list of either or both in any combination, as seen in the following example:

```
10, 11, echo_request
```

The system uses an OR operation to match each value or string that you list. The order in which you list values and strings does not matter. Any single value or string in the list matches the keyword. You receive an error if you attempt to save a rule that includes an unrecognized string or an out-of-range value.

Note in the table that different GTP versions sometimes use different values for the same message type. For example, the `sgsn_context_request` message type has a value of 50 in GTPv0 and GTPv1, but a value of 130 in GTPv2.

The `gtp_type` keyword matches different values depending on the version number in the packet. In the example above, the keyword matches the message type value 50 in a GTPv0 or GTPv1 packet and the value 130 in a GTPv2 packet. The keyword does not match a packet when the message type value in the packet is not a known value for the version specified in the packet.

If you specify an integer for the message type, the keyword matches if the message type in the keyword matches the value in the GTP packet, regardless of the version specified in the packet.

The following table lists the defined values and strings recognized by the system for each GTP message type.

Table 67: GTP Message Types

Value	Version 0	Version 1	Version 2
1	echo_request	echo_request	echo_request
2	echo_response	echo_response	echo_response
3	version_not_supported	version_not_supported	version_not_supported
4	node_alive_request	node_alive_request	N/A
5	node_alive_response	node_alive_response	N/A
6	redirection_request	redirection_request	N/A
7	redirection_response	redirection_response	N/A
16	create_pdp_context_request	create_pdp_context_request	N/A
17	create_pdp_context_response	create_pdp_context_response	N/A
18	update_pdp_context_request	update_pdp_context_request	N/A
19	update_pdp_context_response	update_pdp_context_response	N/A
20	delete_pdp_context_request	delete_pdp_context_request	N/A
21	delete_pdp_context_response	delete_pdp_context_response	N/A
22	create_aa_pdp_context_request	init_pdp_context_activation_request	N/A
23	create_aa_pdp_context_response	init_pdp_context_activation_response	N/A
24	delete_aa_pdp_context_request	N/A	N/A
25	delete_aa_pdp_context_response	N/A	N/A
26	error_indication	error_indication	N/A
27	pdu_notification_request	pdu_notification_request	N/A
28	pdu_notification_response	pdu_notification_response	N/A
29	pdu_notification_reject_request	pdu_notification_reject_request	N/A
30	pdu_notification_reject_response	pdu_notification_reject_response	N/A
31	N/A	supported_ext_header_notification	N/A
32	send_routing_info_request	send_routing_info_request	create_session_request
33	send_routing_info_response	send_routing_info_response	create_session_response
34	failure_report_request	failure_report_request	modify_bearer_request
35	failure_report_response	failure_report_response	modify_bearer_response

Value	Version 0	Version 1	Version 2
36	note_ms_present_request	note_ms_present_request	delete_session_request
37	note_ms_present_response	note_ms_present_response	delete_session_response
38	N/A	N/A	change_notification_request
39	N/A	N/A	change_notification_response
48	identification_request	identification_request	N/A
49	identification_response	identification_response	N/A
50	sgsn_context_request	sgsn_context_request	N/A
51	sgsn_context_response	sgsn_context_response	N/A
52	sgsn_context_ack	sgsn_context_ack	N/A
53	N/A	forward_relocation_request	N/A
54	N/A	forward_relocation_response	N/A
55	N/A	forward_relocation_complete	N/A
56	N/A	relocation_cancel_request	N/A
57	N/A	relocation_cancel_response	N/A
58	N/A	forward_srns_contex	N/A
59	N/A	forward_relocation_complete_ack	N/A
60	N/A	forward_srns_contex_ack	N/A
64	N/A	N/A	modify_bearer_command
65	N/A	N/A	modify_bearer_failure_indication
66	N/A	N/A	delete_bearer_command
67	N/A	N/A	delete_bearer_failure_indication
68	N/A	N/A	bearer_resource_command
69	N/A	N/A	bearer_resource_failure_indication
70	N/A	ran_info_relay	downlink_failure_indication
71	N/A	N/A	trace_session_activation
72	N/A	N/A	trace_session_deactivation
73	N/A	N/A	stop_paging_indication
95	N/A	N/A	create_bearer_request

Value	Version 0	Version 1	Version 2
96	N/A	mbms_notification_request	create_bearer_response
97	N/A	mbms_notification_response	update_bearer_request
98	N/A	mbms_notification_reject_request	update_bearer_response
99	N/A	mbms_notification_reject_response	delete_bearer_request
100	N/A	create_mbms_context_request	delete_bearer_response
101	N/A	create_mbms_context_response	delete_pdn_request
102	N/A	update_mbms_context_request	delete_pdn_response
103	N/A	update_mbms_context_response	N/A
104	N/A	delete_mbms_context_request	N/A
105	N/A	delete_mbms_context_response	N/A
112	N/A	mbms_register_request	N/A
113	N/A	mbms_register_response	N/A
114	N/A	mbms_deregister_request	N/A
115	N/A	mbms_deregister_response	N/A
116	N/A	mbms_session_start_request	N/A
117	N/A	mbms_session_start_response	N/A
118	N/A	mbms_session_stop_request	N/A
119	N/A	mbms_session_stop_response	N/A
120	N/A	mbms_session_update_request	N/A
121	N/A	mbms_session_update_response	N/A
128	N/A	ms_info_change_request	identification_request
129	N/A	ms_info_change_response	identification_response
130	N/A	N/A	sgsn_context_request
131	N/A	N/A	sgsn_context_response
132	N/A	N/A	sgsn_context_ack
133	N/A	N/A	forward_relocation_request
134	N/A	N/A	forward_relocation_response
135	N/A	N/A	forward_relocation_complete

Value	Version 0	Version 1	Version 2
136	N/A	N/A	forward_relocation_complete_ack
137	N/A	N/A	forward_access
138	N/A	N/A	forward_access_ack
139	N/A	N/A	relocation_cancel_request
140	N/A	N/A	relocation_cancel_response
141	N/A	N/A	configuration_transfer_tunnel
149	N/A	N/A	detach
150	N/A	N/A	detach_ack
151	N/A	N/A	cs_paging
152	N/A	N/A	ran_info_relay
153	N/A	N/A	alert_mme
154	N/A	N/A	alert_mme_ack
155	N/A	N/A	ue_activity
156	N/A	N/A	ue_activity_ack
160	N/A	N/A	create_forward_tunnel_request
161	N/A	N/A	create_forward_tunnel_response
162	N/A	N/A	suspend
163	N/A	N/A	suspend_ack
164	N/A	N/A	resume
165	N/A	N/A	resume_ack
166	N/A	N/A	create_indirect_forward_tunnel_request
167	N/A	N/A	create_indirect_forward_tunnel_response
168	N/A	N/A	delete_indirect_forward_tunnel_request
169	N/A	N/A	delete_indirect_forward_tunnel_response
170	N/A	N/A	release_access_bearer_request
171	N/A	N/A	release_access_bearer_response
176	N/A	N/A	downlink_data
177	N/A	N/A	downlink_data_ack

Value	Version 0	Version 1	Version 2
179	N/A	N/A	pgw_restart
180	N/A	N/A	pgw_restart_ack
200	N/A	N/A	update_pdn_request
201	N/A	N/A	update_pdn_response
211	N/A	N/A	modify_access_bearer_request
212	N/A	N/A	modify_access_bearer_response
231	N/A	N/A	mbms_session_start_request
232	N/A	N/A	mbms_session_start_response
233	N/A	N/A	mbms_session_update_request
234	N/A	N/A	mbms_session_update_response
235	N/A	N/A	mbms_session_stop_request
236	N/A	N/A	mbms_session_stop_response
240	data_record_transfer_request	data_record_transfer_request	N/A
241	data_record_transfer_response	data_record_transfer_response	N/A
254	N/A	end_marker	N/A
255	pdu	pdu	N/A

The gtp_info Keyword

A GTP message can include multiple information elements, each of which is identified by both a defined numeric value and a defined string. You can use the `gtp_info` keyword to start inspection at the beginning of a specified information element, and restrict inspection to the specified information element. Because different GTP versions define different message types and information elements, you must also use `gtp_version` when you use this keyword.

You can specify either the defined decimal value or the defined string for an information element. You can specify a single value or string, and you can use multiple `gtp_info` keywords in a rule to inspect multiple information elements.

When a message includes multiple information elements of the same type, all are inspected for a match. When information elements occur in an invalid order, only the last instance is inspected.

Note that different GTP versions sometimes use different values for the same information element. For example, the `cause` information element has a value of 1 in GTPv0 and GTPv1, but a value of 2 in GTPv2.

The `gtp_info` keyword matches different values depending on the version number in the packet. In the example above, the keyword matches the information element value 1 in a GTPv0 or GTPv1 packet and the value 2 in a GTPv2 packet. The keyword does not match a packet when the information element value in the packet is not a known value for the version specified in the packet.

If you specify an integer for the information element, the keyword matches if the message type in the keyword matches the value in the GTP packet, regardless of the version specified in the packet.

The following table lists the values and strings recognized by the system for each GTP information element.

Table 68: GTP Information Elements

Value	Version 0	Version 1	Version 2
1	cause	cause	imsi
2	imsi	imsi	cause
3	rai	rai	recovery
4	tlli	tlli	N/A
5	p_tmsi	p_tmsi	N/A
6	qos	N/A	N/A
8	recording_required	recording_required	N/A
9	authentication	authentication	N/A
11	map_cause	map_cause	N/A
12	p_tmsi_sig	p_tmsi_sig	N/A
13	ms_validated	ms_validated	N/A
14	recovery	recovery	N/A
15	selection_mode	selection_mode	N/A
16	flow_label_data_1	teid_1	N/A
17	flow_label_signalling	teid_control	N/A
18	flow_label_data_2	teid_2	N/A
19	ms_unreachable	teardown_ind	N/A
20	N/A	nsapi	N/A
21	N/A	ranap	N/A
22	N/A	rab_context	N/A
23	N/A	radio_priority_sms	N/A
24	N/A	radio_priority	N/A
25	N/A	packet_flow_id	N/A
26	N/A	charging_char	N/A

Value	Version 0	Version 1	Version 2
27	N/A	trace_ref	N/A
28	N/A	trace_type	N/A
29	N/A	ms_unreachable	N/A
71	N/A	N/A	apn
72	N/A	N/A	ambr
73	N/A	N/A	ebi
74	N/A	N/A	ip_addr
75	N/A	N/A	mei
76	N/A	N/A	msisdn
77	N/A	N/A	indication
78	N/A	N/A	pco
79	N/A	N/A	paa
80	N/A	N/A	bearer_qos
80	N/A	N/A	flow_qos
82	N/A	N/A	rat_type
83	N/A	N/A	serving_network
84	N/A	N/A	bearer_tft
85	N/A	N/A	tad
86	N/A	N/A	uli
87	N/A	N/A	f_teid
88	N/A	N/A	tmsi
89	N/A	N/A	cn_id
90	N/A	N/A	s103pdf
91	N/A	N/A	s1udf
92	N/A	N/A	delay_value
93	N/A	N/A	bearer_context
94	N/A	N/A	charging_id
95	N/A	N/A	charging_char

Value	Version 0	Version 1	Version 2
96	N/A	N/A	trace_info
97	N/A	N/A	bearer_flag
99	N/A	N/A	pdn_type
100	N/A	N/A	pti
101	N/A	N/A	drx_parameter
103	N/A	N/A	gsm_key_tri
104	N/A	N/A	umts_key_cipher_quin
105	N/A	N/A	gsm_key_cipher_quin
106	N/A	N/A	umts_key_quin
107	N/A	N/A	eps_quad
108	N/A	N/A	umts_key_quad_quin
109	N/A	N/A	pdn_connection
110	N/A	N/A	pdn_number
111	N/A	N/A	p_tmsi
112	N/A	N/A	p_tmsi_sig
113	N/A	N/A	hop_counter
114	N/A	N/A	ue_time_zone
115	N/A	N/A	trace_ref
116	N/A	N/A	complete_request_msg
117	N/A	N/A	guti
118	N/A	N/A	f_container
119	N/A	N/A	f_cause
120	N/A	N/A	plmn_id
121	N/A	N/A	target_id
123	N/A	N/A	packet_flow_id
124	N/A	N/A	rab_contex
125	N/A	N/A	src_rnc_pdcp
126	N/A	N/A	udp_src_port

Value	Version 0	Version 1	Version 2
127	charge_id	charge_id	apn_restriction
128	end_user_address	end_user_address	selection_mode
129	mm_context	mm_context	src_id
130	pdp_context	pdp_context	N/A
131	apn	apn	change_report_action
132	protocol_config	protocol_config	fq_csid
133	gsn	gsn	channel
134	msisdn	msisdn	emlpp_pri
135	N/A	qos	node_type
136	N/A	authentication_qu	fqdn
137	N/A	tft	ti
138	N/A	target_id	mbms_session_duration
139	N/A	utran_trans	mbms_service_area
140	N/A	rab_setup	mbms_session_id
141	N/A	ext_header	mbms_flow_id
142	N/A	trigger_id	mbms_ip_multicast
143	N/A	omc_id	mbms_distribution_ack
144	N/A	ran_trans	rfsp_index
145	N/A	pdp_context_pri	uci
146	N/A	addi_rab_setup	csg_info
147	N/A	sgsn_number	csg_id
148	N/A	common_flag	cmi
149	N/A	apn_restriction	service_indicator
150	N/A	radio_priority_lcs	detach_type
151	N/A	rat_type	ldn
152	N/A	user_loc_info	node_feature
153	N/A	ms_time_zone	mbms_time_to_transfer
154	N/A	imei_sv	throttling

Value	Version 0	Version 1	Version 2
155	N/A	camel	arp
156	N/A	mbms_ue_context	epc_timer
157	N/A	tmp_mobile_group_id	signalling_priority_indication
158	N/A	rim_routing_addr	tmgi
159	N/A	mbms_config	mm_srvcc
160	N/A	mbms_service_area	flags_srvcc
161	N/A	src_rnc_pdcp	nmbr
162	N/A	addi_trace_info	N/A
163	N/A	hop_counter	N/A
164	N/A	plmn_id	N/A
165	N/A	mbms_session_id	N/A
166	N/A	mbms_2g3g_indicator	N/A
167	N/A	enhanced_nsapi	N/A
168	N/A	mbms_session_duration	N/A
169	N/A	addi_mbms_trace_info	N/A
170	N/A	mbms_session_repetition_num	N/A
171	N/A	mbms_time_to_data	N/A
173	N/A	bss	N/A
174	N/A	cell_id	N/A
175	N/A	pdu_num	N/A
177	N/A	mbms_bearer_capab	N/A
178	N/A	rim_routing_disc	N/A
179	N/A	list_pfc	N/A
180	N/A	ps_xid	N/A
181	N/A	ms_info_change_report	N/A
182	N/A	direct_tunnel_flags	N/A
183	N/A	correlation_id	N/A
184	N/A	bearer_control_mode	N/A

Value	Version 0	Version 1	Version 2
185	N/A	mbms_flow_id	N/A
186	N/A	mbms_ip_multicast	N/A
187	N/A	mbms_distribution_ack	N/A
188	N/A	reliable_inter_rat_handover	N/A
189	N/A	rfsp_index	N/A
190	N/A	fqdn	N/A
191	N/A	evolved_allocation1	N/A
192	N/A	evolved_allocation2	N/A
193	N/A	extended_flags	N/A
194	N/A	uci	N/A
195	N/A	csg_info	N/A
196	N/A	csg_id	N/A
197	N/A	cmi	N/A
198	N/A	apn_ambr	N/A
199	N/A	ue_network	N/A
200	N/A	ue_ambr	N/A
201	N/A	apn_ambr_nsapi	N/A
202	N/A	ggsn_backoff_timer	N/A
203	N/A	signalling_priority_indication	N/A
204	N/A	signalling_priority_indication_nsapi	N/A
205	N/A	high_bitrate	N/A
206	N/A	max_mbr	N/A
251	charging_gateway_addr	charging_gateway_addr	N/A
255	private_extension	private_extension	private_extension

SCADA Keywords

The rules engine uses Modbus, DNP3, CIP, and S7Commplus rules to access certain protocol fields.

Modbus Keywords

You can use Modbus keywords alone or in combination with other keywords such as `content` and `byte_jump`.

modbus_data

You can use the `modbus_data` keyword to point to the beginning of the Data field in a Modbus request or response.

modbus_func

You can use the `modbus_func` keyword to match against the Function Code field in a Modbus application layer request or response header. You can specify either a single defined decimal value or a single defined string for a Modbus function code.

The following table lists the defined values and strings recognized by the system for Modbus function codes.

Table 69: Modbus Function Codes

Value	String
1	read_coils
2	read_discrete_inputs
3	read_holding_registers
4	read_input_registers
5	write_single_coil
6	write_single_register
7	read_exception_status
8	diagnostics
11	get_comm_event_counter
12	get_comm_event_log
15	write_multiple_coils
16	write_multiple_registers
17	report_slave_id
20	read_file_record
21	write_file_record
22	mask_write_register
23	read_write_multiple_registers
24	read_fifo_queue

Value	String
43	encapsulated_interface_transport

modbus_unit

You can use the `modbus_unit` keyword to match a single decimal value against the Unit ID field in a Modbus request or response header.

DNP3 Keywords

You can use DNP3 keywords alone or in combination with other keywords such as `content` and `byte_jump`.

dnp3_data

You can use the `dnp3_data` keyword to point to the beginning of reassembled DNP3 application layer fragments.

The DNP3 preprocessor reassembles link layer frames into application layer fragments. The `dnp3_data` keyword points to the beginning of each application layer fragment; other rule options can match against the reassembled data within fragments without separating the data and adding checksums every 16 bytes.

dnp3_func

You can use the `dnp3_func` keyword to match against the Function Code field in a DNP3 application layer request or response header. You can specify either a single defined decimal value or a single defined string for a DNP3 function code.

The following table lists the defined values and strings recognized by the system for DNP3 function codes.

Table 70: DNP3 Function Codes

Value	String
0	confirm
1	read
2	write
3	select
4	operate
5	direct_operate
6	direct_operate_nr
7	immed_freeze
8	immed_freeze_nr
9	freeze_clear
10	freeze_clear_nr

Value	String
11	freeze_at_time
12	freeze_at_time_nr
13	cold_restart
14	warm_restart
15	initialize_data
16	initialize_appl
17	start_appl
18	stop_appl
19	save_config
20	enable_unsolicited
21	disable_unsolicited
22	assign_class
23	delay_measure
24	record_current_time
25	open_file
26	close_file
27	delete_file
28	get_file_info
29	authenticate_file
30	abort_file
31	activate_config
32	authenticate_req
33	authenticate_err
129	response
130	unsolicited_response
131	authenticate_resp

dnp3_ind

You can use the `dnp3_ind` keyword to match against flags in the Internal Indications field in a DNP3 application layer response header.

You can specify the string for a single known flag or a comma-separated list of flags, as seen in the following example:

```
class_1_events, class_2_events
```

When you specify multiple flags, the keyword matches against any flag in the list. To detect a combination of flags, use the `dnp3_ind` keyword multiple times in a rule.

The following list provides the string syntax recognized by the system for defined DNP3 internal indications flags.

```
class_1_events
class_2_events
class_3_events
need_time
local_control
device_trouble
device_restart
no_func_code_support
object_unknown
parameter_error
event_buffer_overflow
already_executing
config_corrupt
reserved_2
reserved_1
```

dnp3_obj

You can use the `dnp3_obj` keyword to match against DNP3 object headers in a request or response.

DNP3 data is comprised of a series of DNP3 objects of different types such as analog input, binary input, and so on. Each type is identified with a *group* such as analog input group, binary input group, and so on, each of which can be identified by a decimal value. The objects in each group are further identified by an *object variation* such as 16-bit integers, 32-bit integers, short floating point, and so on, each of which specifies the data format of the object. Each type of object variation can also be identified by a decimal value.

You identify object headers by specifying the decimal number for the type of object header group and the decimal number for the type of object variation. The combination of the two defines a specific type of DNP3 object.

CIP and ENIP Keywords

You can use the following keywords alone or in combination to create custom intrusion rules that identify attacks against CIP and ENIP traffic detected by the CIP preprocessor. For configurable keywords, specify a single integer within the allowed range. See The CIP Preprocessor, on page 300 for more information.

Table 71:

This keyword...	Matches against...	Range
`cip_attribute`	the Object Class/Instance Attribute field in a CIP message. Specify a single defined integer value.	0 - 65535

This keyword...	Matches against...	Range
`cip_class`	the Object Class field in a CIP message. Specify a single defined integer value.	0 - 65535
`cip_conn_path_class`	the Object Class in Connection Path. Specify a single integer value.	0 - 65535
`cip_instance`	the Instance ID field in a CIP message. Specify a single integer value.	0 - 4284927295
`cip_req`	the service request message.	N/A
`cip_rsp`	the service response message.	N/A
`cip_service`	the Service field in a CIP service request message. Specify a single integer value.	0 - 127
`cip_status`	the Status field in a CIP service response message. Specify a single integer value.	0 - 255
`enip_command`	the Command Code in EthNet/IP header. Specify a single integer value.	0 - 65535
`enip_req`	the EthNet/IP request message.	N/A
`enip_rsp`	the EthNet/IP response message.	N/A

S7Commplus Keywords

You can use the S7Commplus keywords alone or in combination to create custom intrusion rules that identify attacks against traffic detected by the S7Commplus preprocessor. For configurable keywords, specify a single known value or a single integer within the allowed range. See The S7Commplus Preprocessor, on page 304 for more information.

Note the following:

- Multiple S7commplus keywords in the same rule are AND-ed.
- Using multiple `s7commplus_func` or `s7commplus_opcode` keywords in the same rule negates the rule and it will never match traffic. To search for multiple values with these keywords, create multiple rules.

s7commplus_content

Before using a `content` or `protected_content` keyword in an S7Commplus intrusion rule, use the `s7commplus_content` keyword to position the cursor to the beginning of the packet payload. See The content and protected_content Keywords, on page 109 for more information.

s7commplus_func

Use the `s7commplus_func` keyword to match against one of the following values in an S7Commplus header:

- explore
- createobject

- deleteobject
- setvariable
- getlink
- setmultivar
- getmultivar
- beginsequence
- endsequence
- invoke
- getvarsubstr
- 0x0 through 0xFFF

  Note that numeric expressions allow for additional values.

s7commplus_opcode

Use the `s7commplus_opcode` keyword to match against one of the following values in an S7Commplus header:

- request
- response
- notification
- response2
- 0x0 through 0xFF

  Note that numeric expressions allow for additional values.

Packet Characteristics

You can write rules that only generate events against packets with specific packet characteristics.

dsize

The `dsize` keyword tests the packet payload size. With it, you can use the greater than and less than operators (< and >) to specify a range of values. You can use the following syntax to specify ranges:

```
>number_of_bytes
<number_of_bytes
number_of_bytes<>number_of_bytes
```

For example, to indicate a packet size greater than 400 bytes, use `>400` as the `dtype` value. To indicate a packet size of less than 500 bytes, use `<500`. To specify that the rule trigger against any packet between 400 and 500 bytes inclusive, use `400<>500`.

Caution The `dsize` keyword tests packets before they are decoded by any preprocessors.

isdataat

The `isdataat` keyword instructs the rules engine to verify that data resides at a specific location in the payload.

The following table lists the arguments you can use with the `isdataat` keyword.

Table 72: isdataat Arguments

Argument	Type	Description
Offset	Required	The specific location in the payload. For example, to test that data appears at byte 50 in the packet payload, you would specify `50` as the offset value. A `!` modifier negates the results of the `isdataat` test; it alerts if a certain amount of data is not present within the payload. You can also use an existing `byte_extract` variable or `byte_math` result to specify the value for this argument.
Relative	Optional	Makes the location relative to the last successful content match. If you specify a relative location, note that the counter starts at byte 0, so calculate the location by subtracting 1 from the number of bytes you want to move forward from the last successful content match. For example, to specify that the data must appear at the ninth byte after the last successful content match, you would specify a relative offset of `8`.
Raw Data	Optional	Specifies that the data is located in the original packet payload before decoding or application layer normalization by any Firepower System preprocessor. You can use this argument with **Relative** if the previous content match was in the raw packet data.

For example, in a rule searching for the content `foo`, if the value for `isdataat` is specified as the following:

- Offset = !10
- Relative = enabled

The system alerts if the rules engine does not detect 10 bytes after `foo` before the payload ends.

sameip

The `sameip` keyword tests that a packet's source and destination IP addresses are the same. It does not take an argument.

fragoffset

The `fragoffset` keyword tests the offset of a fragmented packet. This is useful because some exploits (such as WinNuke denial-of-service attacks) use hand-generated packet fragments that have specific offsets.

For example, to test whether the offset of a fragmented packet is 31337 bytes, specify `31337` as the `fragoffset` value.

You can use the following operators when specifying arguments for the `fragoffset` keyword.

Table 73: fragoffset Keyword Argument Operators

Operator	Description
!	not
>	greater than
<	less than

Note that you cannot use the not (`!`) operator in combination with < or >.

cvs

The `cvs` keyword tests Concurrent Versions System (CVS) traffic for malformed CVS entries. An attacker can use a malformed entry to force a heap overflow and execute malicious code on the CVS server. This keyword can be used to identify attacks against two known CVS vulnerabilities: CVE-2004-0396 (CVS 1.11.x up to 1.11.15, and 1.12.x up to 1.12.7) and CVS-2004-0414 (CVS 1.12.x through 1.12.8, and 1.11.x through 1.11.16). The `cvs` keyword checks for a well-formed entry, and generates alerts when a malformed entry is detected.

Your rule should include the ports where CVS runs. In addition, any ports where traffic may occur should be added to the list of ports for stream reassembly in your TCP policies so state can be maintained for CVS sessions. The TCP ports 2401 (`pserver`) and 514 (`rsh`) are included in the list of client ports where stream reassembly occurs. However, note that if your server runs as an `xinetd` server (i.e., pserver), it can run on any TCP port. Add any non-standard ports to the stream reassembly **Client Ports** list.

Active Response Keywords

The **resp** and **react** keywords provide two approaches to initiating active responses. An intrusion rule that contains either keyword initiates a single active response when a packet triggers the rule. Active response keywords initiate active responses to close TCP connections in response to triggered TCP rules or UDP sessions in response to triggered UDP rules. Active responses are not intended to take the place of a firewall for a number of reasons, including that an attacker may have chosen to ignore or circumvent active responses.

Active responses are supported in inline, including routed or transparent, deployments. For example, in response to the `react` keyword in an inline deployment, the system can insert a TCP reset (RST) packet directly into the traffic for each end of the connection, which normally should close the connection. Active responses are not supported or suited for passive deployments.

Because active responses can be routed back, the system does not allow TCP resets to initiate TCP resets; this prevents an unending sequence of active responses. The system also does not allow ICMP unreachable packets to initiate ICMP unreachable packets in keeping with standard practice.

You can configure the TCP stream preprocessor to detect additional traffic on a TCP connection after an intrusion rule has triggered an active response. When the preprocessor detects additional traffic, it sends additional active responses up to a specified maximum to both ends of the connection or session.

The resp Keyword

You can use the `resp` keyword to actively respond to TCP connections or UDP sessions, depending on whether you specify the TCP or UDP protocol in the rule header.

Keyword arguments allow you to specify the packet direction and whether to use TCP reset (RST) packets or ICMP unreachable packets as active responses.

You can use any of the TCP reset or ICMP unreachable arguments to close TCP connections. You should use only ICMP unreachable arguments to close UDP sessions.

Different TCP reset arguments also allow you to target active responses to the packet source, destination, or both. All ICMP unreachable arguments target the packet source and allow you to specify whether to use an ICMP network, host, or port unreachable packet, or all three.

The following table lists the arguments you can use with the `resp` keyword to specify exactly what you want the system to do when the rule triggers.

Table 74: resp Arguments

Argument	Description
reset_source	Directs a TCP reset packet to the endpoint that sent the packet that triggered the rule. Alternatively, you can specify `rst_snd`, which is supported for backward compatibility.
reset_dest	Directs a TCP reset packet to the intended destination endpoint of the packet that triggered the rule. Alternatively, you can specify `rst_rcv`, which is supported for backward compatibility.
reset_both	Directs a TCP reset packet to both the sending and receiving endpoints. Alternatively, you can specify `rst_all`, which is supported for backward compatibility.
icmp_net	Directs an ICMP network unreachable message to the sender.
icmp_host	Directs an ICMP host unreachable message to the sender.
icmp_port	Directs an ICMP port unreachable message to the sender. This argument is used to terminate UDP traffic.
icmp_all	Directs the following ICMP messages to the sender: • network unreachable • host unreachable • port unreachable

For example, to configure a rule to reset both sides of a connection when a rule is triggered, use `reset_both` as the value for the `resp` keyword.

You can use a comma-separated list to specify multiple arguments as follows:

```
argument,argument,argument
```

The react Keyword

You can use the `react` keyword to send a default HTML page to the TCP connection client when a packet triggers the rule; after sending the HTML page, the system uses TCP reset packets to initiate active responses to both ends of the connection. The `react` keyword does not trigger active responses for UDP traffic.

Optionally, you can specify the following argument:

```
msg
```

When a packet triggers a `react` rule that uses the `msg` argument, the HTML page includes the rule event message.

If you do not specify the `msg` argument, the HTML page includes the following message:

```
You are attempting to access a forbidden site.
Consult your system administrator for details.
```

Note Because active responses can be routed back, ensure that the HTML response page does not trigger a `react` rule; this could result in an unending sequence of active responses. Cisco recommends that you test `react` rules extensively before activating them in a production environment.

The detection_filter Keyword

You can use the `detection_filter` keyword to prevent a rule from generating events unless a specified number of packets trigger the rule within a specified time. This can stop the rule from prematurely generating events. For example, two or three failed login attempts within a few seconds could be expected behavior, but a large number of attempts within the same time could indicate a brute force attack.

The `detection_filter` keyword requires arguments that define whether the system tracks the source or destination IP address, the number of times the detection criteria must be met before triggering an event, and how long to continue the count.

Use the following syntax to delay the triggering of events:

```
track by_src/by_dst, count count, seconds number_of_seconds
```

The `track` argument specifies whether to use the packet's source or destination IP address when counting the number of packets that meet the rule's detection criteria. Select from the argument values described in the following table to specify how the system tracks event instances.

Table 75: detection_filter Track Arguments

Argument	Description
by_src	Detection criteria count by source IP address.
by_dst	Detection criteria count by destination IP address.

The `count` argument specifies the number of packets that must trigger the rule for the specified IP address within the specified time before the rule generates an event.

The `seconds` argument specifies the number of seconds within which the specified number of packets must trigger the rule before the rule generates an event.

Consider the case of a rule that searches packets for the content `foo` and uses the `detection_filter` keyword with the following arguments:

```
track by_src, count 10, seconds 20
```

In the example, the rule will not generate an event until it has detected `foo` in 10 packets within 20 seconds from a given source IP address. If the system detects only 7 packets containing `foo` within the first 20 seconds,

no event is generated. However, if `foo` occurs 40 times in the first 20 seconds, the rule generates 30 events and the count begins again when 20 seconds have elapsed.

Comparing the threshold and detection_filter Keywords

The `detection_filter` keyword replaces the deprecated `threshold` keyword. The `threshold` keyword is still supported for backward compatibility and operates the same as thresholds that you set within an intrusion policy.

The `detection_filter` keyword is a detection feature that is applied before a packet triggers a rule. The rule does not generate an event for triggering packets detected before the specified packet count and, in an inline deployment, does not drop those packets if the rule is set to drop packets. Conversely, the rule does generate events for packets that trigger the rule and occur after the specified packet count and, in an inline deployment, drops those packets if the rule is set to drop packets.

Thresholding is an event notification feature that does not result in a detection action. It is applied after a packet triggers an event. In an inline deployment, a rule that is set to drop packets drops all packets that trigger the rule, independent of the rule threshold.

Note that you can use the `detection_filter` keyword in any combination with the intrusion event thresholding, intrusion event suppression, and rate-based attack prevention features in an intrusion policy. Note also that policy validation fails if you enable an imported local rule that uses the deprecated `threshold` keyword in combination with the intrusion event thresholding feature in an intrusion policy.

The tag Keyword

Use the `tag` keyword to tell the system to log additional traffic for the host or session. Use the following syntax when specifying the type and amount of traffic you want to capture using the `tag` keyword:

```
tagging_type, count, metric, optional_direction
```

The next three tables describe the other available arguments.

You can choose from two types of tagging. The following table describes the two types of tagging. Note that the session tag argument type causes the system to log packets from the same session as if they came from different sessions if you configure only rule header options in the intrusion rule. To group packets from the same session together, configure one or more rule options (such as a `flag` keyword or `content` keyword) within the same intrusion rule.

Table 76: Tag Arguments

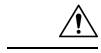
Argument	Description
session	Logs packets in the session that triggered the rule.
host	Logs packets from the host that sent the packet that triggered the rule. You can add a directional modifier to log only the traffic coming from the host (`src`) or going to the host (`dst`).

To indicate how much traffic you want to log, use the following argument:

Table 77: Count Argument

Argument	Description
count	The number of packets or seconds you want to log after the rule triggers. This unit of measure is specified with the metric argument, which follows the count argument.

Select the metric you want to use to log by time or volume of traffic from those described in the following table.



Caution High-bandwidth networks can see thousands of packets per second, and tagging a large number of packets may seriously affect performance, so make sure you tune this setting for your network environment.

Table 78: Logging Metrics Arguments

Argument	Description
packets	Logs the number of packets specified by the count after the rule triggers.
seconds	Logs traffic for the number of seconds specified by the count after the rule triggers.

For example, when a rule with the following `tag` keyword value triggers:

```
host, 30, seconds, dst
```

all packets that are transmitted from the client to the host for the next 30 seconds are logged.

The flowbits Keyword

Use the `flowbits` keyword to assign state names to sessions. By analyzing subsequent packets in a session according to the previously named state, the system can detect and alert on exploits that span multiple packets in a single session.

The `flowbits` state name is a user-defined label assigned to packets in a specific part of a session. You can label packets with state names based on packet content to help distinguish malicious packets from those you do not want to alert on. You can define up to 1024 state names per managed device. For example, if you want to alert on malicious packets that you know only occur after a successful login, you can use the `flowbits` keyword to filter out the packets that constitute an initial login attempt so you can focus only on the malicious packets. You can do this by first creating a rule that labels all packets in the session that have an established login with a `logged_in` state, then creating a second rule where `flowbits` checks for packets with the state you set in the first rule and acts only on those packets.

An optional *group name* allows you to include a state name in a group of states. A state name can belong to several groups. States not associated with a group are not mutually exclusive, so a rule that triggers and sets a state that is not associated with a group does not affect other currently set states.

flowbits Keyword Options

The following table describes the various combinations of operators, states, and groups available to the `flowbits` keyword. Note that state names can contain alphanumeric characters, periods (.), underscores (_), and dashes (-).

Table 79: flowbits Options

Operator	State Option	Group	Description
`set`	`state_name`	optional	Sets the specified state for a packet. Sets the state in the specified group if a group is defined.
`set`	`state_name&state_name`	optional	Sets the specified states for a packet. Sets the states in the specified group if a group is defined.
`setx`	`state_name`	mandatory	Sets the specified state in the specified group for a packet, and unsets all other states in the group.
`setx`	`state_name&state_name`	mandatory	Sets the specified states in the specified group for a packet, and unsets all other states in the group.
`unset`	`state_name`	no group	Unsets the specified state for a packet.
`unset`	`state_name&state_name`	no group	Unsets the specified states for a packet.
`unset`	`all`	mandatory	Unsets all the states in the specified group.
`toggle`	`state_name`	no group	Unsets the specified state if it is set, and sets the specified state if it is unset.
`toggle`	`state_name&state_name`	no group	Unsets the specified states if they are set, and sets the specified states if they are unset.
`toggle`	`all`	mandatory	Unsets all states set in the specified group, and sets all states unset in the specified group.
`isset`	`state_name`	no group	Determines if the specified state is set in the packet.
`isset`	`state_name&state_name`	no group	Determines if the specified states are set in the packet.
`isset`	`state_name|state_name`	no group	Determines if any of the specified states are set in the packet.
`isset`	`any`	mandatory	Determines if any state is set in the specified group.
`isset`	`all`	mandatory	Determines if all states are set in the specified group.
`isnotset`	`state_name`	no group	Determines if the specified state is not set in the packet.

Operator	State Option	Group	Description
`isnotset`	`state_name&state_name`	no group	Determines if the specified states are not set in the packet.
`isnotset`	`state_name|state_name`	no group	Determines if any of the specified states is not set in the packet.
`isnotset`	`any`	mandatory	Determines if any state is not set in the packet.
`isnotset`	`all`	mandatory	Determines if all states are not set in the packet.
`reset`	(no state)	optional	Unsets all states for all packets. Unsets all states in a group if a group is specified.
`noalert`	(no state)	no group	Use this in conjunction with any other operator to suppress event generation.

Guidelines for Using the flowbits Keyword

Note the following when using the `flowbits` keyword:

- When using the `setx` operator, the specified state can only belong to the specified group, and not to any other group.
- You can define the `setx` operator multiple times, specifying different states and the same group with each instance.
- When you use the `setx` operator and specify a group, you cannot use the `set`, `toggle`, or `unset` operators on that specified group.
- The `isset` and `isnotset` operators evaluate for the specified state regardless of whether the state is in a group.
- During intrusion policy saves, intrusion policy reapplies, and access control policy applies (regardless of whether the access control policy references one intrusion policy or multiple intrusion policies), if you enable a rule that contains the `isset` or `isnotset` operator **without** a specified group, and you do not enable at least one rule that affects `flowbits` assignment (`set`, `setx`, `unset`, `toggle`) for the corresponding state name and protocol, all rules that affect `flowbits` assignment for the corresponding state name are enabled.
- During intrusion policy saves, intrusion policy reapplies, and access control policy applies (regardless of whether the access control policy references one intrusion policy or multiple intrusion policies), if you enable a rule that contains the `isset` or `isnotset` operator **with** a specified group, all rules that affect `flowbits` assignment (`set`, `setx`, `unset`, `toggle`) and define a corresponding group name are also enabled.

flowbits Keyword Examples

This section provides three examples that use the `flowbits` keyword.

flowbits Keyword Example: A Configuration Using state_name

This is an example of a flowbits configuration using `state_name`.

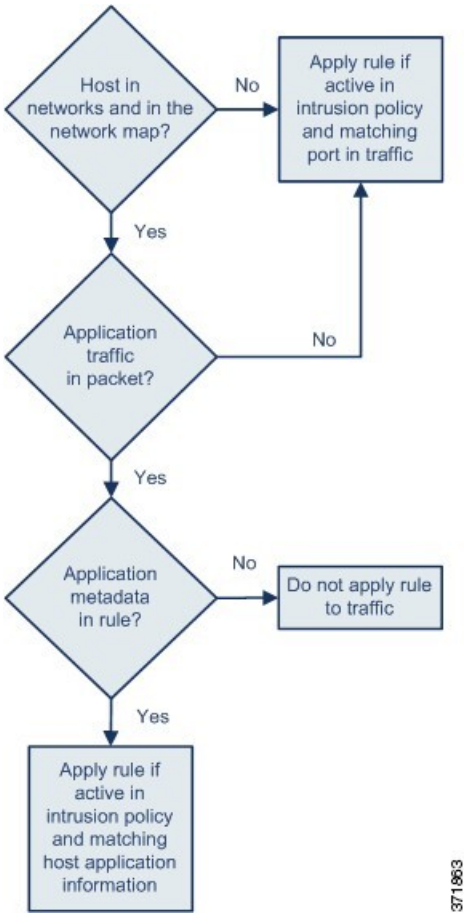
Consider the IMAP vulnerability described in CVE ID 2000-0284. This vulnerability exists in an implementation of IMAP, specifically in the LIST, LSUB, RENAME, FIND, and COPY commands. However, to take advantage of the vulnerability, the attacker must be logged into the IMAP server. Because the LOGIN confirmation from the IMAP server and the exploit that follows are necessarily in different packets, it is difficult to construct non-flow-based rules that catch this exploit. Using the `flowbits` keyword, you can cflowbitonstruct a series of rules that track whether the user is logged into the IMAP server and, if so, generate an event if one of the attacks is detected. If the user is not logged in, the attack cannot exploit the vulnerability and no event is generated.

The two rule fragments that follow illustrate this example. The first rule fragment looks for an IMAP login confirmation from the IMAP server:

```
alert tcp any 143 -> any any (msg:"IMAP login"; content:"OK

LOGIN"; flowbits:set,logged_in; flowbits:noalert;)
```

The following diagram illustrates the effect of the `flowbits` keyword in the preceding rule fragment:



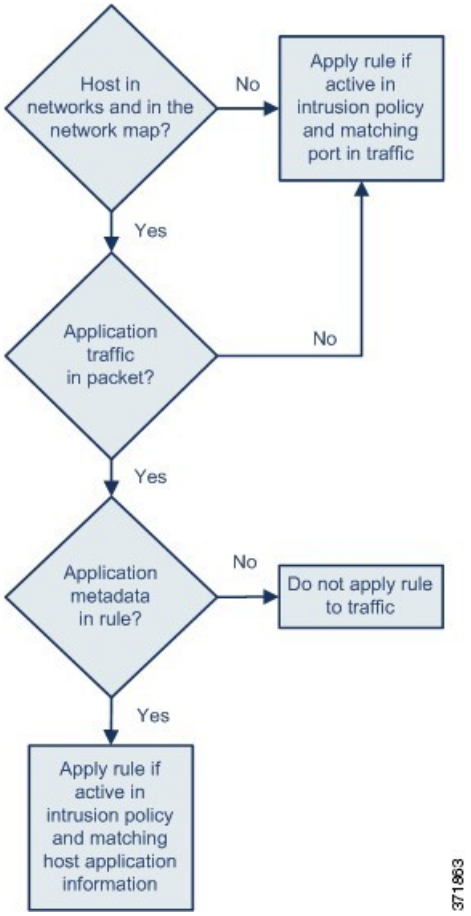
Note that `flowbits:set` sets a state of `logged_in`, while `flowbits:noalert` suppresses the alert because you are likely to see many innocuous login sessions on an IMAP server.

The next rule fragment looks for a LIST string, but does not generate an event unless the logged_in state has been set as a result of some previous packet in the session:

```
alert tcp any any -> any 143 (msg:"IMAP LIST";

content:"LIST"; flowbits:isset,logged_in;)
```

The following diagram illustrates the effect of the `flowbits` keyword in the preceding rule fragment:



In this case, if a previous packet has caused a rule containing the first fragment to trigger, then a rule containing the second fragment triggers and generates an event.

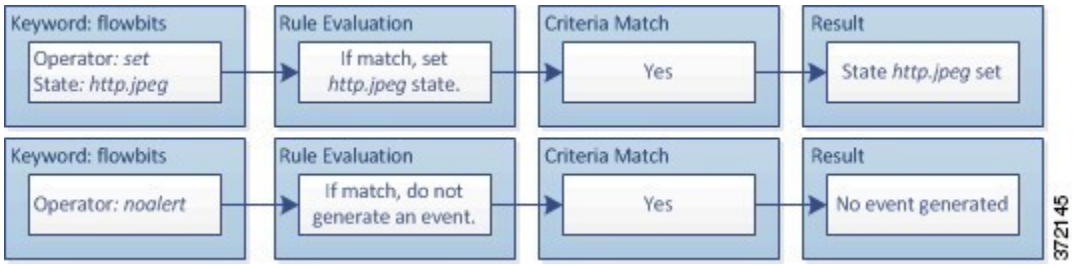
flowbits Keyword Example: A Configuration Resulting in False Positive Events

Including different state names that are set in different rules in a group can prevent false positive events that might otherwise occur when content in a subsequent packet matches a rule whose state is no longer valid. The following example illustrates how you can get false positives when you do not include multiple state names in a group.

Consider the case where the following three rule fragments trigger in the order shown during a single session:

```
(msg:"JPEG transfer";
content:"image/";pcre:"/^Content-?Type\x3a(\s*|\s*\r?\n\s+)image\x2fp?jpe?g/smi";
?flowbits:set,http.jpeg; flowbits:noalert;)
```

The following diagram illustrates the effect of the `flowbits` keyword in the preceding rule fragment:

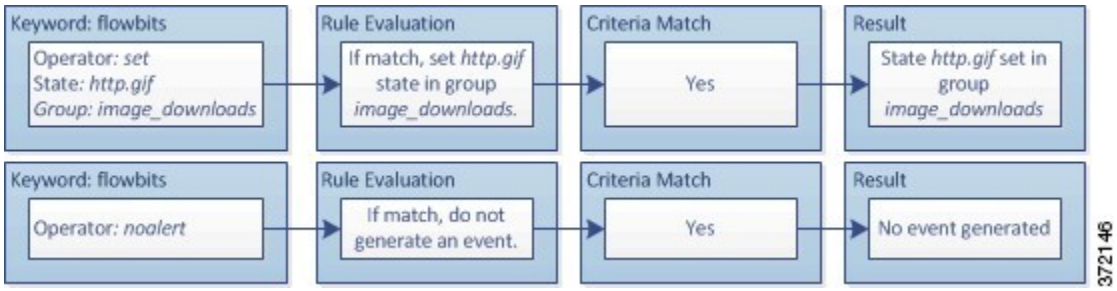


The `content` and `pcre` keywords in the first rule fragment match a JPEG file download, `flowbits:set,http.jpeg` sets the `http.jpeg flowbits` state, and `flowbits:noalert` stops the rule from generating events. No event is generated because the rule's purpose is to detect the file download and set the `flowbits` state so one or more companion rules can test for the state name in combination with malicious content and generate events when malicious content is detected.

The next rule fragment detects a GIF file download subsequent to the JPEG file download above:

```
(msg:"GIF transfer"; content:"image/";
pcre:"/^Content-?Type\x3a(\s*|\s*\r?\n\s+)image\x2fgif/smi";
?flowbits:set,http.jpg,image_downloads; flowbits:noalert;)
```

The following diagram illustrates the effect of the `flowbits` keyword in the preceding rule fragment:

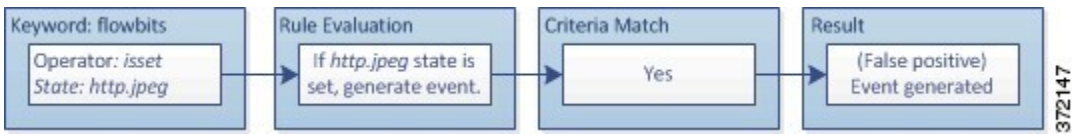


The `content` and `pcre` keywords in the second rule match the GIF file download, `flowbits:set,http.jpg` sets the `http.jpg` flowbit state, and `flowbits:noalert` stops the rule from generating an event. Note that the `http.jpeg` state set by the first rule fragment is still set even though it is no longer needed; this is because the JPEG download must have ended if a subsequent GIF download has been detected.

The third rule fragment is a companion to the first rule fragment:

```
(msg:"JPEG exploit";?flowbits:isset,http.jpeg;content:"|FF|";
pcre:"?/\xFF[\xE1\xE2\xED\xFE]\x00[\x00\x01]/";)
```

The following diagram illustrates the effect of the `flowbits` keyword in the preceding rule fragment:



In the third rule fragment, `flowbits:isset,http.jpeg` determines that the now-irrelevant `http.jpeg` state is set, and `content` and `pcre` match content that would be malicious in a JPEG file but not in a GIF file. The third rule fragment results in a false positive event for a nonexistent exploit in a JPEG file.

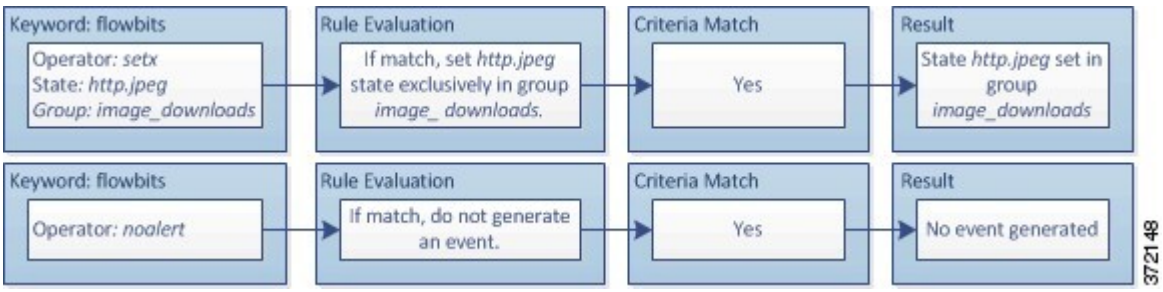
flowbits Keyword Example: A Configuration for Preventing False Positive Events

The following example illustrates how including state names in a group and using the `setx` operator can prevent false positives.

Consider the same case as the previous example, except that the first two rules now include their two different state names in the same state group.

```
(msg:"JPEG transfer";
content:"image/";pcre:"/^Content-?Type\x3a(\s*|\s*\r?\n\s+)image\x2fp?jpe?g/smi";
?flowbits:setx,http.jpeg,image_downloads; flowbits:noalert;)
```

The following diagram illustrates the effect of the `flowbits` keyword in the preceding rule fragment:

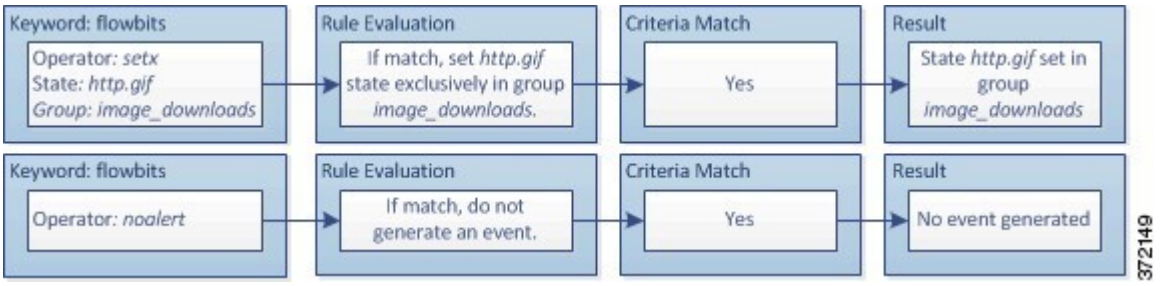


When the first rule fragment detects a JPEG file download, the `flowbits:setx,http.jpeg,image_downloads` keyword sets the `flowbits` state to `http.jpeg` and includes the state in the `image_downloads` group.

The next rule then detects a subsequent GIF file download:

```
(msg:"GIF transfer"; content:"image/";
pcre:"/^Content-?Type\x3a(\s*|\s*\r?\n\s+)image\x2fgif/smi";
?flowbits:setx,http.jpg,image_downloads; flowbits:noalert;)
```

The following diagram illustrates the effect of the `flowbits` keyword in the preceding rule fragment:

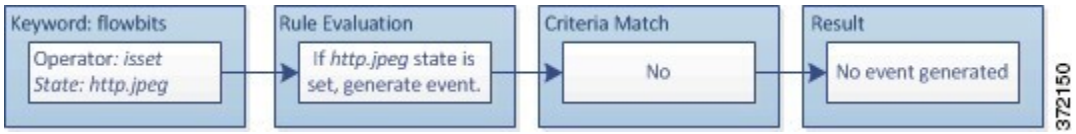


When the second rule fragment matches the GIF download, the `flowbits:setx,http.jpg,image_downloads` keyword sets the `http.jpg flowbits` state and unsets `http.jpeg`, the other state in the group.

The third rule fragment does not result in a false positive:

```
(msg:"JPEG exploit"; ?flowbits:isset,http.jpeg;content:"|FF|";
pcre:"/?\xFF[\xE1\xE2\xED\xFE]\x00[\x00\x01]/";)
```

The following diagram illustrates the effect of the `flowbits` keyword in the preceding rule fragment:



Because `flowbits:isset,http.jpeg` is false, the rules engine stops processing the rule and no event is generated, thus avoiding a false positive even in a case where content in the GIF file matches exploit content for a JPEG file.

The http_encode Keyword

You can use the `http_encode` keyword to generate events on the type of encoding in an HTTP request or response before normalization, either in the HTTP URI, in non-cookie data in an HTTP header, in cookies in HTTP requests headers, or set-cookie data in HTTP responses.

You must configure the HTTP Inspect preprocessor to inspect HTTP responses and HTTP cookies to return matches for rules using the `http_encode` keyword.

Also, you must enable both the decoding and alerting option for each specific encoding type in your HTTP Inspect preprocessor configuration so the `http_encode` keyword in an intrusion rule can trigger events on that encoding type.

The following table describes the encoding types this option can generate events for in HTTP URIs, headers, cookies, and set-cookies:

Table 80: http_encode Encoding Types

Encoding Type	Description
utf8	Detects UTF-8 encoding in the specified location when this encoding type is enabled for decoding by the HTTP Inspect preprocessor.
double_encode	Detects double encoding in the specified location when this encoding type is enabled for decoding by the HTTP Inspect preprocessor.
non_ascii	Detects non-ASCII characters in the specified location when non-ASCII characters are detected but the detected encoding type is not enabled.
uencode	Detects Microsoft %u encoding in the specified location when this encoding type is enabled for decoding by the HTTP Inspect preprocessor.
bare_byte	Detects bare byte encoding in the specified location when this encoding type is enabled for decoding by the HTTP Inspect preprocessor.

http_encode Keyword Syntax

Encoding Location

Specifies whether to search for the specified encoding type in an HTTP URI, header, or cookie, including a set-cookie.

Encoding Type

Specifies one or more encoding types using one of the following formats:

```
encode_type
encode_type|encode_type|encode_type...
```

where `encode_type` is one of the following:

```
utf8
double_encode
non_ascii
uencode
bare_byte.
```

Note that you cannot use the negation (!) and OR (|) operators together.

http_encode Keyword example: Using Two http_endcode Keywords to Search for Two Encodings

The following example uses two `http_encode` keywords in the same rule to search the HTTP URI for UTF-8 AND Microsoft IIS %u encoding:

First, the `http_encode` keyword:

- **Encoding Location**: `HTTP URI`
- **Encoding Type**: `utf8`

Then, the additional `http_encode` keyword:

- **Encoding Location**: `HTTP URI`
- **Encoding Type**: `uencode`

Overview: The file_type and file_group Keywords

The `file_type` and `file_group` keywords allow you to detect files transmitted via FTP, HTTP, SMTP, IMAP, POP3, and NetBIOS-ssn (SMB) based on their type and version. Do **not** use more than one `file_type` or `file_group` keyword in a single intrusion rule.

Tip Updating your vulnerability database (VDB) populates the intrusion rules editor with the most up-to-date file types, versions, and groups.

Note The system does not automatically enable preprocessors to accomodate the `file_type` and `file_group` keywords.

You **must** enable specific preprocessors if you want to generate events and, in an inline deployment, drop offending packets for traffic matching your `file_type` or `file_group` keywords.

Table 81: file_type and file_group Intrusion Event Generation

Protocol	Required Preprocessor or Preprocessor Option
FTP	FTP/Telnet preprocessor and the **Normalize TCP Payload** inline normalization preprocessor option
HTTP	HTTP Inspect preprocessor to generate intrusion events in HTTP traffic
SMTP	SMTP preprocessor to generate intrusion events in HTTP traffic
IMAP	IMAP preprocessor
POP3	POP preprocessor
Netbios-ssn (SMB)	The DCE/RPC preprocessor and the **SMB File Inspection** DCE/RPC preprocessor option

The file_type and file_group Keywords

file_type

The `file_type` keyword allows you to specify the file type and version of a file detected in traffic. File type arguments (for example, **JPEG** and **PDF**) identify the format of the file you want to find in traffic.

Note Do **not** use the `file_type` keyword with another `file_type` or `file_group` keyword in the same intrusion rule.

The system selects **Any Version** by default, but some file types allow you to select version options (for example, PDF version **1.7**) to identify specific file type versions you want to find in traffic.

file_group

The `file_group` keyword allows you to select a Cisco-defined group of similar file types to find in traffic (for example, **multimedia** or **audio**). File groups also include Cisco-defined versions for each file type in the group.

Note Do **not** use the `file_group` keyword with another `file_group` or `file_type` keyword in the same intrusion rule.

The file_data Keyword

The `file_data` keyword provides a pointer that serves as a reference for the positional arguments available for other keywords such as `content`, `byte_jump`, `byte_test`, and `pcre`. The detected traffic determines the type of data the `file_data` keyword points to. You can use the `file_data` keyword to point to the beginning of the following payload types:

- HTTP response body

To inspect HTTP response packets, the HTTP Inspect preprocessor must be enabled and you must configure the preprocessor to inspect HTTP responses. The `file_data` keyword matches if the HTTP Inspect preprocessor detects HTTP response body data.

- Uncompressed gzip file data

  To inspect uncompressed gzip files in the HTTP response body, the HTTP Inspect preprocessor must be enabled and you must configure the preprocessor to inspect HTTP responses and to decompress gzip-compressed files in the HTTP response body. For more information, see the **Inspect HTTP Responses** and **Inspect Compressed Data** Server-Level HTTP Normalization options. The `file_data` keyword matches if the HTTP Inspect preprocessor detects uncompressed gzip data in the HTTP response body.

- Normalized JavaScript

  To inspect normalized JavaScript data, the HTTP Inspect preprocessor must be enabled and you must configure the preprocessor to inspect HTTP responses. The `file_data` keyword matches if the HTTP Inspect preprocessor detects JavaScript in response body data.

- SMTP payload

  To inspect the SMTP payload, the SMTP preprocessor must be enabled. The `file_data` keyword matches if the SMTP preprocessor detects SMTP data.

- Encoded email attachments in SMTP, POP, or IMAP traffic

  To inspect email attachments in SMTP, POP, or IMAP traffic, the SMTP, POP, or IMAP preprocessor, respectively, must be enabled, alone or in any combination. Then, for each enabled preprocessor, you must ensure that the preprocessor is configured to decode each attachment encoding type that you want decoded. The attachment decoding options that you can configure for each preprocessor are: **Base64 Decoding Depth**, **7-Bit/8-Bit/Binary Decoding Depth**, **Quoted-Printable Decoding Depth**, and **Unix-to-Unix Decoding Depth**.

  You can use multiple `file_data` keywords in a rule.

The pkt_data Keyword

The `pkt_data` keyword provides a pointer that serves as a reference for the positional arguments available for other keywords such as `content`, `byte_jump`, `byte_test`, and `pcre`.

When normalized FTP, telnet, or SMTP traffic is detected, the `pkt_data` keyword points to the beginning of the normalized packet payload. When other traffic is detected, the `pkt_data` keyword points to the beginning of the raw TCP or UDP payload.

The following normalization options must be enabled for the system to normalize the corresponding traffic for inspection by intrusion rules:

- Enable the FTP & Telnet preprocessor **Detect Telnet Escape codes within FTP commands** option to normalize FTP traffic for inspection.
- Enable the FTP & Telnet preprocessor **Normalize** telnet option to normalize telnet traffic for inspection.
- Enable the SMTP preprocessor **Normalize** option to normalize SMTP traffic for inspection.

You can use multiple `pkt_data` keywords in a rule.

The base64_decode and base64_data Keywords

You can use the `base64_decode` and `base64_data` keywords in combination to instruct the rules engine to decode and inspect specified data as Base64 data. This can be useful, for example, for inspecting Base64-encoded HTTP Authentication request headers and Base64-encoded data in HTTP PUT and POST requests.

These keywords are particularly useful for decoding and inspecting Base64 data in HTTP requests. However, you can also use them with any protocol such as SMTP that uses the space and tab characters the same way HTTP uses these characters to extend a lengthy header line over multiple lines. When this line extension, which is known as folding, is not present in a protocol that uses it, inspection ends at any carriage return or line feed that is not followed with a space or tab.

base64_decode

The `base64_decode` keyword instructs the rules engine to decode packet data as Base64 data. Optional arguments let you specify the number of bytes to decode and where in the data to begin decoding.

You can use the `base64_decode` keyword once in a rule; it must precede at least one instance of the `base64_data` keyword.

Before decoding Base64 data, the rules engine unfolds lengthy headers that are folded across multiple lines. Decoding ends when the rules engine encounters any the following:

- the end of a header line
- the specified number of bytes to decode
- the end of the packet

The following table describes the arguments you can use with the `base64_decode` keyword.

Table 82: Optional base64_decode Arguments

Argument	Description
Bytes	Specifies the number of bytes to decode. When not specified, decoding continues to the end of a header line or the end of the packet payload, whichever comes first. You can specify a positive, non-zero value.
Offset	Determines the offset relative to the start of the packet payload or, when you also specify **Relative**, relative to the current inspection location. You can specify a positive, non-zero value.
Relative	Specifies inspection relative to the current inspection location.

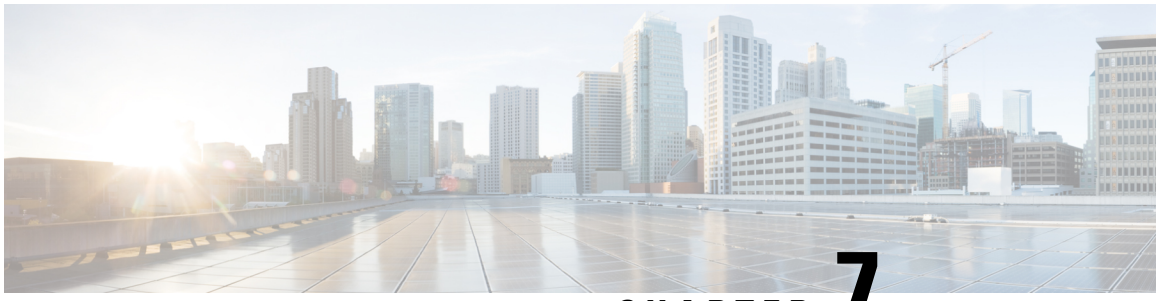
base64_data

The `base64_data` keyword provides a reference for inspecting Base64 data decoded using the `base64_decode` keyword. The `base64_data` keyword sets inspection to begin at the start of the decoded Base64 data. Optionally, you can then use the positional arguments available for other keywords such as `content` or `byte_test` to further specify the location to inspect.

You must use the `base64_data` keyword at least once after using the `base64_decode` keyword; optionally, you can use `base64_data` multiple times to return to the beginning of the decoded Base64 data.

Note the following when inspecting Base64 data:

- You cannot use the fast pattern matcher.
- If you interrupt Base64 inspection in a rule with an intervening HTTP content argument, you must insert another `base64_data` keyword in the rule before further inspecting Base64 data.



CHAPTER 7

Layers in Intrusion and Network Analysis Policies

The following topics explain how to use layers in intrusion and network analysis policies:

Layer Basics

Larger organizations with many managed devices may have many intrusion policies and network analysis policies to support the unique needs of different departments, business units or, in some instances, different companies. Configurations in both policy types are contained in building blocks called *layers,* which you can use to efficiently manage multiple policies.

Layers in intrusion and network analysis policies work in essentially the same way. You can create and edit either policy type without consciously using layers. You can modify your policy configurations and, if you have not added user layers to your policy, the system automatically includes your changes in a single configurable layer that is initially named *My Changes*. You can also add up to 200 layers where you can configure any combination of settings. You can copy, merge, move, and delete user layers and, most important, share individual user layers with other policies of the same type.

License Requirements for Network Analysis and Intrusion Policy Layers

Threat Defense License

IPS

Requirements and Prerequisites for Network Analysis and Intrusion Policy Layers

Model support

Any.

Supported domains

Any

User roles

- Admin
- Intrusion Admin

The Layer Stack

Layer stacks are composed of the following:

User Layers

User-configurable layers. You can copy, merge, move, or delete any user-configurable layer and set any user-configurable layer to be shared by other policies of the same type. This layer includes the automatically-generated layer initially named My Changes.

Built-in Layers

The read-only base policy layer. The policy in this layer can be either a system-provided policy or a custom policy you created.

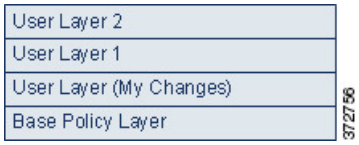
By default, a network analysis or intrusion policy includes a base policy layer and a My Changes layer. You can add user layers as necessary.

Each policy layer contains complete configurations for either all preprocessors in a network analysis policy or all intrusion rules and advanced settings in an intrusion policy. The lowest, base policy layer includes all the settings from the base policy you selected when you created the policy. A setting in a higher layer takes precedence over the same setting in a lower layer. Features not explicitly set in a layer *inherit* their settings from the next highest layer where they are explicitly set. The system *flattens* the layers, that is, it applies only the cumulative effect of all settings, when it handles network traffic.

Tip You can create an intrusion or network analysis policy based solely on the default settings in the base policy. In the case of an intrusion policy, you can also use Firepower rule state recommendations if you want to tailor your intrusion policy to the specific needs of your monitored network.

The following figure shows an example layer stack that, in addition to the base policy layer and the initial My Changes layer, also includes two additional user-configurable layers, *User Layer 1* and *User Layer 2*. Note

in the figure that each user-configurable layer that you add is initially positioned as the highest layer in the stack; thus, User Layer 2 in the figure was added last and is highest in the stack.



Regardless of whether you allow rule updates to modify your policy, changes in a rule update never override changes you make in a layer. This is because changes in a rule update are made in the base policy, which determines the default settings in your base policy layer; your changes are always made in a higher layer, so they override any changes that a rule update makes to your base policy.

The Base Layer

The base layer, also referred to as the base policy, of an intrusion or network analysis policy defines the default settings for all configurations in the policy, and is the lowest layer in the policy. When you create a new policy and change a setting without adding new layers, the change is stored in the My Changes layer, and overrides—but does not change—the setting in the base policy.

System-Provided Base Policies

The system provides several pairs of network analysis and intrusion policies. By using system-provided network analysis and intrusion policies, you can take advantage of the experience of the Talos Intelligence Group. For these policies, Talos sets intrusion and preprocessor rule states, as well as provides the initial configurations for preprocessors and other advanced settings. You can use these system-provided policies as-is, or you can use them as the base for custom policies.

If you use a system-provided policy as your base, importing rule updates may modify settings in your base policy. However, you can configure a custom policy so that the system does not automatically make these changes to its system-provided base policy. This allows you to update system-provided base policies manually, on a schedule independent of rule updates. In either case, changes that a rule update makes to your base policy do not change or override settings in your My Changes or any other layer.

System-provided intrusion and network analysis policies are similarly named but contain different configurations. For example, the Balanced Security and Connectivity network analysis policy and the Balanced Security and Connectivity intrusion policy work together and can both be updated in intrusion rule updates.

Custom Base Policies

You can use a custom policy as your base. You can tune settings in custom policies to inspect traffic in ways that matter most to you so you can improve both the performance of your managed devices and your ability to respond effectively to the events they generate.

If you change the custom policy that you use as the base for another policy, those changes are automatically used as the default settings of the policy that uses the base.

In addition, a rule update may affect your policy even if you use a custom base policy, because all policies have a system-provided policy as the eventual base in a policy chain. If the first custom policy in a chain (the one that uses the system-provided policy as its base) allows rule updates to modify its base policy, your policy may be affected.

Regardless of how changes are made to your base policy—whether by a rule update or when you modify a custom policy that you use as a base policy—they do not change or override settings in your My Changes or any other layer.

The Effect of Rule Updates on Base Policies

When you import rule updates, the system modifies system-provided intrusion, access control, and network analysis policies. Rule updates can include:

- modified network analysis preprocessor settings
- modified advanced settings in intrusion and access control policies
- new and updated intrusion rules
- modified states for existing rules
- new rule categories and default variables

Rule updates can also delete existing rules from system-provided policies.

Changes to default variables and rule categories are handled at the system level.

When you use a system-provided policy as your intrusion or network analysis base policy, you can allow rule updates to modify your base policy which, in this case, is a copy of the system-provided policy. If you allow rule updates to update your base policy, a new rule update makes the same changes in your base policy that it makes to the system-provided policy that you use as your base policy. If you have not modified the corresponding setting, a setting in your base policy determines the setting in your policy. However, rule updates do not override changes you make in your policy.

If you do not allow rule updates to modify your base policy, you can manually update your base policy after importing one or more rule updates.

Rule updates always delete intrusion rules that Talos deletes, regardless of the rule state in your intrusion policy or whether you allow rule updates to modify your base intrusion policy.

Until you re-deploy your changes to network traffic, rules in your currently deployed intrusion policies behave as follows:

- Disabled intrusion rules remain disabled.
- Rules set to **Generate Events** continue to generate events when triggered.
- Rules set to **Drop and Generate Events** continue to generate events and drop offending packets when triggered.

Rule updates do not modify a custom base policy unless both of the following conditions are met:

- You allow rule updates to modify the system-provided base policy of the parent policy, that is, the policy that originated the custom base policy.
- You have not made changes in the parent policy that override the corresponding settings in the parent's base policy.

When both conditions are met, changes in the rule update are passed to the child policy, that is, the policy using the custom base policy, when you save the parent policy.

For example, if a rule update enables a previously disabled intrusion rule, and you have not modified the rule's state in the parent intrusion policy, the modified rule state is passed to the base policy when you save the parent policy.

Likewise, if a rule update modifies a default preprocessor setting and you have not modified the setting in the parent network analysis policy, the modified setting is passed to the base policy when you save the parent policy.

Changing the Base Policy

You can choose a different system-provided or custom policy as your base policy.

You can chain up to five custom policies, with four of the five using one of the other four previously created policies as its base policy; the fifth must use a system-provided policy as its base.

Procedure

Step 1 Choose **Policies** > **Access Control heading** > **Intrusion**.

Step 2 Click **Snort 2 Version** next to the policy you want to edit.

If **View** ( ) appears instead, the configuration belongs to an ancestor domain, or you do not have permission to modify the configuration.

Step 3 Click **Edit** ( ) in the required intrusion policy row.

Step 4 Choose a base policy from the **Base Policy** drop-down list.

Step 5 Click **Save**.

What to do next

- Deploy configuration changes.

The Cisco Recommendations Layer

When you generate rule state recommendations in an intrusion policy, you can choose whether to automatically modify rule states based on the recommendations.

As seen in the following figure, using recommended rule states inserts a read-only, built-in Cisco Recommendations layer immediately above the base layer.

Note that this layer is unique to intrusion policies.

If you subsequently choose not to use recommended rule states, the system removes the Cisco Recommendations layer. You cannot manually delete this layer, but you can add and remove it by choosing to use or not use recommended rule states.

Adding the Cisco Recommendations layer adds a Cisco Recommendations link under Policy Layers in the navigation panel. This link leads you to a read-only view of the Cisco Recommendations layer page where you can access a recommendation-filtered view of the Rules page in read-only mode.

Using recommended rule states also adds a Rules sublink beneath the Cisco Recommendations link in the navigation panel. The Rules sublink provides access to a read-only display of the Rules page in the Cisco Recommendations layer. Note the following in this view:

- When there is no rule state icon in the state column, the state is inherited from the base policy.
- When there is no rule state icon in the Cisco Recommendation column in this or other Rules page views, there is no recommendation for this rule.

Layer Management

The Policy Layers page provides a single-page summary of the complete layer stack for your network analysis or intrusion policy. On this page you can add shared and unshared layers, copy, merge, move, and delete layers, access the summary page for each layer, and access configuration pages for enabled, disabled, and overridden configurations within each layer.

For each layer, you can view the following information:

- whether the layer is a built-in, shared user, or unshared user layer
- which layers contain the highest, that is the effective, preprocessor or advanced setting configurations, by feature name
- in an intrusion policy, the number of intrusion rules whose states are set in the layer, and the number of rules set to each rule state.

The Policy Layers page also provides a summary of the net effect of all enabled preprocessors (network analysis) or advanced settings (intrusion) and, for intrusion policies, intrusion rules.

The feature name in the summary for each layer indicates which configurations are enabled, disabled, overridden, or inherited in the layer, as follows:

When the feature is...	The feature name is...
enabled in the layer	written in plain text
disabled in the layer	struck out
overridden by the configuration in a higher layer	written in italic text
inherited from a lower layer	not present

You can add up to 200 layers to a network analysis or intrusion policy. When you add a layer, it appears as the highest layer in your policy. The initial state is Inherit for all features and, in an intrusion policy, no event filtering, dynamic state, or alerting rule actions are set.

You give a user-configurable layer a unique name when you add the layer to your policy. Later, you can change the name and, optionally, add or modify a description that is visible when you edit the layer.

You can copy a layer, move a layer up or down within the User Layers page area, or delete a user layer, including the initial My Changes layer. Note the following considerations:

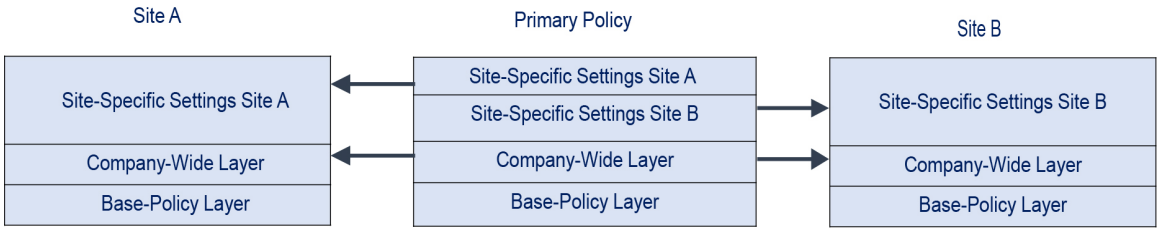
- When you copy a layer, the copy appears as the highest layer.
- Copying a shared layer creates a layer that is initially unshared and which you can then share if you choose.
- You cannot delete a shared layer; a layer with sharing enabled that you have not shared with another policy is not a shared layer.

You can merge a user-configurable layer with another user-configurable layer immediately beneath it. A merged layer retains all settings that were unique to either layer, and accepts the settings from the higher layer if both layers included settings for the same preprocessor, intrusion rule, or advanced setting. The merged layer retains the name of the lower layer. In the policy where you create a sharable layer that you can add to other policies, you can merge an unshared layer immediately above the sharable layer with the sharable layer, but you cannot merge the sharable layer with an unshared layer beneath it. In a policy where you add a shared layer that you created in another policy, you can merge the shared layer into an unshared layer immediately beneath it and the resulting layer is no longer shared; you cannot merge an unshared layer into a shared layer beneath it.

Shared Layers

A *shared layer* is a layer you add to your policy after creating the layer in another policy where you allow it to be shared. A *sharable layer* is a layer you allow to be shared.

The following figure shows an example primary policy where you create the company-wide layer and site-specific layers for sites A and B, and allow these to be shared. You then add these as shared layers to the policies for sites A and B.



The company-wide layer in the primary policy includes settings applicable to sites A and B. The site-specific layers include settings specific to each site. For example, in the case of a network analysis policy Site A might not have web servers on the monitored network and would not require the protection or processing overhead of the HTTP Inspect preprocessor, but both sites would likely require TCP stream preprocessing. You could enable TCP stream processing in the company-wide layer that you share with both sites, disable the HTTP Inspect preprocessor in the site-specific layer that you share with Site A, and enable the HTTP Inspect preprocessor in the site-specific layer that you share with Site B. By editing configurations in a higher layer in the site-specific policies, you could also further tune the policy for each site if necessary with any configuration adjustments.

It is unlikely that the flattened net settings in the example primary policy would be useful for monitoring traffic, but the time saved in configuring and updating the site-specific policies makes this a useful application of policy layers.

Many other layer configurations are possible. For example, you could define policy layers by company, by department, by network, or even by user. In the case of an intrusion policy, you could also include advanced settings in one layer and rule settings in another.

You can allow a user-configurable layer to be shared with other policies of the same type (intrusion or network analysis). When you modify a configuration within a sharable layer and then commit your changes, the system updates all policies that share the layer and provides you with a list of all affected policies. You can only change feature configurations in the policy where you created the layer.

You cannot disable sharing for a layer that you have added to another policy; you must first delete the layer from the other policy or delete the other policy.

You cannot add a shared layer to a policy when your base policy is a custom policy where the layer you want to share was created. To do so would give the policy a circular dependency.

Managing Layers

Procedure

Step 1 While editing your Snort 2 policy, click **Policy Layers** in the navigation panel.

Step 2 You can take any of the following management actions on the Policy Layers page:

- Add a shared layer from another policy — Click **Add Shared Layer Add** (+) next to User Layers, choose the layer from the **Add Shared Layer** drop-down list, then click **OK**.
- Add an unshared layer — Click add layer **Add** (+) next to User Layers, enter a **Name**, and click **OK**.
- Add or change the layer description — Click **Edit** (✎) next to the layer, then add or change the **Description**.
- Allow a layer to be shared with another policy — Click **Edit** (✎) next to the layer, then clear the **Sharing** check box.
- Change the layer name — Click **Edit** (✎) next to the layer, then change the **Name**.
- Copy a layer — Click **Copy** (⧉) for the layer.
- Delete a layer — Click **Delete** (🗑) for the layer, then click **OK**.
- Merge two layers — Click **Merge** (⧉) for the upper of the two layers, then click **OK**.
- Move a layer — Click any open area in the layer summary and drag until the **Position Arrow** points to a line above or below a layer where you want to move the layer.

Step 3 To save changes you made in this policy since the last policy commit, click **Policy Information**, then click **Commit Changes**.

If you leave the policy without committing changes, changes since the last commit are discarded if you edit a different policy.

What to do next

- Deploy configuration changes.

Navigating Layers

Procedure

Step 1 While editing your Snort 2 policy, click **Policy Layers** in the navigation panel. To access your Snort 2 policy, choose **Policies** > **Intrusion** > **Intrusion Policies tab** and then click **Snort 2** against the policy you want to edit.

Step 2 You can take any of the following actions to navigate through your layers:

- Access a preprocessor or advanced settings page — If you want to access a layer-level preprocessor or advanced setting configuration page, click the feature name in the row for the layer. Configuration pages are read-only in the base policy and in shared layers.
- Access a rule page — If you want to access a layer-level rule configuration page filtered by rule state type, click **Drop and Generate Events**, **Generate Events**, or **Disabled** in the summary for the layer. No rules are displayed if the layer contains no rules set to the selected rule state.
- Display the Policy Information page — If you want to display the Policy Information page, click **Policy Summary** in the navigation panel.
- Display a layer summary page — If you want to display the summary page for a layer, click the layer name in the row for the layer or, alternately, click **Edit** (✎) next to a user layer. You can also click **View** (👁) to access the read-only summary page for a shared layer.

Step 3 To save changes you made in this policy since the last policy commit, click **Policy Information**, then click **Commit Changes**.

If you leave the policy without committing changes, changes since the last commit are discarded if you edit a different policy.

What to do next

- Deploy configuration changes.

Intrusion Rules in Layers

You can view individual layer settings on the Rules page for the layer, or view the net effect of all settings on the policy view of the Rules page. When you modify rule settings on the policy view of the Rules page, you are modifying the highest user-configurable layer in the policy. You can switch to another layer using the layer drop-down list on any Rules page.

The following table describes the effects of configuring the same type of setting in multiple layers.

Table 83: Layer Rule Settings

You can set...	Of this setting type...	To...
one	rule state	override a rule state set for the rule in a lower layer, and ignore all thresholds, suppressions, rate-based rule states, and alerts for that rule configured in lower layers. If you want a rule to inherit its state from the base policy or a lower layer, set the rule state to Inherit. Note that when you are working on the intrusion policy Rules page, you cannot set a rule state to Inherit because the intrusion policy Rules page is a composite view of the net effect of all rule settings.
one	threshold SNMP alert	override a setting of the same type for the rule in a lower layer. Note that setting a threshold overwrites any existing threshold for the rule in the layer.
one or more	suppression rate-based rule state	cumulatively combine settings of the same type for each selected rule down to the first layer where a rule state is set for the rule. Settings below the layer where a rule state is set are ignored.
one or more	comment	add a comment to a rule. Comments are rule-specific, not policy- or layer-specific. You can add one or more comments to a rule in any layer.

For example, if you set a rule state to Drop and Generate Events in one layer and to Disabled in a higher layer, the intrusion policy Rules page shows that the rule is disabled.

In another example, if you set a source-based suppression for a rule to 192.168.1.1 in one layer, and you also set a destination-based suppression for the rule to 192.168.1.2 in another layer, the Rules page shows that the cumulative effect is to suppress events for the source address 192.168.1.1 and the destination address 192.168.1.2. Note that suppression and rate-based rule state settings cumulatively combine settings of the same type for each selected rule down to the first layer where a rule state is set for the rule. Settings below the layer where a rule state is set are ignored.

Color-coding on each Rules page for a specific layer indicates whether the effective state is in a higher, lower, or the current layer, as follows:

- red—the effective state is in a higher layer
- yellow—the effective state is in a lower layer
- unshaded—the effective state is in the current layer

Because the intrusion policy Rules page is a composite view of the net effect of all rule settings, rule states are not color-coded on this page.

Configuring Intrusion Rules in Layers

In an intrusion policy, you can set the rule state, event filtering, dynamic state, alerting, and rule comments for a rule in any user-configurable layer. After accessing the layer where you want to make your changes, you add settings on the Rules page for the layer the same as you would on the intrusion policy Rules page.

Procedure

Step 1 While editing your Snort 2 intrusion policy, expand **Policy Layers** in the navigation panel.

Step 2 Expand the policy layer you want to modify.

Step 3 Click **Rules** immediately beneath the policy layer you want to modify.

Step 4 Modify any of the settings described in Tuning Intrusion Policies Using Rules, on page 28.

Tip
To delete an individual setting from an editable layer, double-click the rule message on the Rules page for the layer to display rule details. Click **Delete** next to the setting you want to delete, then click **OK** twice.

Step 5 To save changes you made in this policy since the last policy commit, click **Policy Information**, then click **Commit Changes**.

If you leave the policy without committing changes, changes since the last commit are discarded if you edit a different policy.

Removing Rule Settings from Multiple Layers

You can simultaneously remove a specific type of event filter, dynamic state, or alerting from multiple layers in your intrusion policy. The system removes the selected setting and copies the remaining settings for the rule to the highest editable layer in the policy.

The system removes the setting type downward through each layer where it is set until it removes all the settings or encounters a layer where a rule state is set for the rule. In the latter case, it removes the setting from that layer and stops removing the setting type.

When the system encounters the setting type in a shared layer or in the base policy, and if the highest layer in the policy is editable, the system copies the remaining settings and rule state for the rule to that editable layer. Otherwise, if the highest layer in the policy is a shared layer, the system creates a new editable layer above the shared layer and copies the remaining settings and rule state for the rule to that editable layer.

Note Removing rule settings derived from a shared layer or the base policy causes any changes to this rule from lower layers or the base policy to be ignored. To stop ignoring changes from lower layers or the base policy, set the rule state to **Inherit** on the summary page for the topmost layer.

Procedure

Step 1 While editing your Snort 2 intrusion policy, click **Rules** immediately beneath **Policy Information** in the navigation panel. To access your Snort 2 policy, choose **Policies** > **Intrusion** > **Intrusion Policies tab** and then click **Snort 2** against the policy you want to edit.

Tip
You can also choose **Policy** from the layer drop-down list on the Rules page for any layer, or click **Manage Rules** on the Policy Information page.

Step 2 Choose the rule or rules from which you want to remove multiple settings:

- Choose specific — If you want to choose specific rules, check the check box next to each rule.
- Choose all — If you want to choose all the rules in the current list, check the check box at the top of the column.

Step 3 Choose one of the following options:

- **Event Filtering** > **Remove Thresholds**
- **Event Filtering** > **Remove Suppressions**
- **Dynamic State** > **Remove Rate-Based Rule States**
- **Alerting** > **Remove SNMP Alerts**

Note
Removing rule settings derived from a shared layer or the base policy causes any changes to this rule from lower layers or the base policy to be ignored. To stop ignoring changes from lower layers or the base policy, set the rule state to **Inherit** on the summary page for the topmost layer.

Step 4 Click **OK**.

Step 5 To save changes you made in this policy since the last policy commit, click **Policy Information**, then click **Commit Changes**.

If you leave the policy without committing changes, changes since the last commit are discarded if you edit a different policy.

What to do next

- Deploy configuration changes.

Accepting Rule Changes from a Custom Base Policy

When a custom network analysis or intrusion policy where you have not added layers uses another custom policy as its base policy, you must set a rule to inherit its rule state if:

- you delete an event filter, dynamic state, or SNMP alert that is set for the rule in the base policy, *and*
- you want the rule to accept subsequent changes that you make to it in the other custom policy that you use as your base policy

Procedure

Step 1 While editing your Snort 2 intrusion policy, expand **Policy Layers** in the navigation panel.

Step 2 Expand **My Changes**.

Step 3 Click the **Rules** link immediately beneath **My Changes**.

Step 4 Choose the rule or rules whose settings you want to accept. You have the following choices:

- Choose specific rules — If you want to choose specific rules, check the check box next to each rule.
- Choose all rules — If you want to choose all the rules in the current list, check the check box at the top of the column.

Step 5 Choose **Inherit** from the **Rule State** drop-down list.

Step 6 To save changes you made in this policy since the last policy commit, click **Policy Information**, then click **Commit Changes**.

If you leave the policy without committing changes, changes since the last commit are discarded if you edit a different policy.

What to do next

- Deploy configuration changes.

Preprocessors and Advanced Settings in Layers

You use similar mechanisms to configure preprocessors in a network analysis policy and advanced settings in an intrusion policy. You can enable and disable preprocessors on the network analysis Settings page and intrusion policy advanced settings on the intrusion policy Advanced Settings page. These pages also provide summaries of the effective states for all relevant features. For example, if the network analysis SSL preprocessor is disabled in one layer and enabled in a higher layer, the Settings page shows it as enabled. Changes made on these pages appear in the top layer of the policy. Note that the Back Orifice preprocessor has no user-configurable options.

You can also enable or disable preprocessors or advanced settings and access their configuration pages on the summary page for a user-configurable layer. On this page you can modify the layer name and description and configure whether to share the layer with other policies of the same type. You can switch to the summary page for another layer by selecting the layer name beneath **Policy Layers** in the navigation panel.

When you enable a preprocessor or advanced setting, a sublink to the configuration page for that feature appears beneath the layer name in the navigation panel, and an **Edit** (✎) appears next to the feature on the summary page for the layer; these disappear when you disable the feature in the layer or set it to Inherit.

Setting the state (enabled or disabled) for a preprocessor or advanced setting overrides the state and configuration settings for that feature in lower layers. If you want a preprocessor or advanced setting to inherit its state and configuration from the base policy or a lower layer, set it to **Inherit**. Note that the Inherit selection is not available when you are working in the Settings or Advanced Settings page. Note also that if you inherit a feature that is currently enabled, the feature sublink in the navigation panel and the edit icon on the configuration page no longer appear.

The system uses the configuration in the highest layer where the feature is enabled. Unless you explicitly modify the configuration, the system uses the default configuration. For example, if you enable and modify the network analysis DCE/RPC preprocessor in one layer, and you also enable but do not modify it in a higher layer, the system uses the default configuration in the higher layer.

Color-coding on each layer summary page indicates whether the effective configuration is in a higher, lower, or the current layer, as follows:

- red—the effective configuration is in a higher layer
- yellow—the effective configuration is in a lower layer
- unshaded—the effective configuration is in the current layer

Because the Settings and Advanced Settings pages are composite views of all relevant settings, these page do not use color coding to indicate the positions of effective configurations.

Configuring Preprocessors and Advanced Settings in Layers

Procedure

Step 1 While editing your Snort 2 policy, expand **Policy Layers** in the navigation panel, then click the name of the layer you want to modify.

Step 2 You have the following choices:

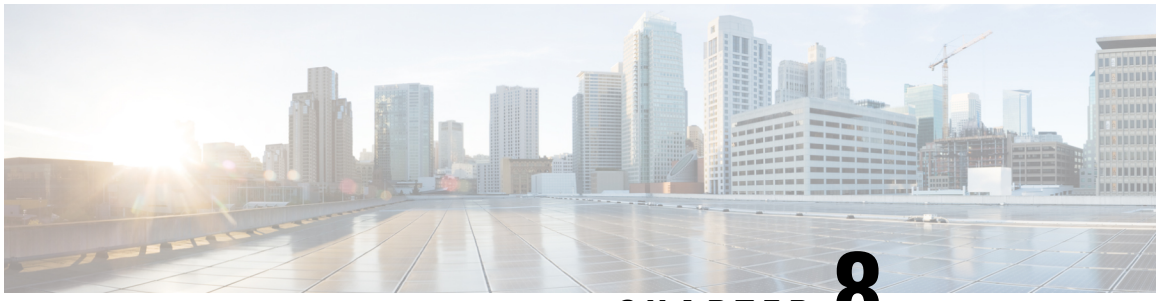
- Change the layer **Name**.
- Add or change the **Description**.
- Check or clear the **Sharing** check box to specify whether a layer can be shared with another policy.
- To access the configuration page for an enabled preprocessor/advanced setting, click **Edit** (✎) or the feature sublink.
- To disable a preprocessor/advanced setting in the current layer, click **Disabled** next to the feature.
- To enable a preprocessor/advanced setting in the current layer, click **Enabled** next to the feature.
- To inherit the preprocessor/advanced setting state and configuration from the settings in the highest layer below the current layer, click **Inherit**.

Step 3 To save changes you made in this policy since the last policy commit, click **Policy Information**, then click **Commit Changes**.

If you leave the policy without committing changes, changes since the last commit are discarded if you edit a different policy.

What to do next

- Deploy configuration changes.



CHAPTER 8

Sensitive Data Detection

The following topics explain sensitive data detection and how to configure it:

Sensitive Data Detection Basics

Sensitive data such as Social Security numbers, credit card numbers, driver's license numbers, and so on may be leaked onto the Internet, intentionally or accidentally. The system provides a sensitive data preprocessor that can detect and generate events on sensitive data in ASCII text, which can be particularly useful in detecting accidental data leaks.

Global sensitive data preprocessor options control how the preprocessor functions. You can modify global options that specify the following:

- whether the preprocessor replaces all but the last four credit card or Social Security numbers in triggering packets
- which destination hosts on your network to monitor for sensitive data
- how many total occurrences of all data types in a single session result in an event

Individual data types identify the sensitive data you can detect and generate events on in your specified destination network traffic. You can modify default settings for data type options that specify the following:

- a threshold that must be met for a detected data type to generate a single per-session event
- the destination ports to monitor for each data type

- the application protocols to monitor for each data type

You can create and modify custom data types to detect data patterns that you specify. For example, a hospital might create a data type to protect patient numbers, or a university might create a data type to detect student numbers that have a unique numbering pattern.

The system detects sensitive data per TCP session by matching individual data types against traffic. You can modify the default settings for each data type and for global options that apply to all data types in your intrusion policy. The system provides predefined, commonly used data types. You can also create custom data types.

A sensitive data preprocessor rule is associated with each data type. You enable sensitive data detection and event generation for each data type by enabling the corresponding preprocessor rule for the data type. A link on the configuration page takes you to a filtered view of sensitive data rules on the Rules page, where you can enable and disable rules and configure other rule attributes.

When you save changes to your intrusion policy, you are given the option to automatically enable the sensitive data preprocessor if the rule associated with a data type is enabled and sensitive data detection is disabled.

Tip The sensitive data preprocessor can detect sensitive data in unencrypted Microsoft Word files that are uploaded and downloaded using FTP or HTTP; this is possible because of the way Word files group ASCII text and formatting commands separately.

The system does not detect encrypted or obfuscated sensitive data, or sensitive data in a compressed or encoded format such as a Base64-encoded email attachment. For example, the system would detect the phone number (555)123-4567, but not an obfuscated version where each number is separated by spaces, as in (5 5 5) 1 2 3 - 4 5 6 7, or by intervening HTML code, such as (555)-123-4567. However, the system would detect, for example, the HTML coded number (555)-123-4567 where no intervening codes interrupt the numbering pattern.

Global Sensitive Data Detection Options

Global sensitive data options are policy-specific and apply to all data types.

Mask

Replaces with Xs all but the last four digits of credit card numbers and Social Security numbers in the triggering packet. The masked numbers appear in the intrusion event packet view in the web interface and in downloaded packets.

Networks

Specifies the destination host or hosts to monitor for sensitive data. You can specify a single IP address, address block, or a comma-separated list of either or both. The system interprets a blank field as `any`, meaning any destination IP address.

Global Threshold

Specifies the total number of all occurrences of all data types during a single session that the preprocessor must detect in any combination before generating a global threshold event. You can specify 1 through 65535.

Cisco recommends that you set the value for this option higher than the highest threshold value for any individual data type that you enable in your policy.

Note the following points regarding global thresholds:

- You must enable preprocessor rule 139:1 to detect and generate events and, in an inline deployment, drop offending packets on combined data type occurrences.
- The preprocessor generates up to one global threshold event per session.
- Global threshold events are independent of individual data type events; that is, the preprocessor generates an event when the global threshold is reached, regardless of whether the event threshold for any individual data type has been reached, and vice versa.

Individual Sensitive Data Type Options

At a minimum, each custom data type must specify an event threshold and at least one port or application protocol to monitor.

Each system-provided data type uses an otherwise inaccessible `sd_pattern` keyword to define a built-in data pattern to detect in traffic. You can also create custom data types for which you use simple regular expressions to specify your own data patterns.

Sensitive data types display in all intrusion policies where Sensitive Data Detection is enabled. System-provided data types display as read-only. For custom data types, the name and pattern fields display as read-only, but you can set the other options to policy-specific values.

Table 84: Individual Data Type Options

Option	Description
Data Type	Specifies the unique name for the data type.
Threshold	Specifies the number of occurrences of the data type when the system generates an event. You can specify 1 through 255. Note that the preprocessor generates one event for a detected data type per session. Note also that global threshold events are independent of individual data type events; that is, the preprocessor generates an event when the data type event threshold is reached, regardless of whether the global event threshold has been reached, and vice versa.
Destination Ports	Specifies destination ports to monitor for the data type. You can specify a single port, a comma-separated list of ports, or `any`, meaning any destination port.
Application Protocols	Specifies up to eight application protocols to monitor for the data type. You must activate application detectors to identify application protocols to monitor. Note that, for Classic devices, this feature requires a Control license.
Pattern	Specifies the pattern to detect. This field is only present for custom data types.

System-Provided Sensitive Data Types

Each intrusion policy includes system-provided data types for detecting commonly used data patterns such as credit card numbers, email addresses, U.S. phone numbers, and U.S. Social Security numbers with and without dashes.

Each system-provided data type is associated with a single sensitive data preprocessor rule that has a generator ID (GID) of 138. You must enable the associated sensitive data rule in the intrusion policy to generate events and, in an inline deployment, drop offending packets for each data type that you want to use in your policy.

The following table describes each data type and lists the corresponding preprocessor rule.

Table 85: System-Provided Sensitive Data Types

Data Type	Description	Preprocessor GID:SID
Credit Card Numbers	Matches Visa®, MasterCard®, Discover® and American Express® fifteen- and sixteen-digit credit card numbers, with or without their normal separating dashes or spaces; also uses the Luhn algorithm to verify credit card check digits.	138:2
Email Addresses	Matches email addresses.	138:5
U.S. Phone Numbers	Matches U.S. phone numbers adhering to the pattern `(\d{3}) ?\d{3}-\d{4}`.	138:6
U.S. Social Security Numbers Without Dashes	Matches 9-digit U.S. Social Security numbers that have valid 3-digit area numbers, valid 2-digit group numbers, and do not have dashes.	138:4
U.S. Social Security Numbers With Dashes	Matches 9-digit U.S. Social Security numbers that have valid 3-digit area numbers, valid 2-digit group numbers, and dashes.	138:3

To reduce false positives from 9-digit numbers other than Social Security numbers, the preprocessor uses an algorithm to validate the 3-digit area number and 2-digit group number that precede the 4-digit serial number in each Social Security number. The preprocessor validates Social Security group numbers through November 2009.

License Requirements for Sensitive Data Detection

Threat Defense License

IPS

Requirements and Prerequisites for Sensitive Data Detection

Model support

Any.

Supported domains

Any

User roles

- Admin
- Intrusion Admin

Configuring Sensitive Data Detection

Because sensitive data detection can have a high impact on the performance of your system, Cisco recommends that you adhere to the following guidelines:

- Choose the No Rules Active default policy as your base intrusion policy.
- Ensure that the following settings are enabled in the corresponding network analysis policy:
  - **FTP and Telnet Configuration** under **Application Layer Preprocessors**
  - **IP Defragmentation** and **TCP Stream Configuration** under **Transport/Network Layer Preprocessors**.

Procedure

Step 1 Choose **Policies** > **Access Control heading** > **Intrusion**

Step 2 Click **Snort 2 Version** next to the policy you want to edit.

If **View** ( ) appears instead, the configuration belongs to an ancestor domain, or you do not have permission to modify the configuration.

Step 3 Click **Advanced Settings** in the navigation panel.

Step 4 If **Sensitive Data Detection** under **Specific Threat Detection** is disabled, click **Enabled**.

Step 5 Click **Edit** ( ) next to **Sensitive Data Detection**.

Step 6 You have the following choices:

- Modify the global settings as described in Global Sensitive Data Detection Options, on page 212.
- Choose a data type in the **Targets** section, and modify the data type configuration as described in Individual Sensitive Data Type Options, on page 213.
- If you want to inspect custom sensitive data, create a custom data type; see Custom Sensitive Data Types, on page 218.

Step 7 Add or remove application protocols to monitor for a data type; see Monitored Application Protocols and Sensitive Data, on page 216.

Note
To detect sensitive data in FTP traffic:

- Ensure that the file policy is enabled for the access control policy.

- You must add the `Ftp data` application protocol.

Step 8 Optionally, to display sensitive data preprocessor rules, click **Configure Rules for Sensitive Data Detection**.

You can enable or disable any of the listed rules. You can also configure sensitive data rules for any of the other actions available on the Rules page, such as rule suppression, rate-based attack prevention, and so on.

Step 9 To save changes you made in this policy since the last policy commit, click **Policy Information** in the navigation panel, then click **Commit Changes**.

If you enable sensitive data preprocessor rules in your policy without enabling sensitive data detection, you are prompted to enable sensitive data detection when you save changes to your policy.

If you leave the policy without committing changes, changes since the last commit are discarded if you edit a different policy.

What to do next

- If you want to generate intrusion events, enable Sensitive Data Detection rules 138:2, 138:3, 138:4, 138:5, 138:6, 138:>999999, or 139:1. For more information, see Intrusion Rule States, on page 42, Global Sensitive Data Detection Options, on page 212, System-Provided Sensitive Data Types, on page 214, and Custom Sensitive Data Types, on page 218.

Monitored Application Protocols and Sensitive Data

You can specify up to eight application protocols to monitor for each data type. At least one detector must be enabled for each application protocol you select. By default, all system-provided detectors are activated. If no detector is enabled for an application protocol, the system automatically enables all system-provided detectors for the application; if none exist, the system enables the most recently modified user-defined detector for the application.

You must specify at least one application protocol or port to monitor for each data type. However, except in the case where you want to detect sensitive data in FTP traffic, Cisco recommends for the most complete coverage that you specify corresponding ports when you specify application protocols. For example, if you specify HTTP, you might also configure the well-known HTTP port 80. If a new host on your network implements HTTP, the system monitors port 80 during the interval when it is discovering the new HTTP application protocol.

In the case where you want to detect sensitive data in FTP traffic, you must specify the `FTP data` application protocol; there is no advantage in specifying a port number.

Selecting Application Protocols to Monitor

You can specify application protocols to monitor in both system-provided and custom sensitive data types. The application protocols you select are policy-specific.

Before you begin

For classic devices, this procedure requires the Control license.

Procedure

Step 1 Choose **Policies** > **Access Control heading** > **Intrusion**.

Step 2 Click **Snort 2 Version** next to the policy you want to edit.

If **View** ( ) appears instead, the configuration belongs to an ancestor domain, or you do not have permission to modify the configuration.

Step 3 Click **Advanced Settings** in the navigation panel.

Step 4 If **Sensitive Data Detection** under **Specific Threat Detection** is disabled, click **Enabled**.

Step 5 Click **Edit** ( ) next to **Sensitive Data Detection**.

Step 6 Click the name of a data type under **Data Types**.

Step 7 Click **Edit** ( ) next to the **Application Protocols** field.

Step 8 You have the following choices:

- To add application protocols for monitoring, choose one or more application protocols from the **Available** list, then click right arrow (>). You can add up to eight application protocols for monitoring.
- To remove an application protocol from monitoring, choose it from the **Enabled** list, then click left arrow (<).

Step 9 Click **OK**.

Step 10 To save changes you made in this policy since the last policy commit, click **Policy Information** in the navigation pane, then click **Commit Changes**.

If you leave the policy without committing changes, changes since the last commit are discarded if you edit a different policy.

What to do next

- Deploy configuration changes.

Special Case: Sensitive Data Detection in FTP Traffic

You usually determine which traffic to monitor for sensitive data by specifying the ports to monitor or specifying application protocols in deployments.

However, specifying ports or application protocols is not sufficient for detecting sensitive data in FTP traffic. Sensitive data in FTP traffic is found in traffic for the FTP application protocol, which occurs intermittently and uses a transient port number, making it difficult to detect. To detect sensitive data in FTP traffic, you **must** include the following in your configuration:

- Specify the `FTP data` application protocol to enable detection of sensitive data in FTP traffic.

In the special case of detecting sensitive data in FTP traffic, specifying the `FTP data` application protocol does not invoke detection; instead, it invokes the rapid processing of the FTP/Telnet processor to detect sensitive data in FTP traffic.

- Ensure that the FTP Data detector, which is enabled by default, is enabled.
- Ensure that your configuration includes at least one port to monitor for sensitive data.
- Ensure that the file policy is enabled for the Access Control Policy.

Note that it is not necessary to specify an FTP port except in the unlikely case where you only want to detect sensitive data in FTP traffic. Most sensitive data configurations will include other ports such as HTTP or email ports. In the case where you do want to specify only one FTP port and no other ports to monitor, Cisco recommends that you specify the FTP command port `23`.

Custom Sensitive Data Types

Each custom data type you create also creates a single sensitive data preprocessor rule that has a Generator ID (GID) of 138 and a Snort ID (SID) of 1000000 or greater, that is, a SID for a local rule.

You must enable the associated sensitive data rule to enable detection, generate events and, in an inline deployment, drop offending packets for each custom data type that you want to use in your policy.

To help you enable sensitive data rules, a link on the configuration page takes you to a filtered view of the intrusion policy Rules page that displays all system-provided and custom sensitive data rules. You can also display custom sensitive data rules along with any custom local rules by choosing the local filtering category on the intrusion policy Rules page. Note that custom sensitive data rules are not listed on the intrusion rules editor page (**Objects** > **Intrusion Rules**).

Once you create a custom data type, you can enable it in any intrusion policy in the system. To enable a custom data type, you must enable the associated sensitive data rule in any policy that you want to use to detect that custom data type.

Data Patterns in Custom Sensitive Data Types

You define the data pattern for a custom data type using a simple set of regular expressions comprised of the following:

- three metacharacters
- escaped characters that allow you to use the metacharacters as literal characters
- six character classes

Metacharacters are literal characters that have special meaning within regular expressions.

Table 86: Sensitive Data Pattern Metacharacters

Metacharacter	Description	Example
`?`	Matches zero or one occurrence of the preceding character or escape sequence; that is, the preceding character or escape sequence is optional.	`colou?r` matches `color` or `colour`

Metacharacter	Description	Example
{n}	Matches the preceding character or escape sequence n times.	For example, \d{2} matches 55, 12, and so on; \l{3} matches AbC, www, and so on; \w{3} matches a1B, 25C, and so on; x{5} matches xxxxx
\	Allows you to use metacharacters as actual characters and is also used to specify a predefined character class.	\? matches a question mark, \\ matches a backslash, \d matches numeric characters, and so on

You must use a backslash to escape certain characters for the sensitive data preprocessor to interpret them correctly as literal characters.

Table 87: Escaped Sensitive Data Pattern Characters

Use this escaped character...	To represent this literal character...
\?	?
\{	{
\}	}
\\	\

When defining a custom sensitive data pattern, you can use character classes.

Table 88: Sensitive Data Pattern Character Classes

Character Class	Description	Character Class Definition
\d	Matches any numeric ASCII character 0-9	0-9
\D	Matches any byte that is not a numeric ASCII character	not 0-9
\l (lowercase “ell”)	Matches any ASCII letter	a-zA-Z
\L	Matches any byte that is not an ASCII letter	not a-zA-Z
\w	Matches any ASCII alphanumeric character Note that, unlike PCRE regular expressions, this does not include an underscore (_).	a-zA-Z0-9
\W	Matches any byte that is not an ASCII alphanumeric character	not a-zA-Z0-9

The preprocessor treats characters entered directly, instead of as part of a regular expression, as literal characters. For example, the data pattern 1234 matches 1234.

The following data pattern example, which is used in system-provided sensitive data rule 138:4, uses the escaped digits character class, the multiplier and option-specifier metacharacters, and the literal dash (-) and left and right parentheses () characters to detect U.S. phone numbers:

```
(\d{3}) ?\d{3}-\d{4}
```

Exercise caution when creating custom data patterns. Consider the following alternative data pattern for detecting phone numbers which, although using valid syntax, could cause many false positives:

```
(?\d{3})? ?\d{3}-?\d{4}
```

Because the second example combines optional parentheses, optional spaces, and optional dashes, it would detect, among others, phone numbers in the following desirable patterns:

- `(555)123-4567`
- `555123-4567`
- `5551234567`

However, the second example pattern would also detect, among others, the following potentially invalid patterns, resulting in false positives:

- `(555 1234567`
- `555)123-4567`
- `555) 123-4567`

Consider finally, for illustration purposes only, an extreme example in which you create a data pattern that detects the lowercase letter `a` using a low event threshold in all destination traffic on a small company network. Such a data pattern could overwhelm your system with literally millions of events in only a few minutes.

Configuring Custom Sensitive Data Types

You cannot delete a data type if the sensitive data rule for that data type is enabled in any intrusion policy.

Procedure

Step 1 Choose **Policies** > **Access Control heading** > **Intrusion**

Step 2 Click **Snort 2 Version** next to the policy you want to edit.

If **View** ( ) appears instead, the configuration belongs to an ancestor domain, or you do not have permission to modify the configuration.

Step 3 Click **Advanced Settings** in the navigation panel.

Step 4 If **Sensitive Data Detection** under **Specific Threat Detection** is disabled, click **Enabled**.

Step 5 Click **Edit** ( ) next to **Sensitive Data Detection**.

Step 6 Click **Add** ( ) next to **Data Types**.

Step 7 Enter a name for the data type.

Step 8 Enter the pattern you want to detect with this data type; see Data Patterns in Custom Sensitive Data Types, on page 218.

Step 9 Click **OK**.

Step 10 Optionally, click the data type name, and modify the options described in Individual Sensitive Data Type Options, on page 213.

Step 11 Optionally, delete a custom data type by clicking **Delete** (), then **OK** to confirm.

Note
If the sensitive data rule for that data type is enabled in any intrusion policy, the system warns that you cannot delete the data type. You must disable the sensitive data rule in affected policies before attempting the deletion again; see Setting Intrusion Rule States, on page 42.

Step 12 To save changes you made in this policy since the last policy commit, click **Policy Information** in the navigation panel, then click **Commit Changes**.

If you leave the policy without committing changes, changes since the last commit are discarded if you edit a different policy.

What to do next

- Enable the associated custom sensitive data preprocessing rule in each policy where you want to use that data type; see Setting Intrusion Rule States, on page 42.
- Deploy configuration changes.

Editing Custom Sensitive Data Types

You can edit all fields in custom sensitive data types. Note, however, that when you modify the name or pattern field, these settings change in all intrusion policies on the system. You can set the other options to policy-specific values.

Procedure

Step 1 Choose **Policies** > **Access Control heading** > **Intrusion**

Step 2 Click **Snort 2 Version** next to the policy you want to edit.

If **View** () appears instead, the configuration belongs to an ancestor domain, or you do not have permission to modify the configuration.

Step 3 Click **Advanced Settings** in the navigation panel.

Step 4 If **Sensitive Data Detection** under **Specific Threat Detection** is disabled, click **Enabled**.

Step 5 Click **Edit** next to **Sensitive Data Detection**.

Step 6 In the **Targets** section, click the name of the custom data type.

Step 7 Click **Edit Data Type Name and Pattern**.

Step 8 Modify the data type name and pattern; see Data Patterns in Custom Sensitive Data Types, on page 218.

Step 9 Click **OK**.

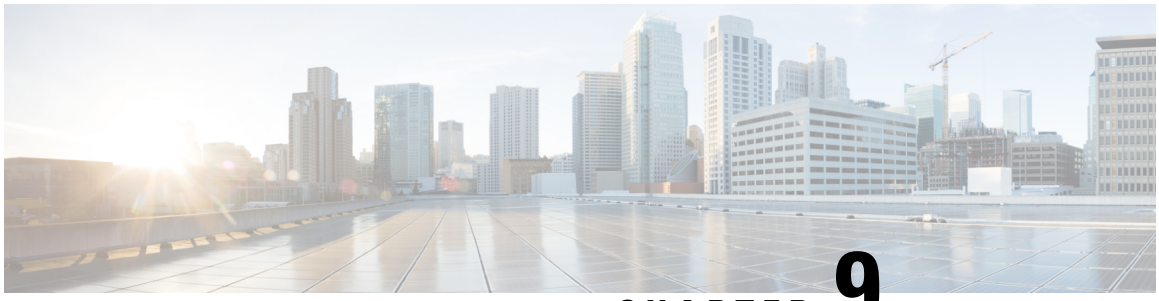
Step 10 Set the remaining options to policy-specific values; see Individual Sensitive Data Type Options, on page 213.

Step 11 To save changes you made in this policy since the last policy commit, click **Policy Information** in the navigation panel, then click **Commit Changes**.

If you leave the policy without committing changes, changes since the last commit are discarded if you edit a different policy.

What to do next

- Deploy configuration changes.



CHAPTER 9

Global Limit for Intrusion Event Logging

The following topics describe how to globally limit intrusion event logging:

Global Rule Thresholding Basics

The global rule threshold sets limits for event logging by an intrusion policy. You can set a global rule threshold across all traffic to limit how often the policy logs events from a specific source or destination and displays those events per specified time period. You can also set thresholds per shared object rule, standard text rule, or preprocessor rule in the policy. When you set a global threshold, that threshold applies for each rule in the policy that does not have an overriding specific threshold. Thresholds can prevent you from being overwhelmed with a large number of events.

Every intrusion policy contains a default global rule threshold that applies by default to all intrusion rules and preprocessor rules. This default threshold limits the number of events on traffic going to a destination to one event per 60 seconds.

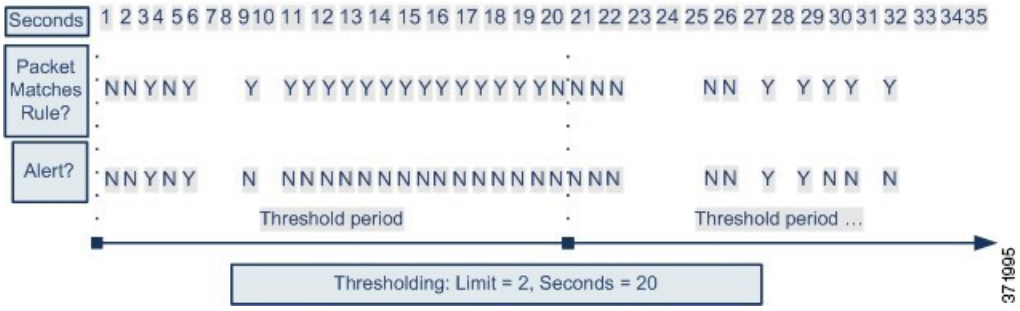
You can:

- Change the global threshold.
- Disable the global threshold.
- Override the global threshold by setting individual thresholds for specific rules.

  For example, you might set a global limit threshold of five events every 60 seconds, but then set a specific threshold of ten events for every 60 seconds for SID 1315. All other rules generate no more than five events in each 60-second period, but the system generates up to ten events for each 60-second period for SID 1315.

Tip A global or individual threshold on a managed device with multiple CPUs may result in a higher number of events than expected.

The following diagram demonstrates how the global rule thresholding works. In this example, an attack is in progress for a specific rule. The global limit threshold is set to limit event generation for each rule to two events every 20 seconds. Note that the period starts at one second and ends at 21 seconds. After the period ends, the cycle starts again and the next two rule matches generate events, then the system does not generate any more events during that period.



Global Rule Thresholding Options

The default threshold limits event generation for each rule to one event every 60 seconds on traffic going to the same destination. The default values for the global rule thresholding options are:

- **Type** — Limit
- **Track By** — Destination
- **Count** — 1
- **Seconds** — 60

You can modify these default values as follows:

Table 89: Thresholding Types

Option	Description
Limit	Logs and displays events for the specified number of packets (specified by the count argument) that trigger the rule during the specified time period. For example, if you set the type to **Limit**, the **Count** to `10`, and the **Seconds** to `60`, and 14 packets trigger the rule, the system stops logging events for the rule after displaying the first 10 that occur within the same minute.
Threshold	Logs and displays a single event when the specified number of packets (specified by the count argument) trigger the rule during the specified time period. Note that the counter for the time restarts after you hit the threshold count of events and the system logs that event. For example, you set the type to **Threshold**, **Count** to `10`, and **Seconds** to `60`, and the rule triggers 10 times by second 33. The system generates one event, then resets the Seconds and Count counters to `0`. The rule then triggers another 10 times in the next 25 seconds. Because the counters reset to `0` at second 33, the system logs another event.

Option	Description
Both	Logs and displays an event once per specified time period, after the specified number (count) of packets trigger the rule. For example, if you set the type to **Both**, **Count** to `2`, and **Seconds** to `10`, the following event counts result: • If the rule is triggered once in 10 seconds, the system does not generate any events (the threshold is not met) • If the rule is triggered twice in 10 seconds, the system generates one event (the threshold is met when the rule triggers the second time) • If the rule is triggered four times in 10 seconds, the system generates one event (the threshold is met when the rule triggered the second time and following events are ignored)

The **Track By** option determines whether the event instance count is calculated per source or destination IP address.

You can also specify the number of instances and time period that define the threshold, as follows:

Table 90: Thresholding Instance/Time Options

Option	Description
Count	For a **Limit** threshold, the number of event instances per specified time period per tracking IP address or address range required to meet the threshold. For a **Threshold** threshold, the number of rule matches you want to use as your threshold.
Seconds	For a **Limit** threshold, the number of seconds that make up the time period when attacks are tracked. For a **Threshold** threshold, the number of seconds that elapse before the count resets. If you set the threshold type to **Limit**, the tracking to **Source**, **Count** to 10, and **Seconds** to 10, the system logs and displays the first 10 events that occur in 10 seconds from a given source port. If only seven events occur in the first 10 seconds, the system logs and displays those, if 40 events occur in the first 10 seconds, the system logs and displays 10, then begins counting again when the 10-second time period elapses.

License Requirements for Global Thresholds

Threat Defense License

IPS

Requirements and Prerequisites for Global Thresholds

Model support

Any

Supported domains

Any

User roles

- Admin
- Intrusion Admin

Configuring Global Thresholds

Procedure

Step 1 Choose **Policies** > **Access Control heading** > **Intrusion**.

Step 2 Click **Snort 2 Version** next to the policy you want to edit.

If **View** () appears instead, the configuration belongs to an ancestor domain, or you do not have permission to modify the configuration.

Step 3 Click **Advanced Settings** in the navigation panel.

Step 4 If **Global Rule Thresholding** under **Intrusion Rule Thresholds** is disabled, click **Enabled**.

Step 5 Click **Edit** () next to **Global Rule Thresholding**.

Step 6 Using **Type**, specify the type of threshold that will apply over the time you specify in the **Seconds** field.

Step 7 Using **Track By**, specify the tracking method.

Step 8 Enter a value in the **Count** field.

Step 9 Enter a value in the **Seconds** field.

Step 10 To save changes you made in this policy since the last policy commit, click **Policy Information**, then click **Commit Changes**.

If you leave the policy without committing changes, changes since the last commit are discarded if you edit a different policy.

What to do next

- Deploy configuration changes.

Disabling the Global Threshold

You can disable global thresholding in the highest policy layer if you want to threshold events for specific rules rather than applying thresholding to every rule by default.

Procedure

Step 1 Choose **Policies** > **Access Control heading** > **Intrusion**

Step 2 Click **Snort 2 Version** next to the policy you want to edit.

If **View** () appears instead, the configuration belongs to an ancestor domain, or you do not have permission to modify the configuration.

Step 3 Click **Advanced Settings** in the navigation panel.

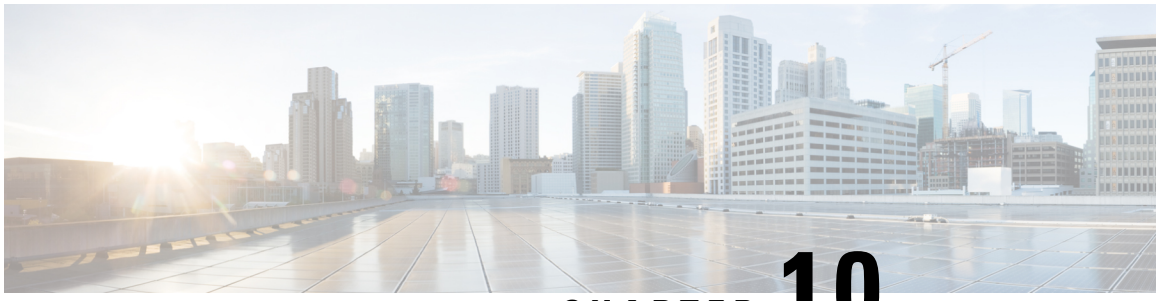
Step 4 Next to **Global Rule Thresholding** under **Intrusion Rule Thresholds**, click **Disabled**.

Step 5 To save changes you made in this policy since the last policy commit, click **Policy Information**, then click **Commit Changes**.

If you leave the policy without committing changes, changes since the last commit are discarded if you edit a different policy.

What to do next

- Deploy configuration changes.



CHAPTER 10

Application Layer Preprocessors

The following topics explain application layer preprocessors and how to configure them:

Introduction to Application Layer Preprocessors

Note This section applies to Snort 2 preprocessors. For information on Snort 3 inspectors, see https://www.cisco.com/go/snort3-inspectors.

Application layer protocols can represent the same data in a variety of ways. The system provides application layer protocol decoders that normalize specific types of packet data into formats that the intrusion rules engine can analyze. Normalizing application-layer protocol encodings allows the rules engine to effectively apply the same content-related rules to packets whose data is represented differently and obtain meaningful results.

When an intrusion rule or rule argument requires a disabled preprocessor, the system automatically uses it with its current configuration even though it remains disabled in the network analysis policy's web interface.

Note that preprocessors do not generate events in most cases unless you enable the accompanying preprocessor rules in an intrusion policy.

License Requirements for Application Layer Preprocessors

Threat Defense License

IPS

Requirements and Prerequisites for Application Layer Preprocessors

Model support

Any.

Supported domains

Any

User roles

- Admin
- Intrusion Admin

The DCE/RPC Preprocessor

Note This section applies to Snort 2 preprocessors. For information on Snort 3 inspectors, see https://www.cisco.com/go/snort3-inspectors.

The DCE/RPC protocol allows processes on separate network hosts to communicate as if the processes were on the same host. These inter-process communications are commonly transported between hosts over TCP and UDP. Within the TCP transport, DCE/RPC might also be further encapsulated in the Windows Server Message Block (SMB) protocol or in Samba, an open-source SMB implementation used for inter-process communication in a mixed environment comprised of Windows and UNIX- or Linux-like operating systems. In addition, Windows IIS web servers on your network might use IIS RPC over HTTP, which provides distributed communication through a firewall, to proxy TCP-transported DCE/RPC traffic.

Note that descriptions of DCE/RPC preprocessor options and functionality include the Microsoft implementation of DCE/RPC known as MSRPC; descriptions of SMB options and functionality refer to both SMB and Samba.

Although most DCE/RPC exploits occur in DCE/RPC client requests targeted for DCE/RPC servers, which could be practically any host on your network that is running Windows or Samba, exploits can also occur in server responses. The DCE/RPC preprocessor detects DCE/RPC requests and responses encapsulated in TCP, UDP, and SMB transports, including TCP-transported DCE/RPC using version 1 RPC over HTTP. The preprocessor analyzes DCE/RPC data streams and detects anomalous behavior and evasion techniques in

DCE/RPC traffic. It also analyzes SMB data streams and detects anomalous SMB behavior and evasion techniques.

The DCE/RPC preprocessor also desegments SMB and defragments DCE/RPC in addition to the IP defragmentation provided by the IP defragmentation preprocessor and the TCP stream reassembly provided by the TCP stream preprocessor.

Finally, the DCE/RPC preprocessor normalizes DCE/RPC traffic for processing by the rules engine.

Connectionless and Connection-Oriented DCE/RPC Traffic

DCE/RPC messages comply with one of two distinct DCE/RPC Protocol Data Unit (PDU) protocols:

connection-oriented DCE/RPC PDU protocol

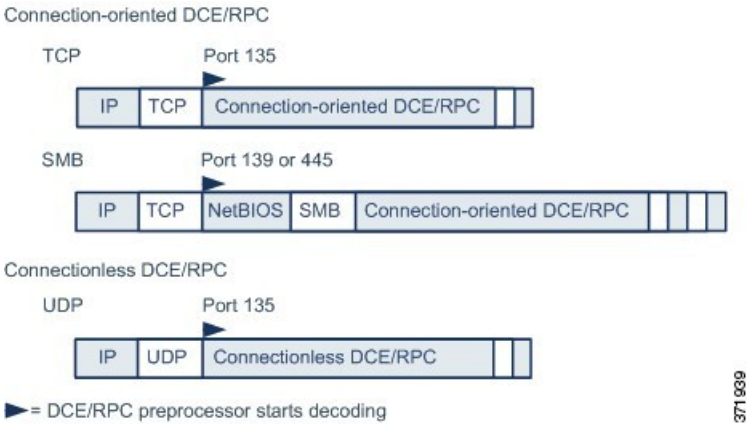
The DCE/RPC preprocessor detects connection-oriented DCE/RPC in the TCP, SMB, and RPC over HTTP transports.

connectionless DCE/RPC PDU protocol

The DCE/RPC preprocessor detects connectionless DCE/RPC in the UDP transport.

The two DCE/RPC PDU protocols have their own unique headers and data characteristics. For example, the connection-oriented DCE/RPC header length is typically 24 bytes and the connectionless DCE/RPC header length is fixed at 80 bytes. Also, correct fragment order of fragmented connectionless DCE/RPC cannot be handled by a connectionless transport and, instead, must be ensured by connectionless DCE/RPC header values; in contrast, the transport protocol ensures correct fragment order for connection-oriented DCE/RPC. The DCE/RPC preprocessor uses these and other protocol-specific characteristics to monitor both protocols for anomalies and other evasion techniques, and to decode and defragment traffic before passing it to the rules engine.

The following diagram illustrates the point at which the DCE/RPC preprocessor begins processing DCE/RPC traffic for the different transports.



Note the following in the figure:

- The well-known TCP or UDP port 135 identifies DCE/RPC traffic in the TCP and UDP transports.
- The figure does not include RPC over HTTP.

  For RPC over HTTP, connection-oriented DCE/RPC is transported directly over TCP as shown in the figure after an initial setup sequence over HTTP.

- The DCE/RPC preprocessor typically receives SMB traffic on the well-known TCP port 139 for the NetBIOS Session Service or the similarly implemented well-known Windows port 445.

  Because SMB has many functions other than transporting DCE/RPC, the preprocessor first tests whether the SMB traffic is carrying DCE/RPC traffic and stops processing if it is not or continues processing if it is.

- IP encapsulates all DCE/RPC transports.
- TCP transports all connection-oriented DCE/RPC.
- UDP transports connectionless DCE/RPC.

DCE/RPC Target-Based Policies

Windows and Samba DCE/RPC implementations differ significantly. For example, all versions of Windows use the DCE/RPC context ID in the first fragment when defragmenting DCE/RPC traffic, and all versions of Samba use the context ID in the last fragment. As another example, Windows Vista uses the opnum (operation number) header field in the first fragment to identify a specific function call, and Samba and all other Windows versions use the opnum field in the last fragment.

There are also significant differences in Windows and Samba SMB implementations. For example, Windows recognizes the SMB OPEN and READ commands when working with named pipes, but Samba does not recognize these commands.

When you enable the DCE/RPC preprocessor, you automatically enable a default target-based policy. Optionally, you can add target-based policies that target other hosts running different Windows or Samba versions. The default target-based policy applies to any host not included in another target-based policy.

In each target-based policy, you can:

- enable one or more transports and specify *detection ports* for each
- enable and specify *auto-detection ports*
- set the preprocessor to detect when there is an attempt to connect to one or more shared SMB resources that you identify
- configure the preprocessor to detect files in SMB traffic and to inspect a specified number of bytes in a detected file
- modify an advanced option that should be modified only by a user with SMB protocol expertise; this option lets you set the preprocessor to detect when a number of chained SMB AndX commands exceed a specified maximum number

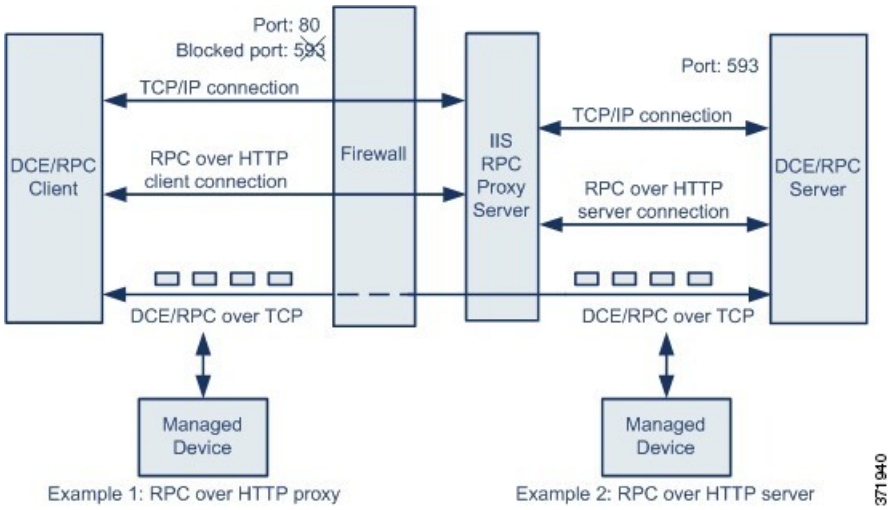
In addition to enabling SMB traffic file detection in the DCE/RPC preprocessor, you can configure a file policy to optionally capture and block these files, or submit them to the Cisco AMP cloud for dynamic analysis. Within that policy, you must create a file rule with an **Action** of **Detect Files** or **Block Files** and a selected **Application Protocol** of **Any** or **NetBIOS-ssn (SMB)**.

Note that for SMB connections, a destination port is assigned as the parent port and rule matching is done as per the rules set on the parent port. To explain this further, consider a scenario in which a control channel is formed on the destination port (for example, port 445). After the control channel is formed, a data channel is formed on another destination port. This means that a parent channel and a child channel is formed. Rule matching is done on the parent channel with the destination port as 445. After this is done, rule matching is not done on the child channel as connection rule matching is done only once. This also results in the creation

of two events on the Management Center - one for the parent channel and one for the child channel. Both these channels are mapped against the same rule ID to which the parent channel is matched.

RPC over HTTP Transport

Microsoft RPC over HTTP allows you to tunnel DCE/RPC traffic through a firewall as shown in the following diagram. The DCE/RPC preprocessor detects version 1 of Microsoft RPC over HTTP.



The Microsoft IIS proxy server and the DCE/RPC server can be on the same host or on different hosts. Separate proxy and server options provide for both cases. Note the following in the figure:

- The DCE/RPC server monitors port 593 for DCE/RPC client traffic, but the firewall blocks port 593.

  Firewalls typically block port 593 by default.
- RPC over HTTP transports DCE/RPC over HTTP using well-known HTTP port 80, which firewalls are likely to permit.
- Example 1 shows that you would choose the **RPC over HTTP proxy** option to monitor traffic between the DCE/RPC client and the Microsoft IIS RPC proxy server.
- Example 2 shows that you would choose the **RPC over HTTP server** option when the Microsoft IIS RPC proxy server and the DCE/RPC server are located on different hosts and the device monitors traffic between the two servers.
- Traffic is comprised solely of connection-oriented DCE/RPC over TCP after RPC over HTTP completes the proxied setup between the DCE/RPC client and server.

DCE/RPC Global Options

Global DCE/RPC preprocessor options control how the preprocessor functions. Note that, except for the **Memory Cap Reached** and **Auto-Detect Policy on SMB Session** options, modifying these options could have a negative impact on performance or detection capability. You should not modify them unless you have a thorough understanding of the preprocessor and the interaction between the preprocessor and enabled DCE/RPC rules.

If no preprocessor rule is mentioned in the following descriptions, the option is not associated with a preprocessor rule.

Maximum Fragment Size

When **Enable Defragmentation** is selected, specifies the maximum DCE/RPC fragment length allowed. The preprocessor truncates larger fragments for processing purposes to the specified size before defragmenting but does not alter the actual packet. A blank field disables this option.

Make sure that the **Maximum Fragment Size** option is greater than or equal to the depth to which the rules need to detect.

Reassembly Threshold

When **Enable Defragmentation** is selected, 0 disables this option, or specifies a minimum number of fragmented DCE/RPC bytes and, if applicable, segmented SMB bytes to queue before sending a reassembled packet to the rules engine. A low value increases the likelihood of early detection but could have a negative impact on performance. You should test for performance impact if you enable this option.

Make sure that the **Reassembly Threshold** option is greater than or equal to the depth to which the rules need to detect.

Enable Defragmentation

Specifies whether to defragment fragmented DCE/RPC traffic. When disabled, the preprocessor still detects anomalies and sends DCE/RPC data to the rules engine, but at the risk of missing exploits in fragmented DCE/RPC data.

Although this option provides the flexibility of not defragmenting DCE/RPC traffic, most DCE/RPC exploits attempt to take advantage of fragmentation to hide the exploit. Disabling this option would bypass most known exploits, resulting in a large number of false negatives.

Memory Cap Reached

Detects when the maximum memory limit allocated to the preprocessor is reached or exceeded. When the maximum memory cap is reached or exceeded, the preprocessor frees all pending data associated with the session that caused the memory cap event and ignores the rest of that session.

You can enable rule 133:1 to generate events and, in an inline deployment, drop offending packets for this option. See Setting Intrusion Rule States, on page 42.

Auto-Detect Policy on SMB Session

Detects the Windows or Samba version that is identified in SMB `Session Setup AndX` requests and responses. When the detected version is different from the Windows or Samba version configured for the **Policy** configuration option, the detected version overrides the configured version for that session only.

For example, if you set **Policy** to Windows XP and the preprocessor detects Windows Vista, the preprocessor uses a Windows Vista policy for that session. Other settings remain in effect.

When the DCE/RPC transport is not SMB (that is, when the transport is TCP or UDP), the version cannot be detected and the policy cannot be automatically configured.

To enable this option, choose one of the following from the drop-down list:

- Choose **Client** to inspect server-to-client traffic for the policy type.

- Choose **Server** to inspect client-to-server traffic for the policy type.
- Choose **Both** to inspect server-to-client and client-to-server traffic for the policy type.

Legacy SMB Inspection Mode

When **Legacy SMB Inspection Mode** is enabled, the system applies SMB intrusion rules only to SMB Version 1 traffic, and applies DCE/RPC intrusion rules to DCE/RPC traffic using SMB Version 1 as a transport. When this option is disabled, the system applies SMB intrusion rules to traffic using SMB Versions 1, 2, and 3, but applies DCE/RPC intrusion rules to DCE/RPC traffic using SMB as a transport only for SMB Version 1.

DCE/RPC Target-Based Policy Options

In each target-based policy, you can enable one or more of the TCP, UDP, SMB, and RPC over HTTP transports. When you enable a transport, you must also specify one or more *detection ports*, that is, ports that are known to carry DCE/RPC traffic.

Cisco recommends that you use the default detection ports, which are either well-known ports or otherwise commonly-used ports for each protocol. You would add detection ports only if you detected DCE/RPC traffic on a non-default port.

You can specify ports for one or more transports in any combination in a Windows target-based policy to match the traffic on your network, but you can only specify ports for the SMB transport in a Samba target-based policy.

Note You must enable at least one DCE/RPC transport in the default target-based policy except when you have added a DCE/RPC target-based policy that has at least one transport enabled. For example, you might want to specify the hosts for all DCE/RPC implementations and not have the default target-based policy deploy to unspecified hosts, in which case you would not enable a transport for the default target-based policy.

Optionally, you can also enable and specify *auto-detection ports*, that is, ports that the preprocessor tests first to determine if they carry DCE/RPC traffic and continues processing only when it detects DCE/RPC traffic.

When you enable auto-detection ports, ensure that they are set to the port range from 1024 to 65535 to cover the entire ephemeral port range.

Note that auto-detection occurs only for ports not already identified by transport detection ports.

It is unlikely that you would enable or specify auto-detection ports for the RPC over HTTP Proxy Auto-Detect Ports option or the SMB Auto-Detect Ports option because there is little likelihood that traffic for either would occur or even be possible except on the specified default detection ports.

Each target-based policy allows you to specify the various options below. If no preprocessor rule is mentioned in the following descriptions, the option is not associated with a preprocessor rule.

Networks

The host IP addresses where you want to deploy the DCE/RPC target-based server policy. Also named the **Server Address** field in the Add Target pop-up window when you add a target-based policy.

You can specify a single IP address or address block, or a comma-separated list of either or both. You can configure up to 255 total profiles including the default policy.

Note that the `default` setting in the default policy specifies all IP addresses on your monitored network segment that are not covered by another target-based policy. Therefore, you cannot and do not need to specify an IP address or CIDR block/prefix length for the default policy, and you cannot leave this setting blank in another policy or use address notation to represent `any` (for example, 0.0.0.0/0 or ::/0).

Policy

The Windows or Samba DCE/RPC implementation used by the targeted host or hosts on your monitored network segment.

Note that you can enable the **Auto-Detect Policy on SMB Session** global option to automatically override the setting for this option on a per session basis when SMB is the DCE/RPC transport.

SMB Invalid Shares

Identifies one or more SMB shared resources the preprocessor will detect when there is an attempt to connect to a shared resource that you specify. You can specify multiple shares in a comma-separated list and, optionally, you can enclose shares in quotes, which was required in previous software versions but is no longer required; for example:

```
"C$", D$, "admin", private
```

The preprocessor detects invalid shares in SMB traffic when you have enabled **SMB Ports**.

Note that in most cases you should append a dollar sign to a drive named by Windows that you identify as an invalid share. For example, identify drive C as C$ or "C$".

Note also that to detect SMB invalid shares, you must also enable **SMB Ports** or **SMB Auto-Detect Ports.**

You can enable rule 133:26 to generate events and, in an inline deployment, drop offending packets for this option. See Setting Intrusion Rule States, on page 42.

SMB Maximum AndX Chain

The maximum number of chained SMB AndX commands to permit. Typically, more than a few chained AndX commands represent anomalous behavior and could indicate an evasion attempt. Specify 1 to permit no chained commands or 0 to disable detecting the number of chained commands.

Note that the preprocessor first counts the number of chained commands and generates an event if accompanying SMB preprocessor rules are enabled and the number of chained commands equals or exceeds the configured value. It then continues processing.

Caution Only someone who is expert in the SMB protocol should modify the setting for the **SMB Maximum AndX Chains** option.

You can enable rule 133:20 to generate events and, in an inline deployment, drop offending packets for this option. See Setting Intrusion Rule States, on page 42.

RPC proxy traffic only

Enabling **RPC over HTTP Proxy Ports** indicates whether detected client-side RPC over HTTP traffic is proxy traffic only or might include other web server traffic. For example, port 80 could carry both proxy and other web server traffic.

When this option is disabled, both proxy and other web server traffic are expected. Enable this option, for example, if the server is a dedicated proxy server. When enabled, the preprocessor tests traffic to determine if it carries DCE/RPC, ignores the traffic if it does not, and continues processing if it does. Note that enabling this option adds functionality only if the **RPC over HTTP Proxy Ports** check box is also enabled.

RPC over HTTP Proxy Ports

Enables detection of DCE/RPC traffic tunneled by RPC over HTTP over each specified port when your managed device is positioned between the DCE/RPC client and the Microsoft IIS RPC proxy server.

When enabled, you can add any ports where you see DCE/RPC traffic, although this is unlikely to be necessary because web servers typically use the default port for both DCE/RPC and other traffic. When enabled, you would not enable **RPC over HTTP Proxy Auto-Detect Ports**, but you would enable the **RPC Proxy Traffic Only** when detected client-side RPC over HTTP traffic is proxy traffic only and does not include other web server traffic.

Note You would rarely, if ever, select this option.

RPC over HTTP Server Ports

Enables detection of DCE/RPC traffic tunneled by RPC over HTTP on each specified port when the Microsoft IIS RPC proxy server and the DCE/RPC server are located on different hosts and the device monitors traffic between the two servers.

Typically, when you enable this option you should also enable **RPC over HTTP Server Auto-Detect Ports** with a port range from 1025 to 65535 for that option even if you are not aware of any proxy web servers on your network. Note that the RPC over HTTP server port is sometimes reconfigured, in which case you should add the reconfigured server port to port list for this option.

TCP Ports

Enables detection of DCE/RPC traffic in TCP on each specified port.

Legitimate DCE/RPC traffic and exploits might use a wide variety of ports, and other ports above port 1024 are common. Typically, when this option is enabled you should also enable **TCP Auto-Detect Ports** with a port range from 1025 to 65535 for that option.

UDP Ports

Enables detection of DCE/RPC traffic in UDP on each specified port.

Legitimate DCE/RPC traffic and exploits might use a wide variety of ports, and other ports above port 1024 are common. Typically, when this option is enabled you should also enable **UDP Auto-Detect Ports** with a port range from 1025 to 65535 for that option.

SMB Ports

Enables detection of DCE/RPC traffic in SMB on each specified port.

You could encounter SMB traffic using the default detection ports. Other ports are rare. Typically, use the default settings.

Note that you can enable the **Auto-Detect Policy on SMB Session** global option to automatically override the policy type configured for a targeted policy on a per session basis when SMB is the DCE/RPC transport.

RPC over HTTP Proxy Auto-Detect Ports

Enables auto-detection of DCE/RPC traffic tunneled by RPC over HTTP on the specified ports when your managed device is positioned between the DCE/RPC client and the Microsoft IIS RPC proxy server.

When enabled, you would typically specify a port range from 1025 to 65535 to cover the entire range of ephemeral ports.

RPC over HTTP Server Auto-Detect Ports

Enables auto-detection of DCE/RPC traffic tunneled by RPC over HTTP on the specified ports when the Microsoft IIS RPC proxy server and the DCE/RPC server are located on different hosts and the device monitors traffic between the two servers.

TCP Auto-Detect Ports

Enables auto-detection of DCE/RPC traffic in TCP on the specified ports.

UDP Auto-Detect Ports

Enables auto-detection of DCE/RPC traffic in UDP on each specified port.

SMB Auto-Detect Ports

Enables auto-detection of DCE/RPC traffic in SMB.

Note You would rarely, if ever, select this option.

SMB File Inspection

Enables inspection of SMB traffic for file detection. You have the following options:

- Select **Off** to disable file inspection.
- Select **Only** to inspect file data without inspecting the DCE/RPC traffic in SMB. Selecting this option can improve performance over inspecting both files and DCE/RPC traffic.
- Select **On** to inspect both files and the DCE/RPC traffic in SMB. Selecting this option can impact performance.

Inspection of SMB traffic for the following is not supported:

- files transferred concurrently in a single TCP or SMB session
- files transferred across multiple TCP or SMB sessions
- files transferred with non-contiguous data, such as when message signing is negotiated
- files transferred with different data at the same offset, overlapping the data
- files opened on a remote client for editing that the client saves to the file server

SMB File Inspection Depth

If **SMB File Inspection** is set to **Only** or **On**, the number of bytes inspected when a file is detected in SMB traffic. Specify one of the following:

- a positive value
- `0` to inspect the entire file
- `-1` to disable file inspection

Enter a value in this field equal to or smaller than the one defined in the File and Malware Settings section of the Advanced tab in your access control policy. If you set a value for this option larger than the one defined for **Limit the number of bytes inspected when doing file type detection**, the system uses the access control policy setting as the functional maximum.

If **SMB File Inspection** is set to **Off**, this field is disabled.

Traffic-Associated DCE/RPC Rules

Most DCE/RPC preprocessor rules trigger against anomalies and evasion techniques detected in SMB, connection-oriented DCE/RPC, or connectionless DCE/RPC traffic. The following table identifies the rules that you can enable for each type of traffic.

Table 91: Traffic-Associated DCE/RPC Rules

Traffic	Preprocessor Rule GID:SID
SMB	133:2 through 133:26, and 133:48 through 133:59
Connection-Oriented DCE/RPC	133:27 through 133:39
Detect Connectionless DCE/RPC	133:40 through 133:43

Configuring the DCE/RPC Preprocessor

Note This section applies to Snort 2 preprocessors. For information on Snort 3 inspectors, see https://www.cisco.com/go/snort3-inspectors.

You configure the DCE/RPC preprocessor by modifying any of the global options that control how the preprocessor functions, and by specifying one or more target-based server policies that identify the DCE/RPC servers on your network by IP address and by either the Windows or Samba version running on them. Target-based policy configuration also includes enabling transport protocols, specifying the ports carrying DCE/RPC traffic to those hosts, and setting other server-specific options.

Before you begin

- Confirm that networks you want to identify in a custom target-based policy match or are a subset of the networks, zones, and VLANs handled by its parent network analysis policy. See Advanced Settings for Network Analysis Policies, on page 81 for more information.

Procedure

Step 1 Choose **Policies** > **Access Control heading** > **Access Control**, and then click **Network Analysis Policy** or **Policies** > **Access Control heading** > **Intrusion**, and then click **Network Analysis Policies**.

Note
If your custom user role limits access to the first path listed here, use the second path to access the policy.

Step 2 Click **Snort 2 Version** next to the policy you want to edit.

Step 3 Click **Edit** (✎) next to the policy you want to edit.

If **View** (👁) appears instead, the configuration belongs to an ancestor domain, or you do not have permission to modify the configuration.

Step 4 Click **Settings** in the navigation panel on the left.

Step 5 If **DCE/RPC Configuration** under **Application Layer Preprocessors** is disabled, click **Enabled**.

Step 6 Click **Edit** (✎) next to **DCE/RPC Configuration**.

Step 7 Modify the options in the **Global Settings** section; see DCE/RPC Global Options, on page 233.

Step 8 You have the following choices:

- Add a server profile — Click **Add** (+) next to **Servers**. Specify one or more IP addresses in the **Server Address** field, then click **OK**.
- Delete a server profile — Click **Delete** (🗑) next to the policy.
- Edit a server profile — Click the configured address for the profile under **Servers**, or click **default**. You can modify any of the settings in the **Configuration** section; see DCE/RPC Target-Based Policy Options, on page 235.

Step 9 To save changes you made in this policy since the last policy commit, click **Policy Information**, then click **Commit Changes**.

If you leave the policy without committing changes, cached changes since the last commit are discarded if you edit a different policy.

What to do next

- If you want to generate intrusion events, enable DCE/RPC preprocessor rules (GID 132 or 133). For more information, see Setting Intrusion Rule States, on page 42, DCE/RPC Global Options, on page 233, DCE/RPC Target-Based Policy Options, on page 235, and Traffic-Associated DCE/RPC Rules, on page 239.
- Deploy configuration changes.

The DNS Preprocessor

Note This section applies to Snort 2 preprocessors. For information on Snort 3 inspectors, see https://www.cisco.com/go/snort3-inspectors.

The DNS preprocessor inspects DNS name server responses for the following specific exploits:

- Overflow attempts on RData text fields
- Obsolete DNS resource record types
- Experimental DNS resource record types

The most common type of DNS name server response provides one or more IP addresses that correspond to domain names in the query that prompted the response. Other types of server responses provide, for example, the destination of an email message or the location of a name server that can provide information not available from the server originally queried.

A DNS response is comprised of:

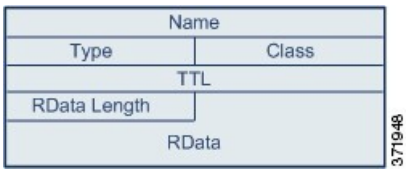
- a message header
- a Question section that contains one or more requests
- three sections that respond to requests in the Question section
  - Answer
  - Authority
  - Additional Information.

Responses in these three sections reflect the information in *resource records* (RR) maintained on the name server. The following table describes these three sections.

Table 92: DNS Name Server RR Responses

This section...	Includes...	For example...
Answer	Optionally, one or more resource records that provide a specific answer to a query	The IP address corresponding to a domain name
Authority	Optionally, one or more resource records that point to an authoritative name server	The name of an authoritative name server for the response
Additional Information	Optionally, one or more resource records that provided additional information related to the Answer sections	The IP address of another server to query

There are many types of resource records, all adhering to the following structure:



Theoretically, any type of resource record can be used in the Answer, Authority, or Additional Information section of a name server response message. The DNS preprocessor inspects any resource record in each of the three response sections for the exploits it detects.

The Type and RData resource record fields are of particular importance to the DNS preprocessor. The Type field identifies the type of resource record. The RData (resource data) field provides the response content. The size and content of the RData field differ depending on the type of resource record.

DNS messages typically use the UDP transport protocol but also use TCP when the message type requires reliable delivery or the message size exceeds UDP capabilities. The DNS preprocessor inspects DNS server responses in both UDP and TCP traffic.

The DNS preprocessor does not inspect TCP sessions picked up in midstream, and ceases inspection if a session loses state because of dropped packets.

DNS Preprocessor Options

Ports

This field specifies the source port or ports the DNS preprocessor should monitor for DNS server responses. Separate multiple ports with commas.

The typical port to configure for the DNS preprocessor is well-known port 53, which DNS name servers use for DNS messages in both UDP and TCP.

Detect Overflow attempts on RData Text fields

When the resource record type is TXT (text), the RData field is a variable-length ASCII text field.

When selected, this option detects a specific vulnerability identified by entry CVE-2006-3441 in MITRE's Current Vulnerabilities and Exposures database. This is a known vulnerability in Microsoft Windows 2000 Service Pack 4, Windows XP Service Pack 1 and Service Pack 2, and Windows Server 2003 Service Pack 1. An attacker can exploit this vulnerability and take complete control of a host by sending or otherwise causing the host to receive a maliciously crafted name server response that causes a miscalculation in the length of an RData text field, resulting in a buffer overflow.

You should enable this option when your network might include hosts running operating systems that have not been upgraded to correct this vulnerability.

You can enable rule 131:3 to generate events and, in an inline deployment, drop offending packets for this option. See Setting Intrusion Rule States, on page 42.

Detect Obsolete DNS RR Types

RFC 1035 identifies several resource record types as obsolete. Because these are obsolete record types, some systems do not account for them and may be open to exploits. You would not expect to encounter these record types in normal DNS responses unless you have purposely configured your network to include them.

You can configure the system to detect known obsolete resource record types. The following table lists and describes these record types.

Table 93: Obsolete DNS Resource Record Types

RR Type	Code	Description
3	MD	a mail destination
4	MF	a mail forwarder

You can enable rule 131:1 to generate events and, in an inline deployment, drop offending packets for this option. See Setting Intrusion Rule States, on page 42.

Detecting Experimental DNS RR Types

RFC 1035 identifies several resource record types as experimental. Because these are experimental record types, some systems do not account for them and may be open to exploits. You would not expect to encounter these record types in normal DNS responses unless you have purposely configured your network to include them.

You can configure the system to detect known experimental resource record types. The following table lists and describes these record types.

Table 94: Experimental DNS Resource Record Types

RR Type	Code	Description
7	MB	a mailbox domain name
8	MG	a mail group member
9	MR	a mail rename domain name
10	NUL	a null resource record

You can enable rule 131:2 to generate events and, in an inline deployment, drop offending packets for this option. See Setting Intrusion Rule States, on page 42.

Configuring the DNS Preprocessor

Note This section applies to Snort 2 preprocessors. For information on Snort 3 inspectors, see https://www.cisco.com/go/snort3-inspectors.

Procedure

Step 1 Choose **Policies** > **Access Control heading** > **Access Control**, and then click **Network Analysis Policy** or **Policies** > **Access Control heading** > **Intrusion**, and then click **Network Analysis Policies**.

Note
If your custom user role limits access to the first path listed here, use the second path to access the policy.

Step 2 Click **Snort 2 Version** next to the policy you want to edit.

Step 3 Click **Edit** (✎) next to the policy you want to edit.

If **View** (👁) appears instead, the configuration belongs to an ancestor domain, or you do not have permission to modify the configuration.

Step 4 Click **Settings** in the navigation panel.

Step 5 If **DNS Configuration** under **Application Layer Preprocessors** is disabled, click **Enabled**.

Step 6 Click **Edit** (✎) next to **DNS Configuration**.

Step 7 Modify the settings as described in DNS Preprocessor Options, on page 242.

Step 8 To save changes you made in this policy since the last policy commit, click **Policy Information**, then click **Commit Changes**.

If you leave the policy without committing changes, cached changes since the last commit are discarded if you edit a different policy.

What to do next

- If you want to generate intrusion events, enable DNS preprocessor rules (GID 131). For more information, see Setting Intrusion Rule States, on page 42 and DNS Preprocessor Options, on page 242.
- Deploy configuration changes.

The FTP/Telnet Decoder

Note This section applies to Snort 2 preprocessors. For information on Snort 3 inspectors, see https://www.cisco.com/go/snort3-inspectors.

The FTP/Telnet decoder analyzes FTP and telnet data streams, normalizing FTP and telnet commands before processing by the rules engine.

Global FTP and Telnet Options

You can set global options to determine whether the FTP/Telnet decoder performs stateful or stateless inspection of packets, whether the decoder detects encrypted FTP or telnet sessions, and whether the decoder continues to check a data stream after it encounters encrypted data.

If no preprocessor rule is mentioned in the following descriptions, the option is not associated with a preprocessor rule.

Stateful Inspection

When selected, causes the FTP/Telnet decoder to save state and provide session context for individual packets and only inspect reassembled sessions. When cleared, analyzes each individual packet without session context.

To check for FTP data transfers, this option must be selected.

Detect Encrypted Traffic

Detects encrypted telnet and FTP sessions.

You can enable rules 125:7 and 126:2 to generate events and, in an inline deployment, drop offending packets for this option. See Setting Intrusion Rule States, on page 42.

Continue to Inspect Encrypted Data

Instructs the preprocessor to continue checking a data stream after it is encrypted, looking for eventual decrypted data that can be processed.

Telnet Options

You can enable or disable normalization of telnet commands by the FTP/Telnet decoder, enable or disable a specific anomaly case, and set the threshold number of Are You There (AYT) attacks to permit.

If no preprocessor rule is mentioned in the following descriptions, the option is not associated with a preprocessor rule.

Ports

Indicates the ports whose telnet traffic you want to normalize. Telnet typically connects to TCP port 23. In the interface, list multiple ports separated by commas.

Caution Because encrypted traffic (SSL) cannot be decoded, adding port 22 (SSH) could yield unexpected results.

Normalize

Normalizes telnet traffic to the specified ports.

Detect Anomalies

Enables detection of Telnet SB (subnegotiation begin) without the corresponding SE (subnegotiation end).

Telnet supports subnegotiation, which begins with SB (subnegotiation begin) and must end with an SE (subnegotiation end). However, certain implementations of Telnet servers will ignore the SB without a corresponding SE. This is anomalous behavior that could be an evasion case. Because FTP uses the Telnet protocol on the control connection, it is also susceptible to this behavior.

You can enable rule 126:3 to generate an event and, in an inline deployment, drop offending packets when this anomaly is detected in Telnet traffic, and rule 125:9 when it is detected on the FTP command channel. See Setting Intrusion Rule States, on page 42.

Are You There Attack Threshold Number

Detects when the number of consecutive AYT commands exceeds the specified threshold. Cisco recommends that you set the AYT threshold to a value no higher than the default value.

You can enable rule 126:1 to generate events and, in an inline deployment, drop offending packets for this option. See Setting Intrusion Rule States, on page 42.

Server-Level FTP Options

You can set options for decoding on multiple FTP servers. Each server profile you create contains the server IP address and the ports on the server where traffic should be monitored. You can specify which FTP commands to validate and which to ignore for a particular server, and set maximum parameter lengths for commands. You can also set the specific command syntax the decoder should validate against for particular commands and set alternate maximum command parameter lengths.

If no preprocessor rule is mentioned in the following descriptions, the option is not associated with a preprocessor rule.

Networks

Use this option to specify one or more IP addresses of FTP servers.

You can specify a single IP address or address block, or a comma-separated list comprised of either or both. You can configure up to 1024 characters, and you can specify up to 255 profiles including the default profile.

Note that the `default` setting in the default policy specifies all IP addresses on your monitored network segment that are not covered by another target-based policy. Therefore, you cannot and do not need to specify an IP address or address block for the default policy, and you cannot leave this setting blank in another policy or use address notation to represent `any` (for example, 0.0.0.0/0 or ::/0).

Ports

Use this option to specify the ports on the FTP server where the managed device should monitor traffic. In the interface, list multiple ports separated by commas. Port 21 is the well-known port for FTP traffic.

File Get Commands

Use this option to define the FTP commands used to transfer files from server to client. Do not change these values unless directed to do so by Support.

Caution Do not modify the **File Get Commands** field unless directed to by Support.

File Put Commands

Use this option to define the FTP commands used to transfer files from client to server. Do not change these values unless directed to do so by Support.

Caution Do not modify the **File Put Commands** field unless directed to by Support.

Additional FTP Commands

Use this line to specify the additional commands that the decoder should detect. Separate additional commands by spaces.

Additional commands you may want to add include `XPWD`, `XCWD`, `XCUP`, `XMKD`, and `XRMD`. For more information on these commands, see RFC 775, the Directory oriented FTP commands specification by the Network Working Group.

Default Max Parameter Length

Use this option to detect the maximum parameter length for commands where an alternate maximum parameter length has not been set. You can add as many alternative maximum parameter lengths as needed.

You can enable rule 125:3 to generate events and, in an inline deployment, drop offending packets for this option. See Setting Intrusion Rule States, on page 42.

Alternate Max Parameter Length

Use this option to specify commands where you want to detect a different maximum parameter length, and to specify the maximum parameter length for those commands. Click **Add** to add lines where you can specify a different maximum parameter length to detect for particular commands.

Check Commands for String Format Attacks

Use this option to check the specified commands for string format attacks.

You can enable rule 125:5 to generate events and, in an inline deployment, drop offending packets for this option. See Setting Intrusion Rule States, on page 42.

Command Validity

Use this option to enter a valid format for a specific command. Click **Add** to add a command validation line.

You can enable rules 125:2 and 125:4 to generate events and, in an inline deployment, drop offending packets for this option. See Setting Intrusion Rule States, on page 42.

Ignore FTP Transfers

Use this option to improve performance on FTP data transfers by disabling all inspection other than state inspection on the data transfer channel.

Note To inspect data transfers, the global FTP/Telnet **Stateful Inspection** option must be selected.

Detect Telnet Escape Codes within FTP Commands

Use this option to detect when telnet commands are used over the FTP command channel.

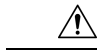
You can enable rule 125:1 to generate events and, in an inline deployment, drop offending packets for this option. See Setting Intrusion Rule States, on page 42.

Ignore Erase Commands during Normalization

When **Detect Telnet Escape Codes within FTP Commands** is selected, use this option to ignore telnet character and line erase commands when normalizing FTP traffic. The setting should match how the FTP server handles telnet erase commands. Note that newer FTP servers typically ignore telnet erase commands, while older servers typically process them.

Troubleshooting Option: Log FTP Command Validation Configuration

Support might ask you during a troubleshooting call to configure your system to print the configuration information for each FTP command listed for the server.



Caution Do not enable **Log FTP Command Validation Configuration** unless instructed to do so by Support.

FTP Command Validation Statements

When setting up a validation statement for an FTP command, you can specify a group of alternative parameters by separating the parameters with spaces. You can also create a binary OR relationship between two parameters by separating them with a pipe character (`|`) in the validation statement. Surrounding parameters by square brackets (`[]`) indicates that those parameters are optional. Surrounding parameters with curly brackets (`{}`) indicates that those parameters are required.

You can create FTP command parameter validation statements to validate the syntax of a parameter received as part of an FTP communication.

Any of the parameters listed in the following table can be used in FTP command parameter validation statements.

Table 95: FTP Command Parameters

If you use...	The following validation occurs...
`int`	The represented parameter must be an integer.
`number`	The represented parameter must be an integer between 1 and 255.
`char` _chars	The represented parameter must be a single character and a member of the characters specified in the _chars argument. For example, defining the command validity for `MODE` with the validation statement `char` SBC checks that the parameter for the `MODE` command comprises the character `S` (representing Stream mode), the character `B` (representing Block mode), or the character `C` (representing Compressed mode).

If you use...	The following validation occurs...
`date` _datefmt	If _datefmt contains `#`, the represented parameter must be a number. If _datefmt contains `c`, the represented parameter must be a character. If _datefmt contains literal strings, the represented parameter must match the literal string.
`string`	The represented parameter must be a string.
`host_port`	The represented parameter must be a valid host port specifier as defined by RFC 959, the File Transfer Protocol specification by the Network Working Group.

You can combine the syntax in the table above as needed to create parameter validation statements that correctly validate each FTP command where you need to validate traffic.

Note When you include a complex expression in a TYPE command, surround it by spaces. Also, surround each operand within the expression by spaces. For example, type `char A | B` , not `char A|B`.

Client-Level FTP Options

Use these options to configure custom FTP client profiles. If an option description does not include a preprocessor rule, the option is not associated with a preprocessor rule.

Networks

Use this option to specify one or more IP addresses of FTP clients.

You can specify a single IP address or address block, or a comma-separated list comprised of either or both. You can specify up to 1024 characters, and you can specify up to 255 profiles including the default profile.

Note that the `default` setting in the default policy specifies all IP addresses on your monitored network segment that are not covered by another target-based policy. Therefore, you cannot and do not need to specify an IP address or address block for the default policy, and you cannot leave this setting blank in another policy or use address notation to represent `any` (for example, 0.0.0.0/0 or ::/0).

Max Response Length

Use this option to specify the maximum allowed response length to an FTP command accepted by the client. This can detect basic buffer overflows.

You can enable rule 125:6 to generate events and, in an inline deployment, drop offending packets for this option. See Setting Intrusion Rule States, on page 42.

Detect FTP Bounce Attempts

Use this option to detect FTP bounce attacks.

You can enable rule 125:8 to generate events and, in an inline deployment, drop offending packets for this option. See Setting Intrusion Rule States, on page 42.

Allow FTP Bounce to

Use this option to configure a list of additional hosts and ports on those hosts on which FTP PORT commands should not be treated as FTP bounce attacks.

Detect Telnet Escape Codes within FTP Commands

Use this option to detect when telnet commands are used over the FTP command channel.

You can enable rule 125:1 to generate events and, in an inline deployment, drop offending packets for this option. See Setting Intrusion Rule States, on page 42.

Ignore Erase Commands During Normalization

When **Detect Telnet Escape Codes within FTP Commands** is selected, use this option to ignore telnet character and line erase commands when normalizing FTP traffic.The setting should match how the FTP client handles telnet erase commands. Note that newer FTP clients typically ignore telnet erase commands, while older clients typically process them.

Configuring the FTP/Telnet Decoder

Note This section applies to Snort 2 preprocessors. For information on Snort 3 inspectors, see https://www.cisco.com/go/snort3-inspectors.

You can configure client profiles for FTP clients to monitor FTP traffic from clients.

Before you begin

- Confirm that any networks you want to identify in a custom target-based policy match or are a subset of the networks, zones, and VLANs handled by its parent network analysis policy. See Advanced Settings for Network Analysis Policies, on page 81 for more information.

Procedure

Step 1 Choose **Policies** > **Access Control heading** > **Access Control**, and then click **Network Analysis Policy** or **Policies** > **Access Control heading** > **Intrusion**, and then **Network Analysis Policies**.

Note
If your custom user role limits access to the first path listed here, use the second path to access the policy.

Step 2 Click **Snort 2 Version** next to the policy you want to edit.

Step 3 Click **Edit** (✎) next to the policy you want to edit.

If **View** (👁) appears instead, the configuration belongs to an ancestor domain, or you do not have permission to modify the configuration.

Step 4 Click **Settings** in the navigation panel.

Step 5 If **FTP and Telnet Configuration** under **Application Layer Preprocessors** is disabled, click **Enabled**.

Step 6 Click **Edit** (✎) next to **FTP and Telnet Configuration**.

Step 7 Set options in the **Global Settings** section as described in Global FTP and Telnet Options, on page 245.

Step 8 Set options in the **Telnet Settings** section as described in Telnet Options, on page 245.

Step 9 Manage FTP server profiles:

- Add a server profile — Click **Add** (+) next to **FTP Server**. Specify one or more IP addresses for the client in the **Server Address** field and click **OK**. You can specify a single IP address or address block, or a comma-separated list of either or both. You can specify up to 1024 characters, and you can configure up to 255 policies, including the default policy.
- Edit a server profile — Click the configured address for a custom profile under **FTP Server**, or click **default**. You can modify the settings in the **Configuration** section; see Server-Level FTP Options, on page 246.
- Delete a server profile — Click **Delete** (🗑) next to the profile.

Step 10 Manage FTP client profiles:

- Add a client profile — Click **Add** (+) next to **FTP Client**. Specify one or more IP addresses for the client in the **Client Address** field and click **OK**. You can specify a single IP address or address block, or a comma-separated list of either or both. You can specify up to 1024 characters, and you can configure up to 255 policies, including the default policy.
- Edit a client profile — Click the configured address for a profile you have added under **FTP Client**, or click **default**. You can modify the settings in the Configuration page area; see Client-Level FTP Options, on page 249.
- Delete a client profile — Click **Delete** (🗑) next to a custom profile.

Step 11 To save changes you made in this policy since the last policy commit, click **Policy Information**, then click **Commit Changes**.

If you leave the policy without committing changes, cached changes since the last commit are discarded if you edit a different policy.

What to do next

- If you want to generate intrusion events, enable FTP and telnet preprocessor rules (GID 125 and 126). For more information, see Setting Intrusion Rule States, on page 42.
- Deploy configuration changes.

The HTTP Inspect Preprocessor

Note This section applies to Snort 2 preprocessors. For information on Snort 3 inspectors, see https://www.cisco.com/go/snort3-inspectors.

The HTTP Inspect preprocessor is responsible for:

- decoding and normalizing HTTP requests sent to and HTTP responses received from web servers on your network

- separating messages sent to web servers into URI, non-cookie header, cookie header, method, and message body components to improve performance of HTTP-related intrusion rules
- separating messages received from web servers into status code, status message, non-set-cookie header, cookie header, and response body components to improve performance of HTTP-related intrusion rules
- detecting possible URI-encoding attacks
- making the normalized data available for additional rule processing
- detecting and preventing attacks through malicious scripts such as JavaScript.

HTTP traffic can be encoded in a variety of formats, making it difficult for rules to appropriately inspect. HTTP Inspect decodes 14 types of encoding, ensuring that your HTTP traffic gets the best inspection possible.

You can configure HTTP Inspect options globally, on a single server, or for a list of servers.

Note that the preprocessor engine performs HTTP normalization *statelessly*. That is, it normalizes HTTP strings on a packet-by-packet basis, and can only process HTTP strings that have been reassembled by the TCP stream preprocessor.

fast_blocking

Among the global configuration options for the HTTP Inspect preprocessor, the fast_blocking option was introduced starting Snort version 2.9.16.0. This option enables inspecting HTTP data before the data is cleared. This enables early IPS rule evaluation so that the block rules are applied and the connection is blocked at the earliest instead of blocking it after clearing the data. This configuration is effective only when inline normalization is enabled.

To enable the fast_blocking option, you must use a network analysis policy with Maximum Detection as the base policy.

Global HTTP Normalization Options

The global HTTP options provided for the HTTP Inspect preprocessor control how the preprocessor functions. Use these options to enable or disable HTTP normalization when ports not specified as web server ports receive HTTP traffic.

Note the following:

- If you enable **Unlimited Decompression**, the **Maximum Compressed Data Depth** and **Maximum Decompressed Data Depth** options are automatically set to 65535 when you commit your changes.
- The highest value is used when the values for **Maximum Compressed Data Depth** or **Maximum Decompressed Data Depth** are different in:
  - the default network analysis policy
  - any other custom network analysis policy invoked by network analysis rules in the same access control policy

If no preprocessor rule is mentioned in the following descriptions, the option is not associated with a preprocessor rule.

Detect Anomalous HTTP Servers

Detects HTTP traffic sent to or received by ports not specified as web server ports.

Note If you turn this option on, be sure to list all ports that do receive HTTP traffic in a server profile on the HTTP Configuration page. If you do not, and you enable this option and the accompanying preprocessor rule, normal traffic to and from the server will generate events. The default server profile contains all ports normally used for HTTP traffic, but if you modified that profile, you may need to add those ports to another profile to prevent events from being generated.

You can enable rule 120:1 to generate events and, in an inline deployment, drop offending packets for this option. See Setting Intrusion Rule States, on page 42.

Detect HTTP Proxy Servers

Detects HTTP traffic using proxy servers not defined by the **Allow HTTP Proxy Use** option.

You can enable rule 119:17 to generate events and, in an inline deployment, drop offending packets for this option. See Setting Intrusion Rule States, on page 42.

Maximum Compressed Data Depth

Sets the maximum size of compressed data to decompress when **Inspect Compressed Data** (and, optionally, **Decompress SWF File (LZMA)**, **Decompress SWF File (Deflate)**, or **Decompress PDF File (Deflate)**) is enabled.

Maximum Decompressed Data Depth

Sets the maximum size of the normalized decompressed data when **Inspect Compressed Data** (and, optionally, **Decompress SWF File (LZMA)**, **Decompress SWF File (Deflate)**, or **Decompress PDF File (Deflate)**) is enabled.

Server-Level HTTP Normalization Options

You can set server-level options for each server you monitor, globally for all servers, or for a list of servers. Additionally, you can use a predefined server profile to set these options, or you can set them individually to meet the needs of your environment. Use these options, or one of the default profiles that set these options, to specify the HTTP server ports whose traffic you want to normalize, the amount of server response payload you want to normalize, and the types of encoding you want to normalize.

If no preprocessor rule is mentioned in the following descriptions, the option is not associated with a preprocessor rule.

Networks

Use this option to specify the IP address of one or more servers. You can specify a single IP address or address block, or a comma-separated list comprised of either or both.

In addition to a limit of up to 255 total profiles, including the default profile, you can include up to 496 characters, or approximately 26 entries, in an HTTP server list, and specify a total of 256 address entries for all server profiles.

Note that the `default` setting in the default policy specifies all IP addresses on your monitored network segment that are not covered by another target-based policy. Therefore, you cannot and do not need to specify an IP address or CIDR block/prefix length for the default policy, and you cannot leave this setting blank in another policy or use address notation to represent `any` (for example, 0.0.0.0/0 or ::/0).

Ports

The ports whose HTTP traffic the preprocessor engine normalizes. Separate multiple port numbers with commas.

Oversize Dir Length

Detects URL directories longer than the specified value.

You can enable rule 119:15 to generate events and, in an inline deployment, drop offending packets when the preprocessor detects a request for a URL that is longer than the specified length.

Client Flow Depth

Specifies the number of bytes for rules to inspect in raw HTTP packets, including header and payload data, in client-side HTTP traffic defined in **Ports**. Client flow depth does not apply when HTTP content rule options within a rule inspect specific parts of a request message.

Specify any of the following:

- A positive value inspects the specified number of bytes in the first packet. If the first packet contains fewer bytes than specified, inspect the entire packet. Note that the specified value applies to both segmented and reassembled packets.

  Note also that a value of 300 typically eliminates inspection of large HTTP Cookies that appear at the end of many client request headers.

- 0 inspects all client-side traffic, including multiple packets in a session and exceeding the upper byte limit if necessary. Note that this value is likely to affect performance.

- -1 ignores all client-side traffic.

Server Flow Depth

Specifies the number of bytes for rules to inspect in raw HTTP packets in server-side HTTP traffic specified by **Ports**. Inspection includes the raw header and payload when **Inspect HTTP Responses** disabled and only the raw response body when **Inspect HTTP Response** is enabled.

Server flow depth specifies the number of bytes of raw server response data in a session for rules to inspect in server-side HTTP traffic defined in **Ports**. You can use this option to balance performance and the level of inspection of HTTP server response data. Server flow depth does not apply when HTTP content options within a rule inspect specific parts of a response message.

Unlike client flow depth, server flow depth specifies the number of bytes per HTTP response, not per HTTP request packet, for rules to inspect.

You can specify any of the following:

- A positive value:

  When **Inspect HTTP Responses** is **enabled**, inspects only the raw HTTP response body, and not raw HTTP headers; also inspects decompressed data when **Inspect Compressed Data** is enabled.

When **Inspect HTTP Responses** is **disabled**, inspects the raw packet header and payload.

If the session includes fewer response bytes than specified, rules fully inspect all response packets in a given session, across multiple packets as needed. If the session includes more response bytes than specified, rules inspect only the specified number of bytes for that session, across multiple packets as needed.

Note that a small flow depth value may cause false negatives from rules that target server-side traffic defined in **Ports**. Most of these rules target either the HTTP header or content that is likely to be in the first hundred or so bytes of non-header data. Headers are usually under 300 bytes long, but header size may vary.

Note also that the specified value applies to both segmented and reassembled packets.

- 0 inspects the entire packet for all HTTP server-side traffic defined in **Ports**, including response data in a session that exceeds 65535 bytes.

  Note that this value is likely to affect performance.

- -1:

  When **Inspect HTTP Responses** is **enabled**, inspects only raw HTTP headers and not the raw HTTP response body.

  When **Inspect HTTP Responses** is **disabled**, ignores all server-side traffic defined in **Ports.**

Maximum Header Length

Detects a header field longer than the specified maximum number of bytes in an HTTP request; also in HTTP responses when **Inspect HTTP Responses** is enabled. A value of 0 disables this option. Specify a positive value to enable it.

You can enable rule 119:19 to generate events and, in an inline deployment, drop offending packets for this option. See Setting Intrusion Rule States, on page 42..

Maximum Number of Headers

Detects when the number of headers exceeds this setting in an HTTP request. A value of 0 disables this option. Specify a positive value to enable it.

You can enable rule 119:20 to generate events and, in an inline deployment, drop offending packets for this option. See Setting Intrusion Rule States, on page 42..

Maximum Number of Spaces

Detects when the number of white spaces in a folded line equals or exceeds this setting in an HTTP request. A value of 0 disables this option. Specify a positive value to enable it.

You can enable rule 119:26 to generate events and, in an inline deployment, drop offending packets for this option. See Setting Intrusion Rule States, on page 42..

HTTP Client Body Extraction Depth

Specifies the number of bytes to extract from the message body of an HTTP client request. You can use an intrusion rule to inspect the extracted data by selecting the `content` or `protected_content` keyword **HTTP Client Body** option.

Specify -1 to ignore the client body. Specify 0 to extract the entire client body. Note that identifying specific bytes to extract can improve system performance. Note also that you must specify a value greater than or equal to 0 for the **HTTP Client Body** option to function in an intrusion rule.

Small Chunk Size

Specifies the maximum number of bytes at which a chunk is considered small. Specify a positive value. A value of 0 disables detection of anomalous consecutive small segments. See the **Consecutive Small Chunks** option for more information.

Consecutive Small Chunks

Specifies how many consecutive small chunks represent an abnormally large number in client or server traffic that uses chunked transfer encoding. The **Small Chunk Size** option specifies the maximum size of a small chunk.

For example, set **Small Chunk Size** to 10 and **Consecutive Small Chunks** to 5 to detect 5 consecutive chunks of 10 bytes or less.

You can enable preprocessor rule 119:27 to generate events and, in an inline deployment, drop offending packets on excessive small chunks in client traffic, and rule 120:7 in server traffic. When **Small Chunk Size** is enabled and this option is set to 0 or 1, enabling these rules would trigger an event on every chunk of the specified size or less.

HTTP Methods

Specifies HTTP request methods in addition to GET and POST that you expect the system to encounter in traffic. Use a comma to separate multiple values.

Intrusion rules use the `content` or `protected_content` keyword with the **HTTP Method** argument to search for content in HTTP methods. You can enable rule 119:31to generate events and, in an inline deployment, drop offending packets when a method other than GET, POST, or a method configured for this option is encountered in traffic. See Setting Intrusion Rule States, on page 42.

No Alerts

Disables intrusion events when accompanying preprocessor rules are enabled.

Note This option does **not** disable HTTP standard text rules and shared object rules.

Normalize HTTP Headers

When **Inspect HTTP Responses** is enabled, enables normalization of non-cookie data in request and response headers. When **Inspect HTTP Responses** is **not** enabled, enables normalization of the entire HTTP header, including cookies, in request and response headers.

Inspect HTTP Cookies

Enables extraction of cookies from HTTP request headers. Also enables extraction of set-cookie data from response headers when **Inspect HTTP Responses** is enabled. Disabling this option when cookie extraction is not required can improve performance.

Note that the `Cookie:` and `Set-Cookie:` header names, leading spaces on the header line, and the `CRLF` that terminates the header line are inspected as part of the header and not as part of the cookie.

Normalize Cookies in HTTP headers

Enables normalization of cookies in HTTP request headers. When **Inspect HTTP Responses** is enabled, also enables normalization of set-cookie data in response headers. You must select **Inspect HTTP Cookies** before selecting this options.

Allow HTTP Proxy Use

Allows the monitored web server to be used as an HTTP proxy. This option is used only in the inspection of HTTP requests.

Inspect URI Only

Inspects only the URI portion of the normalized HTTP request packet.

Inspect HTTP Responses

Enables extended inspection of HTTP responses so, in addition to decoding and normalizing HTTP request messages, the preprocessor extracts response fields for inspection by the rules engine. Enabling this option causes the system to extract the response header, body, status code, and so on, and also extracts set-cookie data when **Inspect HTTP Cookies** is enabled.

You can enable rules 120:2 and 120:3 to generate events and, in an inline deployment, drop offending packets, as follows:

Table 96: Inspect HTTP Response Rules

This rule...	Triggers when...
120:2	an invalid HTTP response status code occurs.
120:3	an HTTP response does not include Content-Length or Transfer-Encoding.

Normalize UTF Encodings to UTF-8

When **Inspect HTTP Responses** is enabled, detects UTF-16LE, UTF-16BE, UTF-32LE, and UTF32-BE encodings in HTTP responses and normalizes them to UTF-8.

You can enable rule 120:4 to generate events and, in an inline deployment, drop offending packets when UTF normalization fails.

Inspect Compressed Data

When **Inspect HTTP Responses** is enabled, enables decompression of gzip and deflate-compatible compressed data in the HTTP response body, and inspection of the normalized decompressed data. The system inspects chunked and non-chunked HTTP response data. The system inspects decompressed data packet by packet across multiple packets as needed; that is, the system does not combine the decompressed data from different packets for inspection. Decompression ends when **Maximum Compressed Data Depth**, **Maximum Decompressed Data Depth**, or the end of the compressed data is reached. Inspection of decompressed data

ends when **Server Flow Depth** is reached unless you also select **Unlimited Decompression**. You can use the `file_data` rule keyword to inspect decompressed data.

You can enable rules 120:6 and 120:24 to generate events and, in an inline deployment, drop offending packets, as follows:

Table 97: Inspect Compressed HTTP Response Rules

This rule...	Triggers when...
120:6	decompression of a compressed HTTP response fails.
120:24	partial decompression of a compressed HTTP response fails.

Unlimited Decompression

When **Inspect Compressed Data** (and, optionally, **Decompress SWF File (LZMA)**, **Decompress SWF File (Deflate)**, or **Decompress PDF File (Deflate)**) is enabled, overrides **Maximum Decompressed Data Depth** across multiple packets; that is, this option enables unlimited decompression across multiple packets. Note that enabling this option does not affect **Maximum Compressed Data Depth** or **Maximum Decompressed Data Depth** within a single packet. Note also that enabling this option sets **Maximum Compressed Data Depth** and **Maximum Decompressed Data Depth** to 65535 when you commit your changes.

Normalize Javascript

When **Inspect HTTP Responses** is enabled, enables detection and normalization of Javascript within the HTTP response body. The preprocessor normalizes obfuscated Javascript data such as the unescape and decodeURI functions and the String.fromCharCode method. The preprocessor normalizes the following encodings within the unescape, decodeURI, and decodeURIComponent functions:

- %XX
- %uXXXX
- 0xXX
- \xXX
- \uXXXX

The preprocessor detects consecutive white spaces and normalizes them into a single space. When this option is enabled, a configuration field allows you to specify the maximum number of consecutive white spaces to permit in obfuscated Javascript data. You can enter a value from 1 to 65535. The value 0 disables event generation, regardless of whether the preprocessor rule (120:10) associated with this field is enabled.

The preprocessor also normalizes the Javascript plus (+) operator and concatenates strings using the operator.

You can use the `file_data` intrusion rule keyword to point intrusion rules to the normalized Javascript data.

You can enable rules 120:9, 120:10, and 120:11 to generate events and, in an inline deployment, drop offending packets, as follows:

Table 98: Normalize Javascript Option Rules

This rule...	Triggers when...
120:9	the obfuscation level within the preprocessor is greater than or equal to 2.
120:10	the number of consecutive white spaces in the Javascript obfuscated data is greater than or equal to the value configured for the maximum number of consecutive white spaces allowed.
120:11	escaped or encoded data includes more than one type of encoding.

Decompress SWF File (LZMA) and Decompress SWF File (Deflate)

When **HTTP Inspect Responses** is enabled, these options decompress the compressed portions of files located within the HTTP response body of HTTP requests.

Note You can **only** decompress the compressed portions of files found in HTTP GET responses.

- **Decompress SWF File (LZMA)** decompresses the LZMA-compatible compressed portions of Adobe ShockWave Flash (`.swf`) files
- **Decompress SWF File (Deflate)** decompresses the deflate-compatible compressed portions of Adobe ShockWave Flash (`.swf`) files

Decompression ends when **Maximum Compressed Data Depth**, **Maximum Decompressed Data Depth**, or the end of the compressed data is reached. Inspection of decompressed data ends when **Server Flow Depth** is reached unless you also select **Unlimited Decompression**. You can use the `file_data` intrusion rule keyword to inspect decompressed data.

You can enable rules 120:12 and 120:13 to generate events and, in an inline deployment, drop offending packets, as follows:

Table 99: Decompress SWF File Option Rules

This rule...	Triggers when...
120:12	deflate file decompression fails.
120:13	LZMA file decompression fails.

Decompress PDF File (Deflate)

When **HTTP Inspect Responses** is enabled, **Decompress PDF File (Deflate)** decompresses the deflate-compatible compressed portions of Portable Document Format (`.pdf`) files located within the HTTP response body of HTTP requests. The system can only decompress PDF files with the `/FlateDecode` stream filter. Other stream filters (including `/FlateDecode /FlateDecode`) are unsupported.

Note You can **only** decompress the compressed portions of files found in HTTP GET responses.

Decompression ends when **Maximum Compressed Data Depth**, **Maximum Decompressed Data Depth**, or the end of the compressed data is reached. Inspection of decompressed data ends when **Server Flow Depth** is reached unless you also select **Unlimited Decompression**. You can use the `file_data` intrusion rule keyword to inspect decompressed data.

You can enable rules 120:14, 120:15, 120:16, and 120:17 to generate events and, in an inline deployment, drop offending packets, as follows:

Table 100: Decompress PDF File (Deflate) Option Rules

This rule...	Triggers when...
120:14	file decompression fails.
120:15	file decompression fails due to an unsupported compression type.
120:16	file decompression fails due to an unsupported PDF stream filter.
120:17	file parsing fails.

Extract Original Client IP Address

Enables the examination of original client IP addresses during intrusion inspection. The system extracts the original client IP address from the X-Forwarded-For (XFF), True-Client-IP, or custom HTTP headers you define in the **XFF Header Priority** option. You can view the extracted original client IP address in the intrusion events table.

You can enable rules 119:23, 119:29, and 119:30 to generate events and, in an inline deployment, drop offending packets for this option. See Setting Intrusion Rule States, on page 42..

XFF Header Priority

Specifies the order in which the system processes original client IP headers when multiple headers are present in an HTTP request. By default, the system examines X-Forwarded-For (XFF) headers, then True-Client-IP headers. Use the up and down arrow icons beside each header type to adjust its priority.

This option also allows you to specify original client IP headers other than XFF or True-Client-IP for extraction and evaluation. Click **Add** to add custom header names to the priority list. The system only supports custom headers that use the same syntax as an XFF or True-Client-IP header.

Keep in mind the following when configuring this option:

- The system uses this priority order when evaluating original client IP address headers for both access control and intrusion inspection.
- If multiple original client IP headers are present, the system processes only the header with the highest priority.
- The XFF header contains a list of IP addresses, which represent the proxy servers through which the request has passed. To prevent spoofing, the system uses the last IP address in the list (that is, the address appended by the trusted proxy) as the original client IP address.

Log URI

Enables extraction of the raw URI, if present, from HTTP request packets and associates the URI with all intrusion events generated for the session.

When this option is enabled, you can display the first fifty characters of the extracted URI in the HTTP URI column of the intrusion events table view. You can display the complete URI, up to 2048 bytes, in the packet view.

Log Hostname

Enables extraction of the host name, if present, from the HTTP request Host header and associates the host name with all intrusion events generated for the session. When multiple Host headers are present, extracts the host name from the first header.

When this option is enabled, you can display the first fifty characters of the extracted host name in the HTTP Hostname column of the intrusion events table view. You can display the complete host name, up to 256 bytes, in the packet view.

You can enable rule 119:25 to generate events and, in an inline deployment, drop offending packets for this option. See Setting Intrusion Rule States, on page 42.

Note that, when enabled, rule 119:24 triggers if it detects multiple Host headers in an HTTP request, regardless of the setting for this option.

Profile

Specifies the types of encoding that are normalized for HTTP traffic. The system provides a default profile appropriate for most servers, default profiles for Apache servers and IIS servers, and custom default settings that you can tailor to meet the needs of your monitored traffic:

- Select **All** to use the standard default profile, appropriate for all servers.
- Select **IIS** to use the system-provided IIS profile.
- Select **Apache** to use the system-provided Apache profile.
- Select **Custom** to create your own server profile.

Server-Level HTTP Normalization Encoding Options

When you set the HTTP server-level **Profile** option to `Custom`, you can specify the types of encoding that are normalized for HTTP traffic, and enable HTTP preprocessor rules to generate events against traffic containing the different encoding types.

If no preprocessor rule is mentioned in the following descriptions, the option is not associated with a preprocessor rule.

ASCII Encoding

Decodes encoded ASCII characters and specifies whether the rules engine generates an event on ASCII-encoded URIs.

You can enable rule 119:1 to generate events and, in an inline deployment, drop offending packets for this option. See Setting Intrusion Rule States, on page 42.

UTF-8 Encoding

Decodes standard UTF-8 Unicode sequences in the URI.

You can enable rule 119:6 to generate events and, in an inline deployment, drop offending packets for this option. See Setting Intrusion Rule States, on page 42.

Microsoft %U Encoding

Decodes the IIS %u encoding scheme that uses %u followed by four characters where the 4 characters are a hex encoded value that correlates to an IIS Unicode codepoint.

Tip Legitimate clients rarely use %u encodings, so Cisco recommends decoding HTTP traffic encoded with %u encodings.

You can enable rule 119:3 to generate events and, in an inline deployment, drop offending packets for this option. See Setting Intrusion Rule States, on page 42.

Bare Byte UTF-8 Encoding

Decodes bare byte encoding, which uses non-ASCII characters as valid values in decoding UTF-8 values.

Tip Bare byte encoding allows the user to emulate an IIS server and interpret non-standard encodings correctly. Cisco recommends enabling this option because no legitimate clients encode UTF-8 this way.

You can enable rule 119:4 to generate events and, in an inline deployment, drop offending packets for this option. See Setting Intrusion Rule States, on page 42.

Microsoft IIS Encoding

Decodes using Unicode codepoint mapping.

Tip Cisco recommends enabling this option, because it is seen mainly in attacks and evasion attempts.

You can enable rule 119:7 to generate events and, in an inline deployment, drop offending packets for this option. See Setting Intrusion Rule States, on page 42.

Double Encoding

Decodes IIS double encoded traffic by making two passes through the request URI performing decodes in each one. Cisco recommends enabling this option because it is usually found only in attack scenarios.

You can enable rule 119:2 to generate events and, in an inline deployment, drop offending packets for this option. See Setting Intrusion Rule States, on page 42.

Multi-Slash Obfuscation

Normalizes multiple slashes in a row into a single slash.

You can enable rule 119:8 to generate events and, in an inline deployment, drop offending packets for this option. See Setting Intrusion Rule States, on page 42.

IIS Backslash Obfuscation

Normalizes backslashes to forward slashes.

You can enable rule 119:9 to generate events and, in an inline deployment, drop offending packets for this option. See Setting Intrusion Rule States, on page 42.

Directory Traversal

Normalizes directory traversals and self-referential directories. If you enable the accompanying preprocessor rules to generate events against this type of traffic, it may generate false positives because some web sites refer to files using directory traversals.

You can enable rules 119:10 and 119:11 to generate events and, in an inline deployment, drop offending packets for this option. See Setting Intrusion Rule States, on page 42.

Tab Obfuscation

Normalizes the non-RFC standard of using a tab for a space delimiter. Apache and other non-IIS web servers use the tab character (0x09) as a delimiter in URLs.

Note Regardless of the configuration for this option, the HTTP Inspect preprocessor treats a tab as white space if a space character (0x20) precedes it.

You can enable rule 119:12 to generate events and, in an inline deployment, drop offending packets for this option. See Setting Intrusion Rule States, on page 42.

Invalid RFC Delimiter

Normalizes line breaks (\n) in URI data.

You can enable rule 119:13 to generate events and, in an inline deployment, drop offending packets for this option. See Setting Intrusion Rule States, on page 42.

Webroot Directory Traversal

Detects directory traversals that traverse past the initial directory in the URL.

You can enable rule 119:18 to generate events and, in an inline deployment, drop offending packets for this option. See Setting Intrusion Rule States, on page 42.

Tab URI Delimiter

Turns on the use of the tab character (0x09) as a delimiter for a URI. Apache, newer versions of IIS, and some other web servers use the tab character as a delimiter in URLs.

Note Regardless of the configuration for this option, the HTTP Inspect preprocessor treats a tab as white space if a space character (0x20) precedes it.

Non-RFC characters

Detects the non-RFC character list you add in the corresponding field when it appears within incoming or outgoing URI data. When modifying this field, use the hexadecimal format that represents the byte character. If and when you configure this option, set the value with care. Using a character that is very common may overwhelm you with events.

You can enable rule 119:14 to generate events and, in an inline deployment, drop offending packets for this option. See Setting Intrusion Rule States, on page 42.

Max Chunk Encoding Size

Detects abnormally large chunk sizes in URI data.

You can enable rules 119:16 and 119:22 to generate events and, in an inline deployment, drop offending packets for this option. See Setting Intrusion Rule States, on page 42.

Disable Pipeline Decoding

Disables HTTP decoding for pipelined requests. When this option is disabled, performance is enhanced because HTTP requests waiting in the pipeline are not decoded or analyzed, and are only inspected using generic pattern matching.

Non-Strict URI Parsing

Enables non-strict URI parsing. Use this option only on servers that will accept non-standard URIs in the format "GET /index.html abc xo qr \n". Using this option, the decoder assumes that the URI is between the first and second space, even if there is no valid HTTP identifier after the second space.

Extended ASCII Encoding

Enables parsing of extended ASCII characters in an HTTP request URI. Note that this option is available in custom server profiles only, and not in the default profiles provided for Apache, IIS, or all servers.

Configuring the HTTP Inspect Preprocessor

Note This section applies to Snort 2 preprocessors. For information on Snort 3 inspectors, see https://www.cisco.com/go/snort3-inspectors.

Before you begin

- Confirm that any networks you want to identify in a custom target-based policy match or are a subset of the networks, zones, and VLANs handled by its parent network analysis policy. See Advanced Settings for Network Analysis Policies, on page 81 for more information.

Procedure

Step 1 Choose **Policies** > **Access Control heading** > **Access Control**, and then click **Network Analysis Policy** or **Policies** > **Access Control heading** > **Intrusion**, and then click **Network Analysis Policies**.

Note
If your custom user role limits access to the first path listed here, use the second path to access the policy.

Step 2 Click **Snort 2 Version** next to the policy you want to edit.

Step 3 Click **Edit** ( ) next to the policy you want to edit.

If **View** ( ) appears instead, the configuration belongs to an ancestor domain, or you do not have permission to modify the configuration.

Step 4 Click **Settings** in the navigation panel.

Step 5 If **HTTP Configuration** under **Application Layer Preprocessors** is disabled, click **Enabled**.

Step 6 Click **Edit** ( ) next to **HTTP Configuration**.

Step 7 Modify the options in the Global Settings page area; see Global HTTP Normalization Options, on page 252.

Step 8 You have three choices:

- Add a server profile — Click **Add** ( ) in the **Servers** section. Specify one or more IP addresses for the client in the **Server Address** field, and click **OK**. You can specify a single IP address or address block, or a comma-separated list of either or both. You can include up to 496 characters in a list, specify a total of 256 address entries for all server profiles, and create a total of 255 profiles including the default profile.
- Edit a server profile — Click the configured address for a profile you have added under **Servers**, or click **default**. You can modify any of the settings in the **Configuration** section; see Server-Level HTTP Normalization Options, on page 253. If you choose **Custom** for the **Profile** value, you can also modify the encoding options described in Server-Level HTTP Normalization Encoding Options, on page 261.
- Delete a server profile — Click **Delete** ( ) next to a custom profile.

Step 9 To save changes you made in this policy since the last policy commit, click **Policy Information**, then click **Commit Changes**.

If you leave the policy without committing changes, cached changes since the last commit are discarded if you edit a different policy.

What to do next

- If you want generate events and, in an inline deployment, drop offending packets, enable HTTP preprocessor rules (GID 119). For more information, see Setting Intrusion Rule States, on page 42.
- Deploy configuration changes.

Additional HTTP Inspect Preprocessor Rules

You can enable the rules in the **Preprocessor Rule GID:SID** column of the following table to generate events for HTTP Inspect preprocessor rules that are not associated with specific configuration options.

Table 101: Additional HTTP Inspect Preprocessor Rules

Preprocessor Rule GID:SID	Triggers when...
119:21	an HTTP request header has more than one `content-length` field.
119:24	an HTTP request has more than one Host header.
119:28	an HTTP POST method has neither a `content-length` header nor chunked `transfer-encoding`.
119:32	HTTP version 0.9 is encountered in traffic. Note that the TCP stream configuration must also be enabled.
119:33	an HTTP URI includes an unescaped space.
119:34	a TCP connection contains 24 or more pipelined HTTP requests.
120:5	UTF-7 encoding is encountered in HTTP response traffic; UTF-7 should only appear where 7-bit parity is required, such as in SMTP traffic.
120:8	the `content-length` or chunk size is invalid.
120:18	an HTTP server response occurs before the client request.
120:19	an HTTP response includes multiple content lengths.
120:20	an HTTP response includes multiple content encodings.
120:25	an HTTP response includes invalid header folding.
120:26	a junk line occurs before an HTTP response header.
120:27	an HTTP response does not include an end of header.
120:28	an invalid chunk size occurs, or chunk size is followed by junk characters.

The Sun RPC Preprocessor

Note This section applies to Snort 2 preprocessors. For information on Snort 3 inspectors, see https://www.cisco.com/go/snort3-inspectors.

Remote Procedure Call (RPC) normalization takes fragmented RPC records and normalizes them to a single record so the rules engine can inspect the complete record. For example, an attacker may attempt to discover

the port where RPC `admind` runs. Some UNIX hosts use RPC `admind` to perform remote distributed system tasks. If the host performs weak authentication, a malicious user could take control of remote administration. The standard text rule (GID: 1) with the Snort ID (SID) 575 detects this attack by searching for content in specific locations to identify inappropriate `portmap GETPORT` requests.

Sun RPC Preprocessor Options

Ports

Specify the ports whose traffic you want to normalize. In the interface, list multiple ports separated by commas. Typical RPC ports are 111 and 32771. If your network sends RPC traffic to other ports, consider adding them.

Detect fragmented RPC records

Detects RPC fragmented records.

You can enable rules 106:1 and 106:5 to generate events and, in an inline deployment, drop offending packets for this option. See Setting Intrusion Rule States, on page 42.

Detect multiple records in one packet

Detects more than one RPC request per packet (or reassembled packet).

You can enable rule 106:2 to generate events and, in an inline deployment, drop offending packets for this option. See Setting Intrusion Rule States, on page 42.

Detect fragmented record sums which exceed one fragment

Detects reassembled fragment record lengths that exceed the current packet length.

You can enable rule 106:3 to generate events and, in an inline deployment, drop offending packets for this option. See Setting Intrusion Rule States, on page 42.

Detect single fragment records which exceed the size of one packet

Detects partial records

You can enable rule 106:4 to generate events and, in an inline deployment, drop offending packets for this option. See Setting Intrusion Rule States, on page 42.

Configuring the Sun RPC Preprocessor

Note This section applies to Snort 2 preprocessors. For information on Snort 3 inspectors, see https://www.cisco.com/go/snort3-inspectors.

Procedure

Step 1 Choose **Policies** > **Access Control heading** > **Access Control**, and then click **Network Analysis Policy** or **Policies** > **Access Control heading** > **Intrusion**, and then click **Network Analysis Policies**.

Note
If your custom user role limits access to the first path listed here, use the second path to access the policy.

Step 2 Click **Snort 2 Version** next to the policy you want to edit.

Step 3 Click **Edit** (✎) next to the policy you want to edit.

If **View** (👁) appears instead, the configuration belongs to an ancestor domain, or you do not have permission to modify the configuration.

Step 4 Click **Settings** in the navigation panel.

Step 5 If **Sun RPC Configuration** under **Application Layer Preprocessors** is disabled, click **Enabled**.

Step 6 Click **Edit** (✎) next to **Sun RPC Configuration**.

Step 7 Modify the settings described in Sun RPC Preprocessor Options, on page 267.

Step 8 To save changes you made in this policy since the last policy commit, click **Policy Information**, then click **Commit Changes**.

If you leave the policy without committing changes, cached changes since the last commit are discarded if you edit a different policy.

What to do next

- If you want to generate events and, in an inline deployment, drop offending packets, enable Sun RPC preprocessor rules (GID 106). For more information, see Setting Intrusion Rule States, on page 42.
- Deploy configuration changes.

The SIP Preprocessor

Note This section applies to Snort 2 preprocessors. For information on Snort 3 inspectors, see https://www.cisco.com/go/snort3-inspectors.

The Session Initiation Protocol (SIP) provides call setup, modification, and teardown of one or more sessions for one or more users of client applications such as Internet telephony, multimedia conferencing, instant messaging, online gaming, and file transfer. A *method* field in each SIP request identifies the purpose of the request, and a Request-URI specifies where to send the request. A status code in each SIP response indicates the outcome of the requested action.

After calls are set up using SIP, the Real-time Transport Protocol (RTP) is responsible for subsequent audio and video communication; this part of the session is sometimes referred to as the call channel, the data channel,

or the audio/video data channel. RTP uses the Session Description Protocol (SDP) within the SIP message body for data-channel parameter negotiation, session announcement, and session invitation.

The SIP preprocessor is responsible for:

- decoding and analyzing SIP 2.0 traffic
- extracting the SIP header and message body, including SDP data when present, and passing the extracted data to the rules engine for further inspection
- generating events when the following conditions are detected and the corresponding preprocessor rules are enabled:
  - anomalies and known vulnerabilities in SIP packets
  - out-of-order and invalid call sequences
- optionally, ignoring the call channel

The preprocessor identifies the RTP channel based on the port identified in the SDP message, which is embedded in the SIP message body, but the preprocessor does not provide RTP protocol inspection.

Note the following when using the SIP preprocessor:

- UDP typically carries media sessions supported by SIP. UDP stream preprocessing provides SIP session tracking for the SIP preprocessor.
- SIP rule keywords allow you to point to the SIP packet header or message body and to limit detection to packets for specific SIP methods or status codes.

SIP Preprocessor Options

For the following options, you can specify a positive value from 1 to 65535 bytes, or 0 to disable event generation for the option regardless of whether the associated rule is enabled.

- **Maximum Request URI Length**
- **Maximum Call ID Length**
- **Maximum Request Name Length**
- **Maximum From Length**
- **Maximum To Length**
- **Maximum Via Length**
- **Maximum Contact Length**
- **Maximum Content Length**

If no preprocessor rule is mentioned in the following descriptions, the option is not associated with a preprocessor rule.

Ports

Specifies the ports to inspect for SIP traffic. You can specify an integer from 0 to 65535. Separate multiple port numbers with commas.

Methods to Check

Specifies SIP methods to detect. You can specify any of the following currently defined SIP methods:

```
ack, benotify, bye, cancel, do, info, invite, join, message,
notify, options, prack, publish, quath, refer, register,
service, sprack, subscribe, unsubscribe, update
```

Methods are case-insensitive. The method name can include alphabetic characters, numbers, and the underscore character. No other special characters are permitted. Separate multiple methods with commas.

Because new SIP methods might be defined in the future, your configuration can include an alphabetic string that is not currently defined. The system supports up to 32 methods, including the 21 currently defined methods and an additional 11 methods. The system ignores any undefined methods that you might configure.

Note that, in addition to any methods you specify for this option, the 32 total methods includes methods specified using the `sip_method` keyword in intrusion rules.

Maximum Dialogs within a Session

Specifies the maximum number of dialogs allowed within a stream session. If more dialogs than this number are created, the oldest dialogs are dropped until the number of dialogs does not exceed the maximum number specified. You can specify an integer from 1 to 4194303.

You can enable rule 140:27 to generate events and, in an inline deployment, drop offending packets for this option. See Setting Intrusion Rule States, on page 42.

Maximum Request URI Length

Specifies the maximum number of bytes to allow in the Request-URI header field. A Longer URI generates an event and, in an inline deployment, drops offending packets when rule 140:3 is enabled. The request URI field indicates the destination path or page for the request.

Maximum Call ID Length

Specifies the maximum number of bytes to allow in the request or response Call-ID header field. A longer Call-ID generates an event and, in an inline deployment, drops offending packets when rule 140:5 is enabled. The Call-ID field uniquely identifies the SIP session in requests and responses.

Maximum Request Name Length

Specifies the maximum number of bytes to allow in the request name, which is the name of the method specified in the CSeq transaction identifier. A longer request name generates an event and, in an inline deployment, drops offending packets when rule 140:7 is enabled.

Maximum From Length

Specifies the maximum number of bytes to allow in the request or response From header field. A longer From generates an event and, in an inline deployment, drops offending packets when rule 140:9 is enabled. The From field identifies the message initiator.

Maximum To Length

Specifies the maximum number of bytes to allow in the request or response To header field. A longer To generates an event and, in an inline deployment, drops offending packets when rule 140:11 is enabled. The To field identifies the message recipient.

Maximum Via Length

Specifies the maximum number of bytes to allow in the request or response Via header field. A longer Via generates an event and, in an inline deployment, drops offending packets when rule 140:13 is enabled. The Via field provides the path followed by the request and, in a response, receipt information.

Maximum Contact Length

Specifies the maximum number of bytes to allow in the request or response Contact header field. A longer Contactgenerates an event and, in an inline deployment, drops offending packets when rule 140:15 is enabled. The Contact field provides a URI that specifies the location to contact with subsequent messages.

Maximum Content Length

Specifies the maximum number of bytes to allow in the content of the request or response message body. Longer content generates an event and, in an inline deployment, drops offending packets when rule 140:16 is enabled.

Ignore Audio/Video Data Channel

Enables and disables inspection of data channel traffic. Note that the preprocessor continues inspection of other non-data-channel SIP traffic when you enable this option.

Configuring the SIP Preprocessor

Note This section applies to Snort 2 preprocessors. For information on Snort 3 inspectors, see https://www.cisco.com/go/snort3-inspectors.

Procedure

Step 1 Choose **Policies** > **Access Control heading** > **Access Control**, and then click **Network Analysis Policy** or **Policies** > **Access Control heading** > **Intrusion**, and then click **Network Analysis Policies**.

Note
If your custom user role limits access to the first path listed here, use the second path to access the policy.

Step 2 Click **Snort 2 Version** next to the policy you want to edit.

Step 3 Click **Edit** ( ) next to the policy you want to edit.

If **View** ( ) appears instead, the configuration belongs to an ancestor domain, or you do not have permission to modify the configuration.

Step 4 Click **Settings** in the navigation panel.

Step 5 If **SIP Configuration** under **Application Layer Preprocessors** is disabled, click **Enabled**.

Step 6 Click **Edit** (✎) next to **SIP Configuration**.

Step 7 Modify the options described in SIP Preprocessor Options, on page 269.

Step 8 To save changes you made in this policy since the last policy commit, click **Policy Information**, then click **Commit Changes**.

If you leave the policy without committing changes, cached changes since the last commit are discarded if you edit a different policy.

What to do next

- If you want to generate events and, in an inline deployment, drop offending packets, enable SIP preprocessor rules (GID 140). For more information, see Setting Intrusion Rule States, on page 42.
- Deploy configuration changes.

Additional SIP Preprocessor Rules

The SIP preprocessor rules in the following table are not associated with specific configuration options. As with other SIP preprocessor rules, you must enable these rules if you want them to generate events and, in an inline deployment, drop offending packets.

Table 102: Additional SIP Preprocessor Rules

Preprocessor Rule GID:SID	Triggers when...
140:1	the preprocessor is monitoring the maximum number of SIP sessions allowed by the system.
140:2	the required Request_URI field is empty in a SIP request.
140:4	the Call-ID header field is empty in a SIP request or response.
140:6	the value for the sequence number in the SIP request or response CSeq field is not a 32-bit unsigned integer less than 231.
140:8	the From header field is empty in a SIP request or response.
140:10	the To header field is empty in a SIP request or response.
140:12	the Via header field is empty in a SIP request or response
140:14	the required Contact header field is empty in a SIP request or response.
140:17	a single SIP request or response packet in UDP traffic contains multiple messages. Note that older SIP versions supported multiple messages, but SIP 2.0 supports only one message per packet.

Preprocessor Rule GID:SID	Triggers when...
140:18	the actual length of the message body in a SIP request or response in UDP traffic does not match the value specified in the Content-Length header field in a SIP request or response.
140:19	the preprocessor does not recognize a method name in the CSeq field of a SIP response.
140:20	the SIP server does not challenge an authenticated invite message. Note that this occurs in the case of the InviteReplay billing attack.
140:21	session information changes before the call is set up. Note that this occurs in the case of the FakeBusy billing attack.
140:22	the response status code is not a three-digit number.
140:23	the Content-Type header field does not specify a content type and the message body contains data.
140:24	the SIP version is not 1, 1.1, or 2.0.
140:25	the method specified in the CSeq header and the method field do not match in a SIP request.
140:26	the preprocessor does not recognize the method named in the SIP request method field.

The GTP Preprocessor

Note This section applies to Snort 2 preprocessors. For information on Snort 3 inspectors, see https://www.cisco.com/go/snort3-inspectors.

The General Service Packet Radio (GPRS) Tunneling Protocol (GTP) provides communication over a GTP core network. The GTP preprocessor detects anomalies in GTP traffic and forwards command channel signaling messages to the rules engine for inspection. You can use the `gtp_version`, `gtp_type`, and `gtp_info` rule keywords to inspect GTP command channel traffic for exploits.

A single configuration option allows you to modify the default setting for the ports that the preprocessor inspects for GTP command channel messages.

GTP Preprocessor Rules

You must enable the GTP preprocessor rules in the following table if you want them to generate events and, in an inline deployment, drop offending packets.

Table 103: GTP Preprocessor Rules

Preprocessor Rule GID:SID	Description
143:1	Generates an event when the preprocessor detects an invalid message length.
143:2	Generates an event when the preprocessor detects an invalid information element length.
143:3	Generates an event when the preprocessor detects information elements that are out of order.

Configuring the GTP Preprocessor

Note This section applies to Snort 2 preprocessors. For information on Snort 3 inspectors, see https://www.cisco.com/go/snort3-inspectors.

You can use this procedure to modify the ports the GTP preprocessor monitors for GTP command messages.

Procedure

Step 1 Choose **Policies** > **Access Control heading** > **Access Control**, and then click **Network Analysis Policy** or **Policies** > **Access Control heading** > **Intrusion**, and then click **Network Analysis Policies**.

Note
If your custom user role limits access to the first path listed here, use the second path to access the policy.

Step 2 Click **Snort 2 Version** next to the policy you want to edit.

Step 3 Click **Edit** (✎) next to the policy you want to edit.

If **View** (👁) appears instead, the configuration belongs to an ancestor domain, or you do not have permission to modify the configuration.

Step 4 Click **Settings** in the navigation panel on the left.

Step 5 If **GTP Command Channel Configuration** under **Application Layer Preprocessors** is disabled, click **Enabled**.

Step 6 Click **Edit** (✎) next to **GTP Command Channel Configuration**.

Step 7 Enter a **Ports** value.

Separate multiple ports with commas.

Step 8 To save changes you made in this policy since the last policy commit, click **Policy Information**, then click **Commit Changes**.

If you leave the policy without committing changes, cached changes since the last commit are discarded if you edit a different policy.

What to do next

- If you want to enable intrusion events, enable GTP preprocessor rules (GID 143). For more information, see Setting Intrusion Rule States, on page 42.
- Deploy configuration changes.

The IMAP Preprocessor

Note This section applies to Snort 2 preprocessors. For information on Snort 3 inspectors, see https://www.cisco.com/go/snort3-inspectors.

The Internet Message Application Protocol (IMAP) is used to retrieve email from a remote IMAP server. The IMAP preprocessor inspects server-to-client IMAP4 traffic and, when associated preprocessor rules are enabled, generates events on anomalous traffic. The preprocessor can also extract and decode email attachments in client-to-server IMAP4 traffic and send the attachment data to the rules engine. You can use the `file_data` keyword in an intrusion rule to point to the attachment data.

Extraction and decoding include multiple attachments, when present, and large attachments that span multiple packets.

IMAP Preprocessor Options

Note that decoding, or extraction when the MIME email attachment does not require decoding, includes multiple attachments when present, and large attachments that span multiple packets.

Note also that the highest value is used when the values for the **Base64 Decoding Depth**, **7-Bit/8-Bit/Binary Decoding Depth**, **Quoted-Printable Decoding Depth**, or **Unix-to-Unix Decoding Depth** options are different in:

- the default network analysis policy
- any other custom network analysis policy invoked by network analysis rules in the same access control policy

If no preprocessor rule is mentioned in the following descriptions, the option is not associated with a preprocessor rule.

Ports

Specifies the ports to inspect for IMAP traffic. You can specify an integer from 0 to 65535. Separate multiple port numbers with commas.

Base64 Decoding Depth

Specifies the maximum number of bytes to extract and decode from each Base64 encoded MIME email attachment. You can specify a positive value, or specify 0 to decode all the Base64 data. Specify -1 to ignore Base64 data.

Note that positive values not divisible by 4 are rounded up to the next multiple of 4 except for the values 65533, 65534, and 65535, which are rounded down to 65532.

When this option is enabled, you can enable rule 141:4 generate events and, in an inline deployment, drop offending packets when decoding fails; decoding could fail, for example, because of incorrect encoding or corrupted data.

7-Bit/8-Bit/Binary Decoding Depth

Specifies the maximum bytes of data to extract from each MIME email attachment that does not require decoding. These attachment types include 7-bit, 8-bit, binary, and various multipart content types such as plain text, jpeg images, mp3 files, and so on. You can specify a positive value, or specify 0 to extract all data in the packet. Specify -1 to ignore non-decoded data.

When this option is enabled, you can enable rule 141:6 to generate events and, in an inline deployment, drop offending packets when extraction fails; extraction could fail, for example, because of corrupted data.

Quoted-Printable Decoding Depth

Specifies the maximum number of bytes to extract and decode from each quoted-printable (QP) encoded MIME email attachment. You can specify a positive value, or specify 0 to decode all QP encoded data in the packet. Specify -1 to ignore QP encoded data.

When this option is enabled, you can enable rule 141:5 to generate events and, in an inline deployment, drop offending packets when decoding fails; decoding could fail, for example, because of incorrect encoding or corrupted data.

Unix-to-Unix Decoding Depth

Specifies the maximum number of bytes to extract and decode from each Unix-to-Unix encoded (uuencoded) email attachment. You can specify a positive value, or specify 0 to decode all uuencoded data in the packet. Specify -1 to ignore uuencoded data.

When this option is enabled, you can enable rule 141:7 to generate events and, in an inline deployment, drop offending packets when decoding fails; decoding could fail, for example, because of incorrect encoding or corrupted data.

Configuring the IMAP Preprocessor

Note This section applies to Snort 2 preprocessors. For information on Snort 3 inspectors, see https://www.cisco.com/go/snort3-inspectors.

Caution For Classic devices only, changing the value for **Base64 Decoding Depth**, **7-Bit/8-Bit/Binary Decoding Depth**, **Quoted-Printable Decoding Depth**, or **Unix-to-Unix Decoding Depth** restarts the Snort process when you deploy configuration changes, temporarily interrupting traffic inspection. Whether traffic drops during this interruption or passes without further inspection depends on how the assigned device handles traffic.

Procedure

Step 1 Choose **Policies** > **Access Control heading** > **Access Control**, and then click **Network Analysis Policy** or **Policies** > **Access Control heading** > **Intrusion**, and then **Network Analysis Policies**.

Note
If your custom user role limits access to the first path listed here, use the second path to access the policy.

Step 2 Click **Snort 2 Version** next to the policy you want to edit.

Step 3 Click **Edit** ( ) next to the policy you want to edit.

If **View** ( ) appears instead, the configuration belongs to an ancestor domain, or you do not have permission to modify the configuration.

Step 4 Click **Settings** in the navigation panel.

Step 5 If **IMAP Configuration** under **Application Layer Preprocessors** is disabled, click **Enabled**.

Step 6 Click **Edit** ( ) next to **IMAP Configuration**.

Step 7 Modify the settings described in IMAP Preprocessor Options, on page 275.

Step 8 To save changes you made in this policy since the last policy commit, click **Policy Information**, then click **Commit Changes**.

If you leave the policy without committing changes, cached changes since the last commit are discarded if you edit a different policy.

What to do next

- If you want to enable intrusion events, enable IMAP preprocessor rules (GID 141); see Setting Intrusion Rule States, on page 42.
- Deploy configuration changes.

Additional IMAP Preprocessor Rules

The IMAP preprocessor rules in the following table are not associated with specific configuration options. As with other IMAP preprocessor rules, you must enable these rules if you want them to generate events and, in an inline deployment, drop offending packets.

Table 104: Additional IMAP Preprocessor Rules

Preprocessor Rule GID:SID	Description
141:1	Generates an event when the preprocessor detects a client command that is not defined in RFC 3501.
141:2	Generates an event when the preprocessor detects a server response that is not defined in RFC 3501.
141:3	Generates an event when the preprocessor is using the maximum amount of memory allowed by the system. At this point, the preprocessor stops decoding until memory becomes available.

The POP Preprocessor

Note This section applies to Snort 2 preprocessors. For information on Snort 3 inspectors, see https://www.cisco.com/go/snort3-inspectors.

The Post Office Protocol (POP) is used to retrieve email from a remote POP mail server. The POP preprocessor inspects server-to-client POP3 traffic and, when associated preprocessor rules are enabled, generates events on anomalous traffic. The preprocessor can also extract and decode email attachments in client-to-server POP3 traffic and send the attachment data to the rules engine. You can use the `file_data` keyword in an intrusion rule to point to attachment data.

Extraction and decoding include multiple attachments, when present, and large attachments that span multiple packets.

POP Preprocessor Options

Note that decoding, or extraction when the MIME email attachment does not require decoding, includes multiple attachments when present, and large attachments that span multiple packets.

Note also that the highest value is used when the values for the **Base64 Decoding Depth**, **7-Bit/8-Bit/Binary Decoding Depth**, **Quoted-Printable Decoding Depth**, or **Unix-to-Unix Decoding Depth** options are different in:

- the default network analysis policy
- any other custom network analysis policy invoked by network analysis rules in the same access control policy

If no preprocessor rule is mentioned in the following descriptions, the option is not associated with a preprocessor rule.

Ports

Specifies the ports to inspect for POP traffic. You can specify an integer from 0 to 65535. Separate multiple port numbers with commas.

Base64 Decoding Depth

Specifies the maximum number of bytes to extract and decode from each Base64 encoded MIME email attachment. You can specify a positive value, or specify 0 to decode all the Base64 data. Specify -1 to ignore Base64 data.

Note that positive values not divisible by 4 are rounded up to the next multiple of 4 except for the values 65533, 65534, and 65535, which are rounded down to 65532.

When this option is enabled, you can enable rule 142:4 to generate an event and, in an inline deployment, drop offending packets when decoding fails; decoding could fail, for example, because of incorrect encoding or corrupted data.

7-Bit/8-Bit/Binary Decoding Depth

Specifies the maximum bytes of data to extract from each MIME email attachment that does not require decoding. These attachment types include 7-bit, 8-bit, binary, and various multipart content types such as plain text, jpeg images, mp3 files, and so on. You can specify a positive value, or specify 0 to extract all data in the packet. Specify -1 to ignore non-decoded data.

When this option is enabled, you can enable rule 142:6 to generate an event and, in an inline deployment, drop offending packets when extraction fails; extraction could fail, for example, because of corrupted data.

Quoted-Printable Decoding Depth

Specifies the maximum number of bytes to extract and decode from each quoted-printable (QP) encoded MIME email attachment. You can specify a positive value, or specify 0 to decode all QP encoded data in the packet. Specify -1 to ignore QP encoded data.

When this option is enabled, you can enable rule 142:5 to generate an event and, in an inline deployment, drop offending packets when decoding fails; decoding could fail, for example, because of incorrect encoding or corrupted data.

Unix-to-Unix Decoding Depth

Specifies the maximum number of bytes to extract and decode from each Unix-to-Unix encoded (uuencoded) email attachment. You can specify a positive value, or specify 0 to decode all uuencoded data in the packet. Specify -1 to ignore uuencoded data.

When this option is enabled, you can enable rule 142:7 to generate an event and, in an inline deployment, drop offending packets when decoding fails; decoding could fail, for example, because of incorrect encoding or corrupted data.

Configuring the POP Preprocessor

Note This section applies to Snort 2 preprocessors. For information on Snort 3 inspectors, see https://www.cisco.com/go/snort3-inspectors.

Caution For Classic devices only, changing the value for **Base64 Decoding Depth**, **7-Bit/8-Bit/Binary Decoding Depth**, **Quoted-Printable Decoding Depth**, or **Unix-to-Unix Decoding Depth** restarts the Snort process when you deploy configuration changes, temporarily interrupting traffic inspection. Whether traffic drops during this interruption or passes without further inspection depends on how the assigned device handles traffic.

Procedure

Step 1 Choose **Policies** > **Access Control heading** > **Access Control**, and then click **Network Analysis Policy** or **Policies** > **Access Control heading** > **Intrusion**, and then click **Network Analysis Policies**.

Note
If your custom user role limits access to the first path listed here, use the second path to access the policy.

Step 2 Click **Snort 2 Version** next to the policy you want to edit.

Step 3 Click **Edit** (✎) next to the policy you want to edit.

If **View** (👁) appears instead, the configuration belongs to an ancestor domain, or you do not have permission to modify the configuration.

Step 4 Click **Settings** in the navigation panel.

Step 5 If **POP Configuration** under **Application Layer Preprocessors** is disabled, click **Enabled**.

Step 6 Click **Edit** (✎) next to **POP Configuration**.

Step 7 Modify the settings described in POP Preprocessor Options, on page 278.

Step 8 To save changes you made in this policy since the last policy commit, click **Policy Information**, then click **Commit Changes**.

If you leave the policy without committing changes, cached changes since the last commit are discarded if you edit a different policy.

What to do next

- If you want to enable intrusion events, enable POP preprocessor rules (GID 142). For more information, see Setting Intrusion Rule States, on page 42.
- Deploy configuration changes.

Additional POP Preprocessor Rules

The POP preprocessor rules in the following table are not associated with specific configuration options. As with other POP preprocessor rules, you must enable these rules if you want them to generate events and, in an inline deployment, drop offending packets.

Table 105: Additional POP Preprocessor Rules

Preprocessor Rule GID:SID	Description
142:1	Generates an event when the preprocessor detects a client command that is not defined in RFC 1939.
142:2	Generates an event when the preprocessor detects a server response that is not defined in RFC 1939.
142:3	Generates an event when the preprocessor is using the maximum amount of memory allowed by the system. At this point, the preprocessor stops decoding until memory becomes available.

The SMTP Preprocessor

Note This section applies to Snort 2 preprocessors. For information on Snort 3 inspectors, see https://www.cisco.com/go/snort3-inspectors.

The SMTP preprocessor instructs the rules engine to normalize SMTP commands. The preprocessor can also extract and decode email attachments in client-to-server traffic and, depending on the software version, extract email file names, addresses, and header data to provide context when displaying intrusion events triggered by SMTP traffic.

SMTP Preprocessor Options

You can enable or disable normalization, and you can configure options to control the types of anomalous traffic the SMTP decoder detects.

Note that decoding, or extraction when the MIME email attachment does not require decoding, includes multiple attachments when present, and large attachments that span multiple packets.

Note also that the highest value is used when the values for the **Base64 Decoding Depth**, **7-Bit/8-Bit/Binary Decoding Depth**, **Quoted-Printable Decoding Depth**, or **Unix-to-Unix Decoding Depth** options are different in:

- the default network analysis policy
- any other custom network analysis policy invoked by network analysis rules in the same access control policy

If no preprocessor rule is mentioned in the following descriptions, the option is not associated with a preprocessor rule.

Ports

Specifies the ports whose SMTP traffic you want to normalize. You can specify a value greater than or equal to 0. Separate multiple ports with commas.

Stateful Inspection

When selected, causes SMTP decoder to save state and provide session context for individual packets and only inspects reassembled sessions. When cleared, analyzes each individual packet without session context.

Normalize

When set to `All`, normalizes all commands. Checks for more than one space character after a command.

When set to `None`, normalizes no commands.

When set to `Cmds`, normalizes the commands listed in **Custom Commands**.

Custom Commands

When **Normalize** is set to `Cmds`, normalizes the listed commands.

Specify commands which should be normalized in the text box. Checks for more than one space character after a command.

The space (ASCII 0x20) and tab (ASCII 0x09) characters count as space characters for normalization purposes.

Ignore Data

Does not process mail data; processes only MIME mail header data.

Ignore TLS Data

Does not process data encrypted under the Transport Layer Security protocol.

No Alerts

Disables intrusion events when accompanying preprocessor rules are enabled.

Detect Unknown Commands

Detects unknown commands in SMTP traffic.

You can enable rule 124:5 to generate events and, in an inline deployment, drop offending packets for this option.

Max Command Line Len

Detects when an SMTP command line is longer than this value. Specify `0` to never detect command line length.

RFC 2821, the Network Working Group specification on the Simple Mail Transfer Protocol, recommends 512 as a maximum command line length.

You can enable rule 124:1 to generate events and, in an inline deployment, drop offending packets for this option.

Max Header Line Len

Detects when an SMTP data header line is longer than this value. Specify `0` to never detect data header line length.

You can enable rules 124:2 and 124:7 to generate events and, in an inline deployment, drop offending packets for this option.

Max Response Line Len

Detects when an SMTP response line is longer than this value. Specify `0` to never detect response line length.

RFC 2821 recommends 512 as a maximum response line length.

You can enable rule 124:3 to generate events and, in an inline deployment, drop offending packets for this option and also for **Alt Mac Command Line Len**, when that option is enabled.

Alt Max Command Line Len

Detects when the SMTP command line for any of the specified commands is longer than this value. Specify `0` to never detect command line length for the specified commands. Different default line lengths are set for numerous commands.

This setting overrides the Max Command Line Len setting for the specified commands.

You can enable rule 124:3 to generate events and, in an inline deployment, drop offending packets for this option and also for **Max Response Line Len** when that option is enabled.

Invalid Commands

Detects if these commands are sent from the client side.

You can enable rule 124:6 to generate events and, in an inline deployment, drop offending packets for this option and also for **Invalid Commands**.

Valid Commands

Permits commands in this list.

Even if this list is empty, the preprocessor permits the following valid commands: ATRN AUTH BDAT DATA DEBUG EHLO EMAL ESAM ESND ESOM ETRN EVFY EXPN HELO HELP IDENT MAIL NOOP ONEX QUEU QUIT RCPT RSET SAML SEND SIZE SOML STARTTLS TICK TIME TURN TURNME VERB VRFY XADR XAUTH XCIR XEXCH50 X-EXPS XGEN XLICENSE X-LINK2STATE XQUE XSTA XTRN XUSR

Note RCPT TO and MAIL FROM are SMTP commands. The preprocessor configuration uses command names of RCPT and MAIL, respectively. Within the code, the preprocessor maps RCPT and MAIL to the correct command name.

You can enable rule 124:4 to generate events and, in an inline deployment, drop offending packets for this option and also for **Invalid Commands** when that option is configured.

Data Commands

Lists commands that initiate sending data in the same way the SMTP DATA command sends data per RFC 5321. Separate multiple commands with spaces.

Binary Data Commands

Lists commands that initiate sending data in a way that is similar to how the BDAT command sends data per RFC 3030. Separate multiple commands with spaces.

Authentication Commands

Lists commands that initiate an authentication exchange between client and server. Separate multiple commands with spaces.

Detect xlink2state

Detects packets that are part of X-Link2State Microsoft Exchange buffer data overflow attacks. In inline deployments, the system can also drop those packets.

You can enable rule 124:8 to generate events and, in an inline deployment, drop offending packets for this option.

Base64 Decoding Depth

When **Ignore Data** is disabled, specifies the maximum number of bytes to extract and decode from each Base64 encoded MIME email attachment. You can specify from a positive value, or specify 0 to decode all the Base64 data. Specify -1 to ignore Base64 data. The preprocessor will not decode data when **Ignore Data** is selected.

Note that positive values not divisible by 4 are rounded up to the next multiple of 4 except for the values 65533, 65534, and 65535, which are rounded down to 65532.

When this option is enabled, you can enable rule 124:10 to generate events and, in an inline deployment, drop offending packets when decoding fails; decoding could fail, for example, because of incorrect encoding or corrupted data.

Note that this option replaces the deprecated options **Enable MIME Decoding** and **Maximum MIME Decoding Depth**, which are still supported in existing intrusion policies for backward compatibility.

7-Bit/8-Bit/Binary Decoding Depth

When **Ignore Data** is disabled, specifies the maximum bytes of data to extract from each MIME email attachment that does not require decoding. These attachment types include 7-bit, 8-bit, binary, and various multipart content types such as plain text, jpeg images, mp3 files, and so on. You can specify a positive value, or specify 0 to extract all data in the packet. Specify -1 to ignore non-decoded data. The preprocessor will not extract data when **Ignore Data** is selected.

Quoted-Printable Decoding Depth

When **Ignore Data** is disabled, specifies the maximum number of bytes to extract and decode from each quoted-printable (QP) encoded MIME email attachment.

You can specify from 1 to 65535 bytes, or specify 0 to decode all QP encoded data in the packet. Specify -1 to ignore QP encoded data. The preprocessor will not decode data when **Ignore Data** is selected.

When this option is enabled, you can enable rule 124:11 to generate events and, in an inline deployment, drop offending packets when decoding fails; decoding could fail, for example, because of incorrect encoding or corrupted data.

Unix-to-Unix Decoding Depth

When **Ignore Data** is disabled, specifies the maximum number of bytes to extract and decode from each Unix-to-Unix encoded (uuencoded) email attachment. You can specify from 1 to 65535 bytes, or specify 0 to decode all uuencoded data in the packet. Specify -1 to ignore uuencoded data. The preprocessor will not decode data when **Ignore Data** is selected.

When this option is enabled, you can enable rule 124:13 to generate events and, in an inline deployment, drop offending packets when decoding fails; decoding could fail, for example, because of incorrect encoding or corrupted data.

Log MIME Attachment Names

Enables extraction of MIME attachment file names from the MIME Content-Disposition header and associates the file names with all intrusion events generated for the session. Multiple file names are supported.

When this option is enabled, you can view file names associated with events in the Email Attachment column of the intrusion events table view.

Log To Addresses

Enables extraction of recipient email addresses from the SMTP RCPT TO command and associates the recipient addresses with all intrusion events generated for the session. Multiple recipients are supported.

When this option is enabled, you can view recipients associated with events in the Email Recipient column of the intrusion events table view.

Log From Addresses

Enables extraction of sender email addresses from the SMTP MAIL FROM command and associates the sender addresses with all intrusion events generated for the session. Multiple sender addresses are supported.

When this option is enabled, you can view senders associated with events in the Email Sender column of the intrusion events table view.

Log Headers

Enables extraction of email headers. The number of bytes to extract is determined by the value specified for **Header Log Depth**.

You can use the `content` or `protected_content` keyword to write intrusion rules that use email header data as a pattern. You can also view the extracted email header in the intrusion event packet view.

Header Log Depth

Specifies the number of bytes of the email header to extract when **Log Headers** is enabled. You can specify 0 to 20480 bytes. A value of 0 disables **Log Headers**.

Configuring SMTP Decoding

Note This section applies to Snort 2 preprocessors. For information on Snort 3 inspectors, see https://www.cisco.com/go/snort3-inspectors.

Caution For Classic devices only, changing the value for **Base64 Decoding Depth**, **7-Bit/8-Bit/Binary Decoding Depth**, **Quoted-Printable Decoding Depth**, or **Unix-to-Unix Decoding Depth** restarts the Snort process when you deploy configuration changes, temporarily interrupting traffic inspection. Whether traffic drops during this interruption or passes without further inspection depends on how the assigned device handles traffic.

Procedure

Step 1 Choose **Policies** > **Access Control heading** > **Access Control**, and then click **Network Analysis Policy** or **Policies** > **Access Control heading** > **Intrusion**, and then click **Network Analysis Policies**.

Note
If your custom user role limits access to the first path listed here, use the second path to access the policy.

Step 2 Click **Snort 2 Version** next to the policy you want to edit.

Step 3 Click **Edit** () next to the policy you want to edit.

If **View** () appears instead, the configuration belongs to an ancestor domain, or you do not have permission to modify the configuration.

Step 4 Click **Settings** in the navigation pane.

Step 5 If **SMTP Configuration** under **Application Layer Preprocessors** is disabled, click **Enabled**.

Step 6 Click **Edit** () next to **SMTP Configuration**.

Step 7 Modify the options described in SMTP Preprocessor Options, on page 281.

Step 8 To save changes you made in this policy since the last policy commit, click **Policy Information**, then click **Commit Changes**.

If you leave the policy without committing changes, cached changes since the last commit are discarded if you edit a different policy.

What to do next

- If you want to generate events and, in an inline deployment, drop offending packets, enable SMTP preprocessor rules (GID 124). For more information, see Setting Intrusion Rule States, on page 42.
- Deploy configuration changes.

The SSH Preprocessor

Note This section applies to Snort 2 preprocessors. For information on Snort 3 inspectors, see https://www.cisco.com/go/snort3-inspectors.

The SSH preprocessor detects:

- The Challenge-Response Buffer Overflow exploit
- The CRC-32 exploit
- The SecureCRT SSH Client Buffer Overflow exploit
- Protocol mismatches
- Incorrect SSH message direction
- Any version string other than version 1 or 2

Challenge-Response Buffer Overflow and CRC-32 attacks occur after the key exchange and are, therefore, encrypted. Both attacks send an uncharacteristically large payload of more than 20 KBytes to the server immediately after the authentication challenge. CRC-32 attacks apply only to SSH Version 1; Challenge-Response Buffer Overflow exploits apply only to SSH Version 2. The version string is read at the beginning of the session. Except for the difference in the version string, both attacks are handled in the same way.

The SecureCRT SSH exploit and protocol mismatch attacks occur when attempting to secure a connection, before the key exchange. The SecureCRT exploit sends an overly long protocol identifier string to the client that causes a buffer overflow. A protocol mismatch occurs when either a non-SSH client application attempts to connect to a secure SSH server or the server and client version numbers do not match.

You can configure the SSH preprocessor to inspect traffic on a specified port or list of ports, or to automatically detect SSH traffic. It will continue to inspect SSH traffic until either a specified number of encrypted packets has passed within a specified number of bytes, or until a specified maximum number of bytes is exceeded within the specified number of packets. If the maximum number of bytes is exceeded, it is assumed that a CRC-32 (SSH Version 1) or a Challenge-Response Buffer Overflow (SSH Version 2) attack has occurred. Note that without configuration the preprocessor detects any version string value other than version 1 or 2.

Also note the that the SSH preprocessor does not handle brute force attacks.

SSH Preprocessor Options

The preprocessor stops inspecting traffic for a session when either of the following occurs:

- a valid exchange between the server and the client has occurred for this number of encrypted packets; the connection continues.
- the **Number of Bytes Sent Without Server Response** is reached before the number of encrypted packets to inspect is reached; the assumption is made that there is an attack.

Each valid server response during **Number of Encrypted Packets to Inspect** resets the **Number of Bytes Sent Without Server Response** and the packet count continues.

Consider the following example SSH preprocessor configuration:

- **Server Ports**: 22
- **Autodetect Ports**: off
- **Maximum Length of Protocol Version String**: 80
- **Number of Encrypted Packets to Inspect**: 25

- **Number of Bytes Sent Without Server Response**: 19,600
- All detect options are enabled.

In the example, the preprocessor inspects traffic only on port 22. That is, auto-detection is disabled, so it inspects only on the specified port.

Additionally, the preprocessor in the example stops inspecting traffic when either of the following occurs:

- The client sends 25 encrypted packets which contain no more than 19,600 bytes, cumulative. The assumption is there is no attack.
- The client sends more than 19,600 bytes within 25 encrypted packets. In this case, the preprocessor considers the attack to be the Challenge-Response Buffer Overflow exploit because the session in the example is an SSH Version 2 session.

The preprocessor in the example will also detect any of the following that occur while it is processing traffic:

- a server overflow, triggered by a version string greater than 80 bytes and indicating a SecureCRT exploit
- a protocol mismatch
- a packet flowing in the wrong direction

Finally, the preprocessor will automatically detect any version string other than version 1 or version 2.

If no preprocessor rule is mentioned in the following descriptions, the option is not associated with a preprocessor rule.

Server Ports

Specifies on which ports the SSH preprocessor should inspect traffic.

You can configure a single port or a comma-separated list of ports.

Autodetect Ports

Sets the preprocessor to automatically detect SSH traffic.

When this option is selected, the preprocessor inspects all traffic for an SSH version number. It stops processing when neither the client nor the server packet contains a version number. When disabled, the preprocessor inspects only the traffic identified by the **Server Ports** option.

Number of Encrypted Packets to Inspect

Specifies the number of stream reassembled encrypted packets to examine per session.

Setting this option to zero will allow all traffic to pass.

Reducing the number of encrypted packets to inspect may result in some attacks escaping detection. Raising the number of encrypted packets to inspect may negatively affect performance.

Number of Bytes Sent Without Server Response

Specifies the maximum number of bytes an SSH client may send to a server without getting a response before assuming there is a Challenge-Response Buffer Overflow or CRC-32 attack.

Increase the value for this option if the preprocessor generates false positives on the Challenge-Response Buffer Overflow or CRC-32 exploit.

Maximum Length of Protocol Version String

Specifies the maximum number of bytes allowed in the server's version string before considering it to be a SecureCRT exploit.

Detect Challenge-Response Buffer Overflow Attack

Enables or disables detecting the Challenge-Response Buffer Overflow exploit.

You can enable rule 128:1 to generate events and, in an inline deployment, drop offending packets for this option. Note that an SFTP session can occasionally trigger rule 128:1.

Detect SSH1 CRC-32 Attack

Enables or disables detecting the CRC-32 exploit.

You can enable rule 128:2 to generate events and, in an inline deployment, drop offending packets for this option.

Detect Server Overflow

Enables or disables detecting the SecureCRT SSH Client Buffer Overflow exploit.

You can enable rule 128:3 to generate events and, in an inline deployment, drop offending packets for this option.

Detect Protocol Mismatch

Enables or disables detecting protocol mismatches.

You can enable rule 128:4 to generate events and, in an inline deployment, drop offending packets for this option.

Detect Bad Message Direction

Enables or disables detecting when traffic flows in the wrong direction (that is, if the presumed server generates client traffic, or if a client generates server traffic).

You can enable rule 128:5 to generate events and, in an inline deployment, drop offending packets for this option.

Detect Payload Size Incorrect for the Given Payload

Enables or disables detecting packets with an incorrect payload size such as when the length specified in the SSH packet is not consistent with the total length specified in the IP header or the message is truncated, that is, there is not enough data for a full SSH header.

You can enable rule 128:6 to generate events and, in an inline deployment, drop offending packets for this option.

Detect Bad Version String

Note that, when enabled, the preprocessor detects without configuration any version string other than version 1 or 2.

You can enable rule 128:7 to generate events and, in an inline deployment, drop offending packets for this option.

Configuring the SSH Preprocessor

Note This section applies to Snort 2 preprocessors. For information on Snort 3 inspectors, see https://www.cisco.com/go/snort3-inspectors.

Procedure

Step 1 Choose **Policies** > **Access Control heading** > **Access Control**, and then click **Network Analysis Policy** or **Policies** > **Access Control heading** > **Intrusion**, and then click **Network Analysis Policies**.

Note
If your custom user role limits access to the first path listed here, use the second path to access the policy.

Step 2 Click **Snort 2 Version** next to the policy you want to edit.

Step 3 Click **Edit** () next to the policy you want to edit.

If **View** () appears instead, the configuration belongs to an ancestor domain, or you do not have permission to modify the configuration.

Step 4 Click **Settings** in the navigation panel.

Step 5 If **SSH Configuration** under **Application Layer Preprocessors** is disabled, click **Enabled**.

Step 6 Click **Edit** () next to **SSH Configuration**.

Step 7 Modify the options described in SSH Preprocessor Options, on page 287.

Step 8 To save changes you made in this policy since the last policy commit, click **Policy Information**, then click **Commit Changes**.

If you leave the policy without committing changes, cached changes since the last commit are discarded if you edit a different policy.

What to do next

- If you want to enable intrusion events, enable SSH preprocessor rules (GID 128). For more information, see Setting Intrusion Rule States, on page 42.
- Deploy configuration changes.

The SSL Preprocessor

Note This section applies to Snort 2 preprocessors. For information on Snort 3 inspectors, see https://www.cisco.com/go/snort3-inspectors.

The SSL preprocessor allows you to configure SSL inspection, which can block encrypted traffic, decrypt it, or inspect the traffic with access control. Whether or not you configure SSL inspection, the SSL preprocessor also analyzes SSL handshake messages when detected in traffic and determines when a session becomes encrypted. Identifying encrypted traffic allows the system to stop intrusion and file inspection of encrypted payloads, which helps reduce false positives and improve performance.

The SSL preprocessor can also examine encrypted traffic to detect attempts to exploit the Heartbleed bug, and generate events when it detects such exploits.

You can suspend inspecting traffic for intrusions and malware once the session is encrypted. If you configure SSL inspection, the SSL preprocessor also identifies encrypted traffic you can block, decrypt, or inspect with access control.

Using the SSL preprocessor to decrypt encrypted traffic does not require a license. All other SSL preprocessor functionality, including halting inspection of encrypted payloads for malware and intrusions, and detecting Heartbleed bug exploits, requires a Protection license.

How SSL Preprocessing Works

The SSL preprocessor stops intrusion and file inspection of encrypted data, and inspects encrypted traffic with an SSL policy if you configure SSL inspection. This can help to eliminate false positives. The SSL preprocessor maintains state information as it inspects the SSL handshake, tracking both the state and SSL version for that session. When the preprocessor detects that a session state is encrypted, the system marks the traffic in that session as encrypted. You can configure the system to stop processing on all packets in an encrypted session when encryption is established, as well as generate an event when it detects an attempt to exploit the Heartbleed bug.

For each packet, the SSL preprocessor verifies that the traffic contains an IP header, a TCP header, and a TCP payload, and that it occurs on the ports specified for SSL preprocessing. For qualifying traffic, the following scenarios determine whether the traffic is encrypted:

- The system observes all packets in a session, **Server side data is trusted** is not enabled, and the session includes a Finished message from both the server and the client and at least one packet from each side with an Application record and without an Alert record.
- The system misses some of the traffic, **Server side data is trusted** is not enabled, and the session includes at least one packet from each side with an Application record that is not answered with an Alert record.
- The system observes all packets in a session, **Server side data is trusted** is enabled, and the session includes a Finished message from the client and at least one packet from the client with an Application record and without an Alert record.
- The system misses some of the traffic, **Server side data is trusted** is enabled, and the session includes at least one packet from the client with an Application record that is not answered with an Alert record.

If you choose to stop processing on encrypted traffic, the system ignores future packets in a session after it marks the session as encrypted.

In addition, during the SSL handshake, the preprocessor monitors heartbeat requests and responses. The preprocessor generates an event if it detects:

- a heartbeat request containing a payload length value greater than the payload itself
- a heartbeat response that is larger than the value stored in the Max Heartbeat Length field

Note You can add the `ssl_state` and `ssl_version` keywords to a rule to use SSL state or version information within the rule.

SSL Preprocessor Options

Note The system-provided network analysis policies enable the SSL preprocessor by default. Cisco recommends that you do not disable the SSL preprocessor in custom deployments if you expect encrypted traffic to cross your network.

Without SSL inspection configured, the system attempts to inspect encrypted traffic for malware and intrusions without decrypting it. When you enable the SSL preprocessor, it detects when a session becomes encrypted. After the SSL preprocessor is enabled, the rules engine can invoke the preprocessor to obtain SSL state and version information. If you enable rules using the `ssl_state` and `ssl_version` keywords in an intrusion policy, you should also enable the SSL preprocessor in that policy.

Ports

Specifies the ports, separated by commas, where the SSL preprocessor should monitor traffic for encrypted sessions. Only ports specified in this field will be checked for encrypted traffic.

Note If the SSL preprocessor detects non-SSL traffic over the ports specified for SSL monitoring, it tries to decode the traffic as SSL traffic, and then flags it as corrupt.

Stop inspecting encrypted traffic

Enables or disables inspection of traffic in a session after the session is marked as encrypted.

Enable this option to disable inspection and reassembly for encrypted sessions. The SSL preprocessor maintains state for the session so it can disable inspection of all traffic in the session. When this option is enabled a few packets of a session are verified to ensure the flow is encrypted after which deep inspection is bypassed. Every bypassed session increases the fast-forwarded flows count shown in the response of the **show snort statistics** command. Moreover, since deep inspection is bypassed, the initiator and responder bytes in the connection event are not accurate. They are less than the value of the actual session, since it only includes the packets inspected by Snort and it does not include any packets after the deep inspection is bypassed. This behavior holds good for connection summary events and all traffic values shown in the widgets.

The system only stops inspecting traffic in encrypted sessions if both:

- SSL preprocessing is enabled
- this option is selected

If you clear this option, you cannot modify the **Server side data is trusted** option.

Server side data is trusted

When Stop inspecting encrypted traffic is enabled, enables identification of encrypted traffic based only on the client-side traffic,

Max Heartbeat Length

By specifying a number of bytes, enables inspection of heartbeat requests and responses within the SSL handshake for Heartbleed bug exploit attempts. You can specify an integer from `1` to `65535`, or `0` to disable the option.

If the preprocessor detects a heartbeat request whose payload length is greater than the actual payload length and rule 137:3 is enabled, or a heartbeat response greater in size than the value configured for this option when rule 137:4 is enabled, the preprocessor generates an event and, in an inline deployment, drops offending packets.

Configuring the SSL Preprocessor

Note This section applies to Snort 2 preprocessors. For information on Snort 3 inspectors, see https://www.cisco.com/go/snort3-inspectors.

Procedure

Step 1 Choose **Policies** > **Access Control heading** > **Access Control**, and then click **Network Analysis Policy** or **Policies** > **Access Control heading** > **Intrusion**, and then click **Network Analysis Policies**.

Note
If your custom user role limits access to the first path listed here, use the second path to access the policy.

Step 2 Click **Snort 2 Version** next to the policy you want to edit.

Step 3 Click **Edit** (✎) next to the policy you want to edit.

If **View** (👁) appears instead, the configuration belongs to an ancestor domain, or you do not have permission to modify the configuration.

Step 4 Click **Settings** in the navigation panel.

Step 5 If **SSL Configuration** under **Application Layer Preprocessors** is disabled, click **Enabled**.

Step 6 Click **Edit** (✎) next to **SSL Configuration**.

Step 7 Modify any of the settings described in SSL Preprocessor Options, on page 292.

- Enter a value in the **Ports** field. Separate multiple values with commas.
- Check or clear the **Stop inspecting encrypted traffic** check box.
- If you checked **Stop inspecting encrypted traffic**, check or clear **Server side data is trusted**.
- Enter a value in the **Max Heartbeat Length** field.

Tip
A value of `0` disables this option.

Step 8 To save changes you made in this policy since the last policy commit, click **Policy Information**, then click **Commit Changes**.

If you leave the policy without committing changes, cached changes since the last commit are discarded if you edit a different policy.

What to do next

- If you want to enable intrusion events, enable SSL preprocessor rules (GID 137). For more information, see Setting Intrusion Rule States, on page 42.
- Deploy configuration changes.

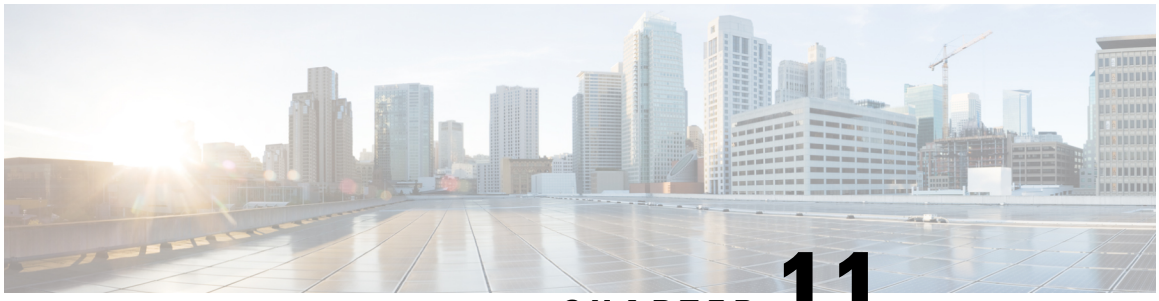
SSL Preprocessor Rules

If you want to generate events and, in an inline deployment, drop offending packets, enable SSL preprocessor rules (GID 137).

The following table describes the SSL preprocessor rules you can enable.

Table 106: SSL Preprocessor Rules

Preprocessor Rule GID:SID	Description
137:1	Detects a ClientHello message after a ServerHello message, which is invalid and considered to be anomalous behavior.
137:2	Detects a ServerHello message without a ClientHello message when the SSL preprocessor option **Server side data is trusted** is disabled, which is invalid and considered to be anomalous behavior.
137:3	Detects a heartbeat request with a payload length greater than the payload itself when the SSL preprocessor option **Max Heartbeat Length** contains a non-zero value, which indicates an attempt to exploit the Heartbleed bug.
137:4	Detects a heartbeat response larger than a non-zero value specified in the SSL preprocessor option **Max Heartbeat Length**, which indicates an attempt to exploit the Heartbleed bug.



CHAPTER 11

SCADA Preprocessors

The following topics explain preprocessors for Supervisory Control and Data Acquisition (SCADA) protocols, and how to configure them:

Introduction to SCADA Preprocessors

Note This section applies to Snort 2 preprocessors. For information on Snort 3 inspectors, see https://www.cisco.com/go/snort3-inspectors.

Supervisory Control and Data Acquisition (SCADA) protocols monitor, control, and acquire data from industrial, infrastructure, and facility processes such as manufacturing, production, water treatment, electric power distribution, airport and shipping systems, and so on. The system provides preprocessors for the Modbus, Distributed Network Protocol (DNP3), Common Industrial Protocol (CIP), and S7Commplus SCADA protocols that you can configure as part of your network analysis policy.

If the Modbus, DNP3, CIP, or S7Commplus preprocessor is disabled, and you enable and deploy an intrusion rule that requires one of these preprocessors, the system automatically uses the required preprocessor, with its current settings, although the preprocessor remains disabled in the web interface for the corresponding network analysis policy.

License Requirements for SCADA Preprocessors

Threat Defense License

IPS

Requirements and Prerequisites for SCADA Preprocessors

Model support

Any.

Supported domains

Any

User roles

- Admin
- Intrusion Admin

The Modbus Preprocessor

Note This section applies to Snort 2 preprocessors. For information on Snort 3 inspectors, see https://www.cisco.com/go/snort3-inspectors.

The Modbus protocol, which was first published in 1979 by Modicon, is a widely used SCADA protocol. The Modbus preprocessor detects anomalies in Modbus traffic and decodes the Modbus protocol for processing by the rules engine, which uses Modbus keywords to access certain protocol fields.

A single configuration option allows you to modify the default setting for the port that the preprocessor inspects for Modbus traffic.

Modbus Preprocessor Ports Option

Ports

Specifies the ports that the preprocessor inspects for Modbus traffic. Separate multiple ports with commas.

Configuring the Modbus Preprocessor

Note This section applies to Snort 2 preprocessors. For information on Snort 3 inspectors, see https://www.cisco.com/go/snort3-inspectors.

You should not enable this preprocessor in a network analysis policy that you apply to traffic if your network does not contain any Modbus-enabled devices.

Procedure

Step 1 Choose **Policies** > **Access Control heading** > **Access Control**, and then click **Network Analysis Policy** or **Policies** > **Access Control heading** > **Intrusion**, and then click **Network Analysis Policies**.

Note
If your custom user role limits access to the first path listed here, use the second path to access the policy.

Step 2 Click **Snort 2 Version** next to the policy you want to edit.

Step 3 Click **Edit** (✎) next to the policy you want to edit.

If **View** (👁) appears instead, the configuration belongs to an ancestor domain, or you do not have permission to modify the configuration.

Step 4 Click **Settings** in the navigation panel.

Step 5 If **Modbus Configuration** under **SCADA Preprocessors** is disabled, click **Enabled**.

Step 6 Click **Edit** (✎) next to **Modbus Configuration**.

Step 7 Enter a value in the **Ports** field.

Separate multiple values with commas.

Step 8 To save changes you made in this policy since the last policy commit, click **Policy Information**, then click **Commit Changes**.

If you leave the policy without committing changes, cached changes since the last commit are discarded if you edit a different policy.

What to do next

- If you want to generate events and, in an inline deployment, drop offending packets, enable Modbus preprocessor rules (GID 144). For more information, see Setting Intrusion Rule States, on page 42 and Modbus Preprocessor Rules, on page 297.
- Deploy configuration changes.

Modbus Preprocessor Rules

You must enable the Modbus preprocessor rules in the following table if you want these rules to generate events and, in an inline deployment, drop offending packets.

Table 107: Modbus Preprocessor Rules

Preprocessor Rule GID:SID	Description
144:1	Generates an event when the length in the Modbus header does not match the length required by the Modbus function code. Each Modbus function has an expected format for requests and responses. If the length of the message does not match the expected format, this event is generated.
144:2	Generates an event when the Modbus protocol ID is non-zero. The protocol ID field is used for multiplexing other protocols with Modbus. Because the preprocessor does not process these other protocols, this event is generated instead.
144:3	Generates an event when the preprocessor detects a reserved Modbus function code.

The DNP3 Preprocessor

Note This section applies to Snort 2 preprocessors. For information on Snort 3 inspectors, see https://www.cisco.com/go/snort3-inspectors.

The Distributed Network Protocol (DNP3) is a SCADA protocol that was originally developed to provide consistent communication between electrical stations. DNP3 has also become widely used in the water, waste, transportation, and many other industries.

The DNP3 preprocessor detects anomalies in DNP3 traffic and decodes the DNP3 protocol for processing by the rules engine, which uses DNP3 keywords to access certain protocol fields.

DNP3 Preprocessor Options

Ports

Enables inspection of DNP3 traffic on each specified port. You can specify a single port or a comma-separated list of ports.

Log bad CRCs

Validates the checksums contained in DNP3 link layer frames. Frames with invalid checksums are ignored.

You can enable rule 145:1 to generate events and, in an inline deployment, drop offending packets when invalid checksums are detected.

Configuring the DNP3 Preprocessor

Note This section applies to Snort 2 preprocessors. For information on Snort 3 inspectors, see https://www.cisco.com/go/snort3-inspectors.

You should not enable this preprocessor in a network analysis policy that you apply to traffic if your network does not contain any DNP3-enabled devices.

Procedure

Step 1 Choose **Policies** > **Access Control heading** > **Access Control**, and then click **Network Analysis Policy** or **Policies** > **Access Control heading** > **Intrusion**, and then click **Network Analysis Policies**.

Note
If your custom user role limits access to the first path listed here, use the second path to access the policy.

Step 2 Click **Snort 2 Version** next to the policy you want to edit.

Step 3 Click **Edit** (✎) next to the policy you want to edit.

If **View** (👁) appears instead, the configuration belongs to an ancestor domain, or you do not have permission to modify the configuration.

Step 4 Click **Settings** in the navigation panel.

Step 5 If **DNP3 Configuration** under **SCADA Preprocessors** is disabled, click **Enabled**.

Step 6 Click **Edit** (✎) next to **DNP3 Configuration**.

Step 7 Enter a value for **Ports**.

Separate multiple values with commas.

Step 8 Check or clear the **Log bad CRCs** check box.

Step 9 To save changes you made in this policy since the last policy commit, click **Policy Information**, then click **Commit Changes**.

If you leave the policy without committing changes, cached changes since the last commit are discarded if you edit a different policy.

What to do next

- If you want to generate events and, in an inline deployment, drop offending packets, enable DNP3 preprocessor rules (GID 145). For more information, see Setting Intrusion Rule States, on page 42, DNP3 Preprocessor Options, on page 298, and DNP3 Preprocessor Rules, on page 300.
- Deploy configuration changes.

DNP3 Preprocessor Rules

You must enable the DNP3 preprocessor rules in the following table if you want these rules to generate events and, in an inline deployment, drop offending packets.

Table 108: DNP3 Preprocessor Rules

Preprocessor Rule GID:SID	Description
145:1	When **Log bad CRC** is enabled, generates an event when the preprocessor detects a link layer frame with an invalid checksum.
145:2	Generates an event and blocks the packet when the preprocessor detects a DNP3 link layer frame with an invalid length.
145:3	Generates an event and blocks the packet during reassembly when the preprocessor detects a transport layer segment with an invalid sequence number.
145:4	Generates an event when the DNP3 reassembly buffer is cleared before a complete fragment can be reassembled. This happens when a segment carrying the FIR flag appears after other segments have been queued.
145:5	Generates an event when the preprocessor detects a DNP3 link layer frame that uses a reserved address.
145:6	Generates an event when the preprocessor detects a DNP3 request or response that uses a reserved function code.

The CIP Preprocessor

Note This section applies to Snort 2 preprocessors. For information on Snort 3 inspectors, see https://www.cisco.com/go/snort3-inspectors.

The Common Industrial Protocol (CIP) is a widely used application protocol that supports industrial automation applications. EtherNet/IP (ENIP) is an implementation of CIP that is used on Ethernet-based networks.

The CIP preprocessor detects CIP and ENIP traffic running on TCP or UDP and sends it to the intrusion rules engine. You can use CIP and ENIP keywords in custom intrusion rules to detect attacks in CIP and ENIP traffic. See CIP and ENIP Keywords. Additionally, you can control traffic by specifying CIP and ENIP application conditions in access control rules.

CIP Preprocessor Options

Ports

Specifies the ports to inspect for CIP and ENIP traffic. You can specify an integer from 0 to 65535. Separate multiple port numbers with commas.

Note You must add the default CIP detection port 44818 and any other ports you list to the TCP stream **Perform Stream Reassembly on Both Ports** list.

Default Unconnected Timeout (seconds)

When a CIP request message does not contain a protocol-specific timeout value and **Maximum number of concurrent unconnected requests per TCP connection** is reached, the system times the message for the number of seconds specified by this option. When the timer expires, the message is removed to make room for future requests. You can specify an integer from 0 to 360. When you specify 0, all traffic that does not have a protocol-specific timeout times out first.

Maximum number of concurrent unconnected requests per TCP connection

The number of concurrent requests that can go unanswered before the system closes the connection. You can specify an integer from 1 to 10000.

Maximum number of CIP connections per TCP connection

The maximum number of simultaneous CIP connections allowed by the system per TCP connection. You can specify an integer from 1 to 10000.

CIP Events

By design, application detectors detect and event viewers display the same application one time per session. A CIP session can include multiple applications in different packets, and a single CIP packet can contain multiple applications. The CIP preprocessor handles all CIP and ENIP traffic according to the corresponding intrusion rule.

The following table shows the CIP values displayed in event views.

Table 109: CIP Event Field Values

Event Field	Displayed Value
Application Protocol	CIP or ENIP
Client	CIP Client or ENIP Client

Event Field	Displayed Value
Web Application	The specific application detected, which is: • For access control rules that allow or monitor traffic, the last application protocol det Access control rules that you configure to log connections might not generate even applications, and access control rules that you do not configure to log connections events for CIP applications. • For access control rules that block traffic, the application protocol that triggered th When an access control rule blocks a list of CIP applications, event viewers display that is detected.

CIP Preprocessor Rules

If you want the CIP preprocessor rules listed in the following table to generate events, you must enable them. See Setting Intrusion Rule States, on page 42 for information on enabling rules.

Table 110: CIP Preprocessor Rules

GID:SID	Rule Message
148:1	CIP_MALFORMED
148:2	CIP_NON_CONFORMING
148:3	CIP_CONNECTION_LIMIT
148:4	CIP_REQUEST_LIMIT

Guidelines for Configuring the CIP Preprocessor

Note the following when configuring the CIP preprocessor:

- You must add the default CIP detection port 44818 and any other CIP **Ports** you list to the TCP stream **Perform Stream Reassembly on Both Ports** list.
- Event viewers give special handling to CIP applications. See CIP Events, on page 301.
- We recommend that you use an intrusion prevention action as the default action of your access control policy.
- The CIP preprocessor does not support an access control policy default action of **Access Control: Trust All Traffic**, which may produce undesirable behavior, including not dropping traffic triggered by CIP applications specified in intrusion rules and access control rules.
- The CIP preprocessor does not support an access control policy default action of **Access Control: Block All Traffic**, which may produce undesirable behavior, including blocking CIP applications that you do not expect to be blocked.
- The CIP preprocessor does not support application visibility for CIP applications, including network discovery.

- To detect CIP and ENIP applications and use them in access control rules, intrusion rules and so on, you must manually enable the CIP preprocessor in the corresponding custom network analysis policy. See Creating a Custom Network Analysis Policy, on page 74, Setting the Default Network Analysis Policy, and Configuring Network Analysis Rules, on page 85.
- To drop traffic that triggers CIP preprocessor rules and CIP intrusion rules, ensure that **Drop when Inline** is enabled in the corresponding intrusion policy.
- To block CIP or ENIP application traffic using access control rules, ensure that the inline normalization preprocessor and its **Inline Mode** option are enabled (the default setting) in the corresponding network analysis policy. See Creating a Custom Network Analysis Policy, on page 74, Setting the Default Network Analysis Policy, and Preprocessor Traffic Modification in Inline Deployments, on page 77.

Configuring the CIP Preprocessor

Note This section applies to Snort 2 preprocessors. For information on Snort 3 inspectors, see https://www.cisco.com/go/snort3-inspectors.

Before you begin

- You must add the default CIP detection port 44818 and any other ports you list as CIP **Ports** to the TCP stream **Perform Stream Reassembly on Both Ports** list.
- Familiarize yourself with Guidelines for Configuring the CIP Preprocessor, on page 302.
- The CIP preprocessor is not supported for Firewall Threat Defense devices.

Procedure

Step 1 Choose **Policies** > **Access Control heading** > **Access Control**, and then click **Network Analysis Policy** or **Policies** > **Access Control heading** > **Intrusion**, and then click **Network Analysis Policies**.

Note
If your custom user role limits access to the first path listed here, use the second path to access the policy.

Step 2 Click **Snort 2 Version** next to the policy you want to edit.

Step 3 Click **Edit** (✎) next to the policy you want to edit.

If **View** (👁) appears instead, the configuration belongs to an ancestor domain, or you do not have permission to modify the configuration.

Step 4 Click **Settings** in the navigation panel.

Step 5 If **CIP Configuration** under **SCADA Preprocessors** is disabled, click **Enabled**.

Step 6 You can modify any of the options described in CIP Preprocessor Options, on page 301.

Step 7 To save changes you made in this policy since the last policy commit, click **Policy Information**, then click **Commit Changes**.

If you leave the policy without committing changes, cached changes since the last commit are discarded if you edit a different policy.

What to do next

- If you want to generate events and, in an inline deployment, drop offending packets, enable CIP intrusion rules and, optionally, CIP preprocessor rules (GID 148). For more information, see Setting Intrusion Rule States, on page 42, CIP Preprocessor Rules, on page 302, and CIP Events, on page 301.
- Deploy configuration changes.

The S7Commplus Preprocessor

Note This section applies to Snort 2 preprocessors. For information on Snort 3 inspectors, see https://www.cisco.com/go/snort3-inspectors.

The S7Commplus preprocessor detects S7Commplus traffic. You can use S7Commplus keywords in custom intrusion rules to detect attacks in S7Commplus traffic. See S7Commplus Keywords, on page 177.

Configuring the S7Commplus Preprocessor

Note This section applies to Snort 2 preprocessors. For information on Snort 3 inspectors, see https://www.cisco.com/go/snort3-inspectors.

The S7Commplus preprocessor is supported on all Firewall Threat Defense devices.

Procedure

Step 1 Choose **Policies** > **Access Control heading** > **Access Control**, and then click **Network Analysis Policy** or **Policies** > **Access Control heading** > **Intrusion**, and then click **Network Analysis Policies**.

Note
If your custom user role limits access to the first path listed here, use the second path to access the policy.

Step 2 Click **Snort 2 Version** next to the policy you want to edit.

Step 3 Click **Edit** (✎) next to the policy you want to edit.

If **View** (👁) appears instead, the configuration belongs to an ancestor domain, or you do not have permission to modify the configuration.

Step 4 Click **Settings** in the navigation panel.

Step 5 If **S7Commplus Configuration** under **SCADA Preprocessors** is disabled, click **Enabled**.

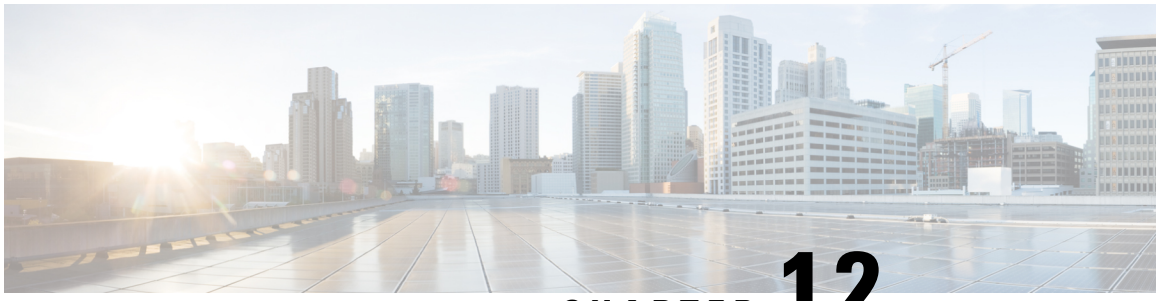
Step 6 Optionally, click **Edit** (✎) next to **S7Commplus Configuration** and modify **s7commplus_ports** to identify ports that the preprocessor inspects for S7Commplus traffic. Separate multiple ports with commas.

Step 7 To save changes you made in this policy since the last policy commit, click **Policy Information**, then click **Commit Changes**.

If you leave the policy without committing changes, cached changes since the last commit are discarded if you edit a different policy.

What to do next

- If you want to generate intrusion events, enable S7Commplus preprocessor rules (GID 149). For more information, see Setting Intrusion Rule States, on page 42
- Deploy configuration changes.



CHAPTER 12

Specific Threat Detection

The following topics explain how to use preprocessors in a network analysis policy to detect specific threats:

Introduction to Specific Threat Detection

Note This section applies to Snort 2 preprocessors. For information on Snort 3 inspectors, see https://www.cisco.com/go/snort3-inspectors.

You can use several preprocessors in a network analysis policy to detect specific threats to your monitored network, such as Back Orifice attacks, several portscan types, and rate-based attacks that attempt to overwhelm your network with excessive traffic. When the GID Signatures specific to pre-processor is enabled, the Network Analysis Policy on Web will show disabled. However, the pre-processors will be turned on device using the available default settings.

You can also use sensitive data detection, which you configure in an intrusion policy, to detect unsecured transmission of sensitive numerical data.

License Requirements for Specific Threat Detection

Threat Defense License

IPS

Requirements and Prerequisites for Specific Threat Detection

Model support

Any.

Supported domains

Any

User roles

- Admin
- Intrusion Admin

Back Orifice Detection

Note This section applies to Snort 2 preprocessors. For information on Snort 3 inspectors, see https://www.cisco.com/go/snort3-inspectors.

The system provides a preprocessor that detects the existence of the Back Orifice program. This program can be used to gain admin access to your Windows hosts.

Back Orifice Detection Preprocessor

The Back Orifice preprocessor analyzes UDP traffic for the Back Orifice magic cookie, "`*!*QWTY?`", which is located in the first eight bytes of the packet and is XOR-encrypted.

The Back Orifice preprocessor has a configuration page, but no configuration options. When it is enabled, you must also enable preprocessor rules for the preprocessor to generate events and, in an inline deployment, drop offending packets.

Table 111: Back Orifice GID:SIDs

Preprocessor rule GID:SID	Description
105:1	Back Orifice traffic detected
105:2	Back Orifice client traffic detected
105:3	Back Orifice server traffic detected
105:4	Back Orifice Snort buffer attack detected

Detecting Back Orifice

Note This section applies to Snort 2 preprocessors. For information on Snort 3 inspectors, see https://www.cisco.com/go/snort3-inspectors.

Procedure

Step 1 Choose **Policies** > **Access Control heading** > **Access Control**, and then click **Network Analysis Policy** or **Policies** > **Access Control heading** > **Intrusion**, and then click **Network Analysis Policies**.

Note
If your custom user role limits access to the first path listed here, use the second path to access the policy.

Step 2 Click **Snort 2 Version** next to the policy you want to edit.

Step 3 Click **Edit** ( ) next to the policy you want to edit.

If **View** ( ) appears instead, the configuration belongs to an ancestor domain, or you do not have permission to modify the configuration.

Step 4 Click **Settings** in the navigation panel.

Step 5 If **Back Orifice Detection** under **Specific Threat Detection** is disabled, click **Enabled**.

Note
There are no user-configurable options for Back Orifice.

Step 6 To save changes you made in this policy since the last policy commit, click **Policy Information**, then click **Commit Changes**.

If you leave the policy without committing changes, changes since the last commit are discarded if you edit a different policy.

What to do next

- If you want to generate events and, in an inline deployment, drop offending packets, enable Back Orifice Detection rules 105:1, 105:2, 105:3, or 105:4. For more information, see Intrusion Rule States, on page 42 and Back Orifice Detection Preprocessor, on page 308.
- Deploy configuration changes.

Portscan Detection

Note This section applies to Snort 2 preprocessors. For information on Snort 3 inspectors, see https://www.cisco.com/go/snort3-inspectors.

A portscan is a form of network reconnaissance that is often used by attackers as a prelude to an attack. In a portscan, an attacker sends specially crafted packets to a targeted host. By examining the packets that the host responds with, the attacker can often determine which ports are open on the host and, either directly or by inference, which application protocols are running on these ports.

By itself, a portscan is not evidence of an attack. In fact, some of the portscanning techniques used by attackers can also be employed by legitimate users on your network. Cisco's portscan detector is designed to help you determine which portscans might be malicious by detecting patterns of activity.

Attention Devices load-balance inspection across internal resources. If portscan detection is not working as expected, you may need to configure the sensitivity level as **High**.

We strongly recommend that you upgrade to Snort 3 and use the portscan feature introduced in version 7.2.0. For more details, see the *Custom Snort 3 Intrusion Policies for Access Control* and the *Snort 3 Inspector Reference*.

Portscan Types, Protocols, and Filtered Sensitivity Levels

Note This section applies to Snort 2 preprocessors. For information on Snort 3 inspectors, see https://www.cisco.com/go/snort3-inspectors.

Attackers are likely to use several methods to probe your network. Often they use different protocols to draw out different responses from a target host, hoping that if one type of protocol is blocked, another may be available.

Table 112: Protocol Types

Protocol	Description
TCP	Detects TCP probes such as SYN scans, ACK scans, TCP connect() scans, and scans with unusual flag combinations such as Xmas tree, FIN, and NULL
UDP	Detects UDP probes such as zero-byte UDP packets
ICMP	Detects ICMP echo requests (pings)
IP	Detects IP protocol scans. These scans differ from TCP and UDP scans because the attacker, instead of looking for open ports, is trying to discover which IP protocols are supported on a target host.

Portscans are generally divided into four types based on the number of targeted hosts, the number of scanning hosts, and the number of ports that are scanned.

Table 113: Portscan Types

Type	Description
Portscan Detection	A one-to-one portscan in which an attacker uses one or a few hosts to scan multiple ports on a single target host. One-to-one portscans are characterized by: • a low number of scanning hosts • a single host that is scanned • a high number of ports scanned This option detects TCP, UDP, and IP portscans.
Port Sweep	A one-to-many portsweep in which an attacker uses one or a few hosts to scan a single port on multiple target hosts. Portsweeps are characterized by: • a low number of scanning hosts • a high number of scanned hosts • a low number of unique ports scanned This option detects TCP, UDP, ICMP, and IP portsweeps.
Decoy Portscan	A one-to-one portscan in which the attacker mixes spoofed source IP addresses with the actual scanning IP address. Decoy portscans are characterized by: • a high number of scanning hosts • a low number of ports that are scanned only once • a single (or a low number of) scanned hosts The decoy portscan option detects TCP, UDP, and IP protocol portscans.
Distributed Portscan	A many-to-one portscan in which multiple hosts query a single host for open ports. Distributed portscans are characterized by: • a high number of scanning hosts • a high number of ports that are scanned only once • a single (or a low number of) scanned hosts The distributed portscan option detects TCP, UDP, and IP protocol portscans.

The information that the portscan detector learns about a probe is largely based on seeing negative responses from the probed hosts. For example, when a web client tries to connect to a web server, the client uses port 80/tcp and the server can be counted on to have that port open. However, when an attacker probes a server, the attacker does not know in advance if it offers web services. When the portscan detector sees a negative response (that is, an ICMP unreachable or TCP RST packet), it records the response as a potential portscan. The process is more difficult when the targeted host is on the other side of a device such as a firewall or router that filters negative responses. In this case, the portscan detector can generate *filtered* portscan events based on the sensitivity level that you select.

Table 114: Sensitivity Levels

Level	Description
Low	Detects only negative responses from targeted hosts. Select this sensitivity level to suppress false positives, but keep in mind that some types of portscans (slow scans, filtered scans) might be missed. This level uses the shortest time window for portscan detection.
Medium	Detects portscans based on the number of connections to a host, which means that you can detect filtered portscans. However, very active hosts such as network address translators and proxies may generate false positives. Note that you can add the IP addresses of these active hosts to the **Ignore Scanned** field to mitigate this type of false positive. This level uses a longer time window for portscan detection.
High	Detects portscans based on a time window, which means that you can detect time-based portscans. However, if you use this option, you should be careful to tune the detector over time by specifying IP addresses in the **Ignore Scanned** and **Ignore Scanner** fields. This level uses a much longer time window for portscan detection.

Portscan Event Generation

When portscan detection is enabled, you must enable rules with Generator ID (GID) 122 and a Snort ID (SID) from among SIDs 1 through 27 to detect the various portscans and portsweeps.

Note For events generated by the portscan connection detector, the protocol number is set to 255. Because portscan does not have a specific protocol associated with it by default, the Internet Assigned Numbers Authority (IANA) does not have a protocol number assigned to it. IANA designates 255 as a reserved number, so that number is used in portscan events to indicate that there is not an associated protocol for the event.

Table 115: Portscan Detection SIDs (GID 122)

Portscan Type	Protocol	Sensitivity Level	Preprocessor Rule SID
Portscan Detection	TCP	Low	1
	UDP	Medium or High	5
	ICMP	Low	17
	IP	Medium or High	21
		Low	Does not generate events.
		Medium or High	Does not generate events.
		Low	9
		Medium or High	13
Port Sweep	TCP	Low	3, 27
	UDP	Medium or High	7
	ICMP	Low	19
	IP	Medium or High	23
		Low	25
		Medium or High	26
		Low	11
		Medium or High	15
Decoy Portscan	TCP	Low	2
	UDP	Medium or High	6
	ICMP	Low	18
	IP	Medium or High	22
		Low	Does not generate events.
		Medium or High	Does not generate events.
		Low	10
		Medium or High	14

Portscan Type	Protocol	Sensitivity Level	Preprocessor Rule SID
Distributed Portscan	TCP	Low	4
	UDP	Medium or High	8
	ICMP	Low	20
	IP	Medium or High	24
		Low	Does not generate events.
		Medium or High	Does not generate events.
		Low	12
		Medium or High	16

Portscan Event Packet View

When you enable the accompanying preprocessor rules, the portscan detector generates intrusion events that you can view just as you would any other intrusion event. However, the information presented on the packet view is different from the other types of intrusion events.

Begin by using the intrusion event views to drill down to the packet view for a portscan event. Note that you cannot download a portscan packet because single portscan events are based on multiple packets; however, the portscan packet view provides all usable packet information.

For any IP address, you can click the address to view the context menu and select **whois** to perform a lookup on the IP address or **View Host Profile** to view the host profile for that host.

Table 116: Portscan Packet View

Information	Description
Device	The device that detected the event.
Time	The time when the event occurred.
Message	The event message generated by the preprocessor.
Source IP	The IP address of the scanning host.
Destination IP	The IP address of the scanned host.
Priority Count	The number of negative responses (for example, TCP RSTs and ICMP unreachables) from the scanned host. The higher the number of negative responses, the higher the priority count.
Connection Count	The number of active connections on the hosts. This value is more accurate for connection-based scans such as TCP and IP.

Information	Description
IP Count	The number of times that the IP addresses that contact the scanned host changes. For example, if the first IP address is 10.1.1.1, the second IP is 10.1.1.2, and the third IP is 10.1.1.1, then the IP count is 3. This number is less accurate for active hosts such as proxies and DNS servers.
Scanner/Scanned IP Range	The range of IP addresses for the scanned hosts or the scanning hosts, depending on the type of scan. For portsweeps, this field shows the IP range of scanned hosts. For portscans, this shows the IP range of the scanning hosts.
Port/Proto Count	For TCP and UDP portscans, the number of times that the port being scanned changes. For example, if the first port scanned is 80, the second port scanned is 8080, and the third port scanned is again 80, then the port count is 3. For IP protocol portscans, the number of times that the protocol being used to connect to the scanned host changes.
Port/Proto Range	For TCP and UDP portscans, the range of the ports that were scanned. For IP protocol portscans, the range of IP protocol numbers that were used to attempt to connect to the scanned host.
Open Ports	The TCP ports that were open on the scanned host. This field appears only when the portscan detects one or more open ports.

Configuring Portscan Detection

Note This section applies to Snort 2 preprocessors. For information on Snort 3 inspectors, see https://www.cisco.com/go/snort3-inspectors.

The portscan detection configuration options allow you to finely tune how the portscan detector reports scan activity.

Procedure

Step 1 Choose **Policies** > **Access Control heading** > **Access Control**, and then click **Network Analysis Policy** or **Policies** > **Access Control heading** > **Intrusion**, and then click **Network Analysis Policies**.

Note
If your custom user role limits access to the first path listed here, use the second path to access the policy.

Step 2 Click **Snort 2 Version** next to the policy you want to edit.

Step 3 Click **Edit** ( ) next to the policy you want to edit.

If **View** ( ) appears instead, the configuration belongs to an ancestor domain, or you do not have permission to modify the configuration.

Step 4 Click **Settings**.

Step 5 If **Portscan Detection** under **Specific Threat Detection** is disabled, click **Enabled**.

Step 6 Click **Edit** (✎) next to **Portscan Detection**.

Step 7 In the **Protocol** field, specify protocols to enable.

Note
You must ensure TCP stream processing is enabled to detect scans over TCP, and that UDP stream processing is enabled to detect scans over UDP.

Step 8 In the **Scan Type** field, specify portscan types you want to detect.

Step 9 Choose a level from the **Sensitivity Level** list; see Portscan Types, Protocols, and Filtered Sensitivity Levels, on page 310.

Step 10 If you want to monitor specific hosts for signs of portscan activity, enter the host IP address in the **Watch IP** field.

You can specify a single IP address or address block, or a comma-separated lists of either or both. Leave the field blank to watch all network traffic.

Step 11 If you want to ignore hosts as scanners, enter the host IP address in the **Ignore Scanners** field.

You can specify a single IP address or address block, or a comma-separated lists of either or both.

Step 12 If you want to ignore hosts as targets of a scan, enter the host IP address in the **Ignore Scanned** field.

You can specify a single IP address or address block, or a comma-separated lists of either or both.

Tip
Use the **Ignore Scanners** and **Ignore Scanned** fields to indicate hosts on your network that are especially active. You may need to modify this list of hosts over time.

Step 13 If you want to discontinue monitoring of sessions picked up in mid-stream, clear the **Detect Ack Scans** check box.

Note
Detection of mid-stream sessions helps to identify ACK scans, but may cause false events, particularly on networks with heavy traffic and dropped packets.

Step 14 To save changes you made in this policy since the last policy commit, click **Policy Information**, then click **Commit Changes**.

If you leave the policy without committing changes, changes since the last commit are discarded if you edit a different policy.

What to do next

- If you want portscan detection to detect various portscans and portsweeps, enable rules 122:1 through 122:27. For more information, see Intrusion Rule States, on page 42 and Portscan Event Generation, on page 312.
- Deploy configuration changes.

Rate-Based Attack Prevention

Note This section applies to Snort 2 preprocessors. For information on Snort 3 inspectors, see https://www.cisco.com/go/snort3-inspectors.

Rate-based attacks are attacks that depend on frequency of connection or repeated attempts to perpetrate the attack. You can use rate-based detection criteria to detect a rate-based attack as it occurs and respond to it when it happens, then return to normal detection settings after it stops.

You can configure your network analysis policy to include rate-based filters that detect excessive activity directed at hosts on your network. You can use this feature on managed devices deployed in inline mode to block rate-based attacks for a specified time, then revert to only generating events and not drop traffic.

The SYN attack prevention option helps you protect your network hosts against SYN floods. You can protect individual hosts or whole networks based on the number of packets seen over a period of time. If your device is deployed passively, you can generate events. If your device is placed inline, you can also drop the malicious packets. After the timeout period elapses, if the rate condition has stopped, the event generation and packet dropping stops.

For example, you could configure a setting to allow a maximum number of SYN packets from any one IP address, and block further connections from that IP address for 60 seconds.

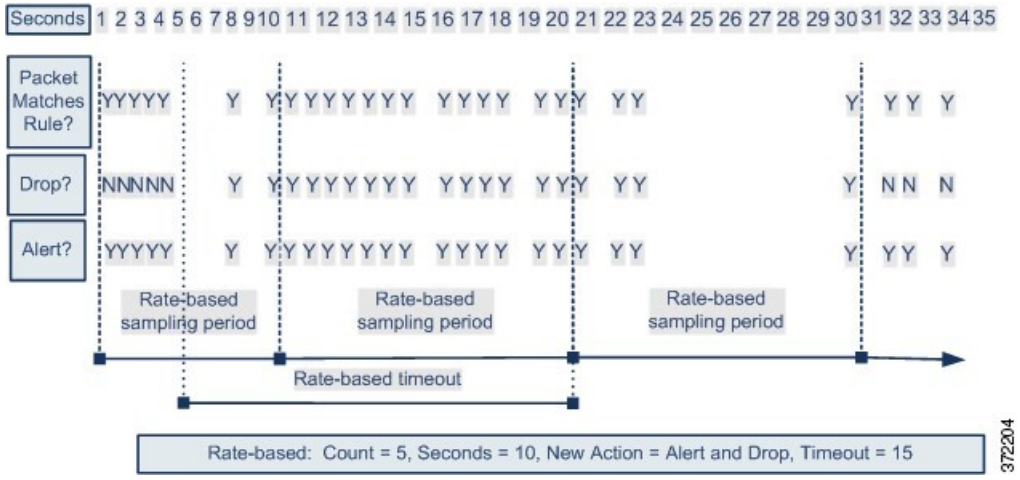
You can also limit TCP/IP connections to or from hosts on your network to prevent denial of service (DoS) attacks or excessive activity by users. When the system detects the configured number of successful connections to or from a specified IP address or range of addresses, it generates events on additional connections. The rate-based event generation continues until the timeout period elapses without the rate condition occurring. In an inline deployment you can choose to drop packets until the rate condition times out.

For example, you could configure a setting to allow a maximum of 10 successful simultaneous connections from any one IP address, and block further connections from that IP address for 60 seconds.

Note Devices load-balance inspection across internal resources. When you configure rate-based attack prevention, you configure the triggering rate per resource, not per device. If rate-based attack prevention is not working as expected, you may need to lower the triggering rate. It triggers alert, if users send too many connection attempts within prescribed time intervals. Hence it is recommended to rate limit the rule. For help determining the correct rate, contact Support.

The following diagram shows an example where an attacker is attempting to access a host. Repeated attempts to find a password trigger a rule which has rate-based attack prevention configured. The rate-based settings change the rule attribute to Drop and Generate Events after rule matches occur five times in a 10-second span. The new rule attribute times out after 15 seconds.

After the timeout, note that packets are still dropped in the rate-based sampling period that follows. If the sampled rate is above the threshold in the current or previous sampling period, the new action continues. The new action reverts to generating events only after a sampling period completes where the sampled rate is below the threshold rate.



Rate-Based Attack Prevention Examples

The `detection_filter` keyword and the thresholding and suppression features provide other ways to filter either the traffic itself or the events that the system generates. You can use rate-based attack prevention alone or in any combination with thresholding, suppression, or the `detection_filter` keyword.

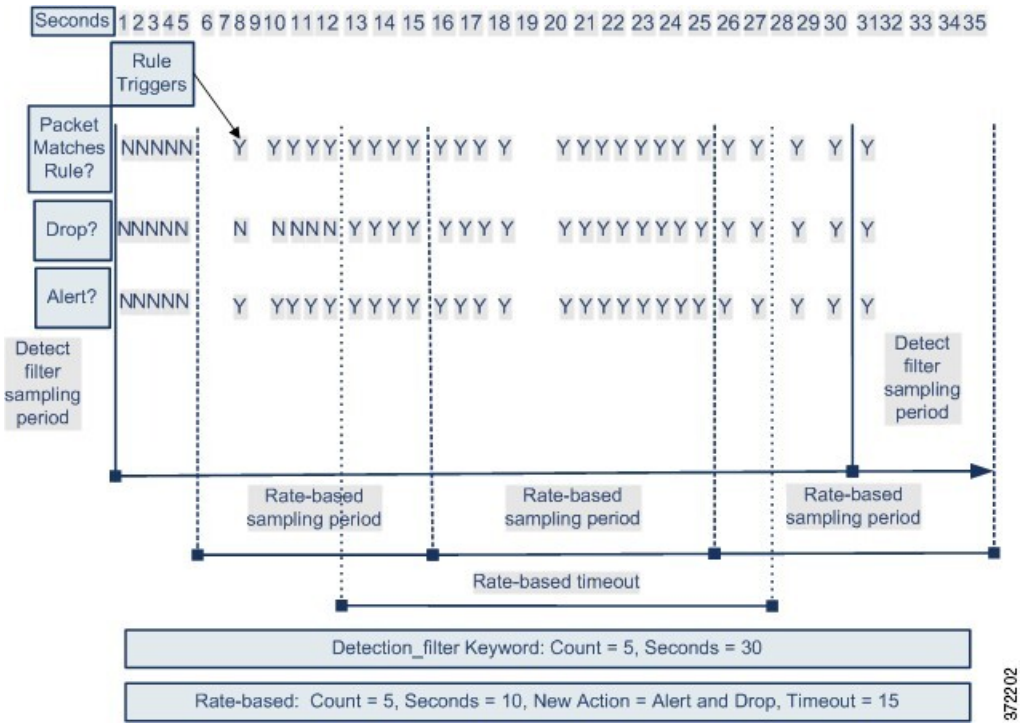
The `detection_filter` keyword, thresholding or suppression, and rate-based criteria may all apply to the same traffic. When you enable suppression for a rule, events are suppressed for the specified IP addresses even if a rate-based change occurs.

detection_filter Keyword Example

The following example shows an attacker attempting a brute-force login. Repeated attempts to find a password trigger a rule that also includes the `detection_filter` keyword, with a count set to 5. This rule has rate-based attack prevention configured. The rate-based settings change the rule attribute to Drop and Generate Events for 20 seconds when there are five hits on the rule in a 10-second span.

As shown in the diagram, the first five packets matching the rule do not generate events because the rule does not trigger until the rate exceeds the rate indicated by the `detection_filter` keyword. After the rule triggers, event notification begins, but the rate-based criteria do not trigger the new action of Drop and Generate Events until five more packets pass.

After the rate-based criteria are met, events are generated and the packets are dropped until the rate-based timeout period expires and the rate falls below the threshold. After twenty seconds elapse, the rate-based action times out. After the timeout, note that packets are still dropped in the rate-based sampling period that follows. Because the sampled rate is above the threshold rate in the previous sampling period when the timeout happens, the rate-based action continues.



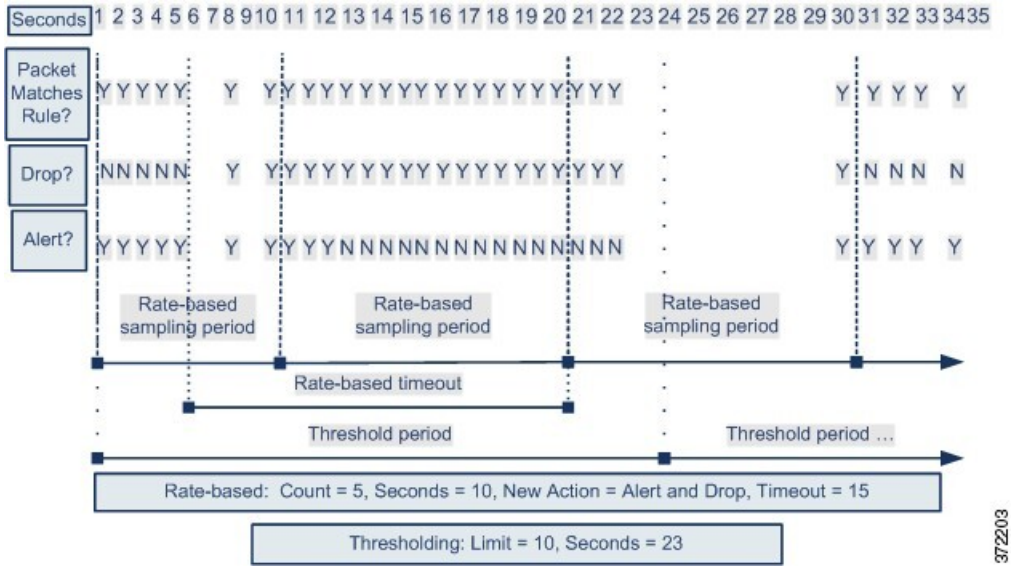
Note that although the example does not depict this, you can use the Drop and Generate Events rule state in combination with the `detection_filter` keyword to start dropping traffic when hits for the rule reach the specified rate. When deciding whether to configure rate-based settings for a rule, consider whether setting the rule to Drop and Generate Events and including the `detection_filter` keyword would achieve the same result, or whether you want to manage the rate and timeout settings in the intrusion policy.

Dynamic Rule State Thresholding or Suppression Example

The following example shows an attacker attempting a brute-force login. Repeated attempts to find a password trigger a rule that has rate-based attack prevention configured. The rate-based settings change the rule attribute to Drop and Generate Events for 15 seconds when there are five hits on the rule in 10 seconds. In addition, a limit threshold limits the number of events the rule can generate to 10 events in 23 seconds.

As shown in the diagram, the rule generates events for the first five matching packets. After five packets, the rate-based criteria trigger the new action of Drop and Generate Events, and for the next five packets the rule generates events and the system drops the packet. After the tenth packet, the limit threshold has been reached, so for the remaining packets the system does not generate events but does drop the packets.

After the timeout, note that packets are still dropped in the rate-based sampling period that follows. If the sampled rate is above the threshold rate in the current or previous sampling period, the new action continues. The new action reverts to Generate Events only after a sampling period completes where the sampled rate is below the threshold rate.



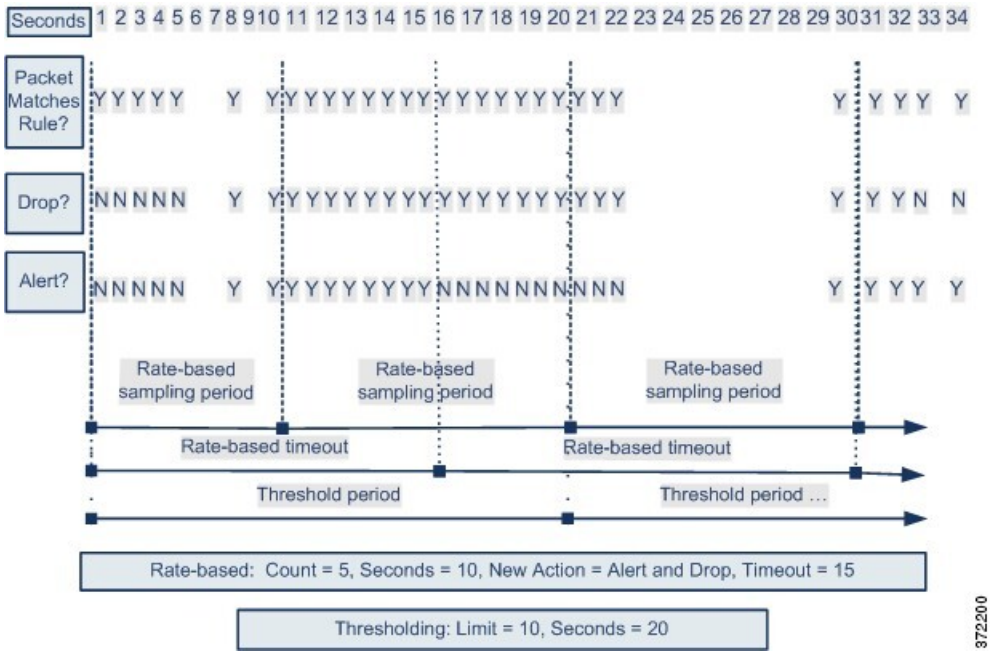
Note that although it is not shown in this example, if a new action triggers because of rate-based criteria *after* a threshold has been reached, the system generates a single event to indicate the change in action. So, for example, when the limit threshold of 10 is reached and the system stops generating events and the action changes from Generate Events to Drop and Generate Events on the 14th packet, the system generates an eleventh event to indicate the change in action.

Policy-Wide Rate-Based Detection and Thresholding or Suppression Example

The following example shows an attacker attempting denial of service (DoS) attacks on hosts in your network. Many simultaneous connections to hosts from the same sources trigger a policy-wide Control Simultaneous Connections setting. The setting generates events and drops malicious traffic when there are five connections from one source in 10 seconds. In addition, a global limit threshold limits the number of events any rule or setting can generate to 10 events in 20 seconds.

As shown in the diagram, the policy-wide setting generates events for the first ten matching packets and drops the traffic. After the tenth packet, the limit threshold is reached, so for the remaining packets no events are generated but the packets are dropped.

After the timeout, note that packets are still dropped in the rate-based sampling period that follows. If the sampled rate is above the threshold rate in the current or previous sampling period, the rate-based action of generating events and dropping traffic continues. The rate-based action stops only after a sampling period completes where the sampled rate is below the threshold rate.



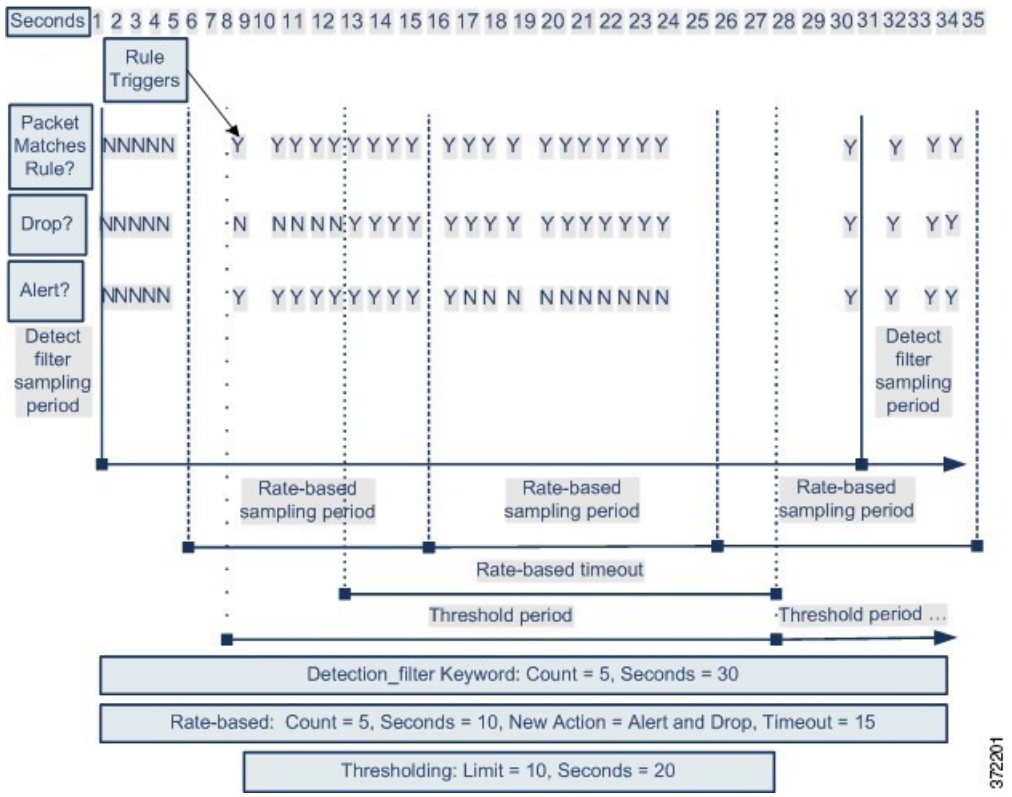
Note that although it is not shown in this example, if a new action triggers because of rate-based criteria *after* a threshold has been reached, the system generates a single event to indicate the change in action. So, for example, if the limit threshold of 10 has been reached and the system stops generating events and the action changes to Drop and Generate events on the 14th packet, the system generates an eleventh event to indicate the change in action.

Rate-Based Detection with Multiple Filtering Methods Example

The following example shows an attacker attempting a brute force login, and describes a case where a `detection_filter` keyword, rate-based filtering, and thresholding interact. Repeated attempts to find a password trigger a rule which includes the `detection_filter` keyword, with a count set to 5. This rule also has rate-based attack prevention settings that change the rule attribute to Drop and Generate Events for 30 seconds when there are five rule hits in 15 seconds. In addition, a limit threshold limits the rule to 10 events in 30 seconds.

As shown in the diagram, the first five packets matching the rule do not cause event notification because the rule does not trigger until the rate indicated in the `detection_filter` keyword is exceeded. After the rule triggers, event notification begins, but the rate-based criteria do not trigger the new action of Drop and Generate Events until five more packets pass. After the rate-based criteria are met, the system generates events for packets 11-15 and drops the packets. After the fifteenth packet, the limit threshold has been reached, so for the remaining packets the system does not generate events but does drop the packets.

After the rate-based timeout, note that packets are still dropped in the rate-based sampling period that follows. Because the sampled rate is above the threshold rate in the previous sampling period, the new action continues.



Rate-Based Attack Prevention Options and Configuration

Rate-based attack prevention identifies abnormal traffic patterns and attempts to minimize the impact of that traffic on legitimate requests. Rate-based attacks usually have one of the following characteristics:

- Any traffic containing excessive incomplete connections to hosts on the network, indicating a SYN flood attack
- Any traffic containing excessive complete connections to hosts on the network, indicating a TCP/IP connection flood attack
- Excessive rule matches in traffic going to a particular destination IP address or addresses or coming from a particular source IP address or addresses
- Excessive matches for a particular rule across all traffic

In a network analysis policy, you can either configure SYN flood or TCP/IP connection flood detection for the entire policy; in an intrusion policy, you can set rate-based filters for individual intrusion or preprocessor rules. Note that you cannot manually add a rate-based filter to GID 135 rules or modify their rule state. Rules with GID 135 use the client as the source value and the server as the destination value.

When **SYN Attack Prevention** is enabled, rule 135:1 triggers if a defined rate condition is exceeded.

When **Control Simultaneous Connections** is enabled, rule 135:2 triggers if a defined rate condition is exceeded, and rule 135:3 triggers if a session closes or times out.

Note Devices load-balance inspection across internal resources. When you configure rate-based attack prevention, you configure the triggering rate per resource, not per device. If rate-based attack prevention is not working as expected, you may need to lower the triggering rate. It triggers alert, if users send too many connection attempts within prescribed time intervals. Hence it is recommended to rate limit the rule. For help determining the correct rate, contact Support.

Each rate-based filter contains several components:

- For policy-wide or rule-based source or destination settings, the network address designation
- The rule matching rate, which you configure as a count of rule matches within a specific number of seconds
- A new action to be taken when the rate is exceeded

  When you set a rate-based setting for the entire policy, the system generates events when it detects a rate-based attack, and can drop the traffic in an inline deployment. When setting rate-based actions for individual rules, you have three available actions: Generate Events, Drop and Generate Events, and Disable.
- The duration of the action, which you configure as a timeout value

Note that when started, the new action occurs until the timeout is reached, even if the rate falls below the configured rate during that time period. When the timeout period expires, if the rate has fallen below the threshold, the action for the rule reverts to the action initially configured for the rule. For policy-wide settings, the action reverts to the action of each rule the traffic matches or stops if it does not match any rules.

You can configure rate-based attack prevention in an inline deployment to block attacks, either temporarily or permanently. Without rate-based configuration, rules set to Generate Events create events, but the system does not drop packets for those rules. However, if the attack traffic matches rules that have rate-based criteria configured, the rate action may cause packet dropping to occur for the period of time that the rate action is active, even if those rules are not initially set to Drop and Generate Events.

Note Rate-based actions cannot enable disabled rules or drop traffic that matches disabled rules. However, if you set a rate-based filter at the policy level, you can generate events on or generate events on and drop traffic that contains an excessive number of SYN packets or SYN/ACK interactions within a designated time period.

You can define multiple rate-based filters on the same rule. The first filter listed in the intrusion policy has the highest priority. Note that when two rate-based filter actions conflict, the system implements the action of the first rate-based filter. Similarly, policy-wide rate-based filters override rate-based filters set on individual rules if the filters conflict.

Rate-Based Attack Prevention, Detection Filtering, and Thresholding or Suppression

The `detection_filter` keyword prevents a rule from triggering until a threshold number of rule matches occur within a specified time. When a rule includes the `detection_filter` keyword, the system tracks the number of incoming packets matching the pattern in the rule per timeout period. The system can count hits for that rule from particular source or destination IP addresses. After the rate exceeds the rate in the rule, event notification for that rule begins.

You can use thresholding and suppression to reduce excessive events by limiting the number of event notifications for a rule, a source, or destination, or by suppressing notifications altogether for that rule. You can also configure a global rule threshold that applies to each rule that does not have an overriding specific threshold.

If you apply suppression to a rule, the system suppresses event notifications for that rule for all applicable IP addresses even if a rate-based action change occurs because of a policy-wide or rule-specific rate-based setting.

Configuring Rate-Based Attack Prevention

Note This section applies to Snort 2 preprocessors. For information on Snort 3 inspectors, see https://www.cisco.com/go/snort3-inspectors.

You can configure rate-based attack prevention at the policy level to stop SYN flood attacks. You can also stop excessive connections from a specific source or to a specific destination.

Procedure

Step 1 Choose **Policies** > **Access Control heading** > **Access Control**, and then click **Network Analysis Policy** or **Policies** > **Access Control heading** > **Intrusion**, and then click **Network Analysis Policies**.

Note
If your custom user role limits access to the first path listed here, use the second path to access the policy.

Step 2 Click **Snort 2 Version** next to the policy you want to edit.

Step 3 Click **Edit** (✎) next to the policy you want to edit.

If **View** (👁) appears instead, the configuration belongs to an ancestor domain, or you do not have permission to modify the configuration.

Step 4 Click **Settings**.

Step 5 If **Rate-Based Attack Prevention** under **Specific Threat Detection** is disabled, click **Enabled**.

Step 6 Click **Edit** (✎) next to **Rate-Based Attack Prevention**.

Step 7 You have two choices:

- To prevent incomplete connections intended to flood a host, click **Add** under **SYN Attack Prevention**.
- To prevent excessive numbers of connections, click **Add** under **Control Simultaneous Connections**.

Step 8 Specify how you want to track traffic:

- To track all traffic from a specific source or range of sources, choose **Source** from the **Track By** drop-down list, and enter a single IP address or address block in the **Network** field.
- To track all traffic to a specific destination or range of destinations, choose **Destination** from the **Track By** drop-down list, and enter an IP address or address block in the **Network** field.

Note
- Do not enter the IP address 0.0.0.0/0 in the Network field to monitor all subnets or IPs. The system does not support this IP address (which is usually used to identify all subnets or IPs) for Rate Based Attack Prevention.

- The system tracks traffic separately for each IP address included in the **Network** field. Traffic from an IP address that exceeds the configured rate results in generated events only for that IP address. As an example, you might set a source CIDR block of `10.1.0.0/16` for the network setting and configure the system to generate events when there are ten simultaneous connections open. If eight connections are open from 10.1.4.21 and six from 10.1.5.10, the system does not generate events, because neither source has the triggering number of connections open. However, if eleven simultaneous connections are open from 10.1.4.21, the system generates events only for the connections from 10.1.4.21.

Step 9 Specify the triggering rate for the rate tracking setting:

- For SYN attack configuration, enter the number of SYN packets per number of seconds in the **Rate** fields.
- For simultaneous connection configuration, enter the number of connections in the **Count** field.

Devices load-balance inspection across internal resources. When you configure rate-based attack prevention, you configure the triggering rate per resource, not per device. If rate-based attack prevention is not working as expected, you may need to lower the triggering rate. It triggers alert, if users send too many connection attempts within prescribed time intervals. Hence it is recommended to rate limit the rule. For help determining the correct rate, contact Support.

Step 10 To drop packets matching the rate-based attack prevention settings, check the **Drop** check box.

Step 11 In the **Timeout** field, enter the time period after which to stop generating events (and if applicable, dropping) for traffic with the matching pattern of SYNs or simultaneous connections.

Caution
Setting a high timeout value may entirely block connection to a host in an inline deployment.

Step 12 Click **OK**.

Step 13 To save changes you made in this policy since the last policy commit, click **Policy Information**, then click **Commit Changes**.

If you leave the policy without committing changes, changes since the last commit are discarded if you edit a different policy.

What to do next

- Deploy configuration changes.