Intrusion Prevention System with 5.x Format Signatures Configuration Example

Document ID: 105625

Contents

Introduction

This document describes how to configure 5.x format signatures in Cisco IOS® IPS and is organized into two sections:

- Section I. Getting Started Configuration Steps This section provides the steps necessary to use the Cisco IOS command−line interface (CLI) in order to get started with IOS IPS 5.x format signatures. This section describes these steps:

  Step 1. Download the IOS IPS files.

  Step 2. Create an IOS IPS configuration directory on Flash.

  Step 3. Configure an IOS IPS crypto key.

  Step 4. Enable IOS IPS.

  Step 5. Load the IOS IPS signature package to the router.

  Each step and specific commands are described in detail, as well as additional commands and references. An example configuration is displayed below each command.
- Section II. Advanced Configuration Options This section provides instructions and examples on advanced options for signature tuning. It contains these options:

  Retire or Unretire Signatures

  Enable or Disable Signatures

Change Signature Actions

Prerequisites

Requirements

Ensure you have the proper components (as described in Components Used) before you complete the steps in this document.

Components Used

The information in this document is based on these software and hardware versions:

- A Cisco Integrated Services Router (87x, 18xx, 28xx, or 38xx)
- 128MB or more DRAM and at least 2MB free flash memory
- Console or telnet connectivity to the router
- Cisco IOS Release 12.4(15)T3 or later
- A valid CCO (Cisco.com) login user name and password
- A current Cisco IPS Service Contract for licensed signature update services

The information in this document was created from the devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. If your network is live, make sure that you understand the potential impact of any command.

Conventions

Refer to Cisco Technical Tips Conventions for more information on document conventions.

Section I. Getting Started Configuration Steps

Step 1. Download IOS IPS Files

The first step is to download IOS IPS signature package files and public crypto key from Cisco.com.

Download the required signature files from Cisco.com to your PC:

- Location: http://www.cisco.com/pcgi−bin/tablebuild.pl/ios−v5sigup (registered customers only)
- Files to download:
    - IOS−Sxxx−CLI.pkg (registered customers only) This is the latest signature package.
    - realm−cisco.pub.key.txt (registered customers only) This is the public crypto key used by IOS IPS.

Step 2. Create an IOS IPS Configuration Directory on Flash

The second step is to create a directory on your router's flash where you store the required signature files and configurations. Alternatively, you can use a Cisco USB flash drive connected to the router's USB port to store the signature files and configurations. The USB flash drive must remain connected to the router's USB port if it is used as the IOS IPS configuration directory location. IOS IPS also supports any IOS file system as its configuration location with proper write access.

In order to create a directory, enter this command at the router prompt: **mkdir** <*directory name*>

For example:

```
router#mkdir ips
Create directory filename [ips]?
Created dir flash:ips
```

Additional Commands and References

In order to verify the contents of the flash, enter this command at the router prompt: **show flash:**

For example:

```
router#dir flash:
Directory of flash:/
   5 -rw-   51054864 Feb 8 2008 15:46:14 -08:00
                     c2800nm-advipservicesk9-mz.124-15.T3.bin
   6 drw-       0 Feb 14 2008 11:36:36 -08:00 ips
64016384 bytes total (12693504 bytes free)
```

In order to rename the directory name, use this command: **rename** <*current name*> <*new name*>

For example:

```
router#rename ips ips_new
Destination filename [ips_new]?
```

Step 3. Configure an IOS IPS Crypto Key

The third step is to configure the crypto key used by IOS IPS. This key is located in the realm-cisco.pub.key.txt file that was downloaded in Step 1.

The crypto key is used to verify the digital signature for the master signature file (sigdef-default.xml) whose contents are signed by a Cisco private key to guarantee its authenticity and integrity at every release.

1. Open the text file, and copy the contents of the file.
2. Use the **configure terminal** command in order to enter router configure mode.
3. Paste the text file content at the `<hostname>(config)#` prompt.
4. Exit router configuration mode.
5. Enter the **show run** command at the router prompt in order to confirm that the crypto key is configured. You should see this output in the configuration:

```
crypto key pubkey-chain rsa
named-key realm-cisco.pub signature
key-string
30820122 300D0609 2A864886 F70D0101 01050003 82010F00 3082010A 02820101
00C19E93 A8AF124A D6CC7A24 5097A975 206BE3A2 06FBA13F 6F12CB5B 4E441F16
17E630D5 C02AC252 912BE27F 37FDD9C8 11FC7AF7 DCDD81D9 43CDABC3 6007D128
B199ABCB D34ED0F9 085FADC1 359C189E F30AF10A C0EFB624 7E0764BF 3E53053E
5B2146A9 D7A5EDE3 0298AF03 DED7A5B8 9479039D 20F30663 9AC64B93 C0112A35
FE3F0C87 89BCB7BB 994AE74C FA9E481D F65875D6 85EAF974 6D9CC8E3 F0B08B85
50437722 FFBE85B9 5E4189FF CC189CB9 69C46F9C A84DFBA5 7A0AF99E AD768C36
006CF498 079F88F8 A3B3FB1F 9FB7B3CB 5539E1D1 9693CCBB 551F78D2 892356AE
2F56D826 8918EF3C 80CA4F4D 87BFCA3B BFF668E9 689782A5 CF31CB6E B4B094D3
F3020301 0001
Quit
```

6. Use this command in order to save the configuration:

   copy running-configure startup-configure

If the key is configured incorrectly, you must remove the crypto key first and then reconfigure it:

1. In order to remove the key, enter these commands in the order listed below:

```
router#configure terminal
router(config)#no crypto key pubkey-chain rsa
router(config-pubkey-chain)#no named-key realm-cisco.pub signature
router(config-pubkey-chain)#exit
router(config)#exit
```

2. Use the **show run** command in order to verify that the key is removed from the configuration.
3. Complete the procedure in Step 3 in order to reconfigure the key.

Step 4. Enable IOS IPS

The fourth step is to configure IOS IPS. Complete this procedure in order to configure IOS IPS:

1. Use the **ip ips name** <*rule name*> < *optional ACL*> command in order to create a rule name. (This will be used on an interface to enable IPS.)

   For example:

```
router#configure terminal
router(config)#ip ips name iosips
```

   You can specify an optional extended or standard access control list (ACL) in order to filter the traffic that will be scanned by this rule name. All traffic that is permitted by the ACL is subject to inspection by the IPS. Traffic that is denied by the ACL is not inspected by the IPS.

```
router(config)#ip ips name ips list ?
<1-199> Numbered access list
WORD Named access list
```

2. Use the **ip ips config location flash:**<*directory name*> command in order to configure IPS signature storage location. (This is the *ips* directory created in Step 2.)

   For example:

```
router(config)#ip ips config location flash:ips
```

3. Use the **ip ips notify sdee** command in order to enable IPS SDEE event notification.

   For example:

```
router(config)#ip ips notify sdee
```

   In order to use SDEE, the HTTP server must be enabled (with the **ip http server** command). If the HTTP server is not enabled, the router cannot respond to the SDEE clients because it cannot see the requests. SDEE notification is disabled by default and must be explicitly enabled.

   IOS IPS also supports the use of syslog in order to send event notification. SDEE and syslog can be used independently or enabled at the same time in order to send IOS IPS event notification. Syslog notification is enabled by default. If logging console is enabled, you will see IPS syslog messages. In order to enable syslog, use this command:

```
router(config)#ip ips notify log
```

4. Configure IOS IPS to use one of the predefined signature categories.

IOS IPS with Cisco 5.x format signatures operates with signature categories (just like Cisco IPS appliances). All signatures are grouped into categories, and the categories are hierarchical. This helps classify signatures for easy grouping and tuning.

Warning: The *all* signature category contains all signatures in a signature release. Since IOS IPS cannot compile and use all the signatures contained in a signature release at one time, *do not unretire the all category*; otherwise, the router will run out of memory.

Note: When you configure IOS IPS, you must first retire all the signatures in the *all* category, and then unretire selected signature categories.

Note: The order in which the signature categories are configured on the router is also important. IOS IPS processes the category commands in the order listed in the configuration. Some signatures belong to multiple categories. If multiple categories are configured and a signature belongs to more than one of them, the signature's properties (for example, retired, unretired, actions, etc.) in the last configured category are used by IOS IPS.

In this example, all the signatures in the "all" category are retired, and then the *IOS IPS Basic* category is unretired.

```
router(config)#ip ips signature-category
router(config-ips-category)#category all
router(config-ips-category-action)#retired true
router(config-ips-category-action)#exit
router(config-ips-category)#category ios_ips basic
router(config-ips-category-action)#retired false
router(config-ips-category-action)#exit
router(config-ips-category)#exit
Do you want to accept these changes? [confirm]y
router(config)#
```

5. Use these commands in order to enable IPS rule on the desired interface, and specify the direction in which the rule will be applied:

interface <*interface name*>

ip ips <*rule name*> **[***in* | *out***]**

For example:

```
router(config)#interface GigabitEthernet 0/1
router(config-if)#ip ips iosips in
router(config-if)#exit
router(config)#exit
router#
```

The *in* argument means only traffic going into the interface is inspected by IPS. The *out* argument means only traffic going out of the interface is inspected by IPS.

In order to enable IPS to inspect both in and out traffic of the interface, enter separately the IPS rule name for *in* and *out* on the same interface:

```
router(config)#interface GigabitEthernet 0/1
router(config-if)#ip ips iosips in
router(config-if)#ip ips iosips out
router(config-if)#exit
router(config)#exit
router#
```

Step 5. Load the IOS IPS Signature Package to the Router

The last step is to load to the router the signature package downloaded in Step 1.

Note: The most common way to load the signature package to the router is to use either FTP or TFTP. This procedure uses FTP. Please refer to the *Additional Commands and References* section in this procedure for an alternative method to load the IOS IPS signature package. If you use a telnet session, use the **terminal monitor** command in order to view the console outputs.

In order to load the signature package to the router, complete these steps:

1. Use this command in order to copy the downloaded signature package from the FTP server to the router:

   copy ftp://<*ftp_user:password@Server_IP_address*>/<*signature_package*> **idconf**

   Note: Please remember to use the *idconf* parameter at the end of the **copy** command.

   Note: For example:

   ```
   router#copy ftp://cisco:cisco@10.1.1.1/IOS-S310-CLI.pkg idconf
   Loading IOS-S310-CLI.pkg !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
   [OK - 7608873/4096 bytes]
   ```

   Signature compiling begins immediately after the signature package is loaded to the router. You can see the logs on the router with logging level 6 or above enabled.

   ```
   *Feb 14 16:44:47 PST: %IPS-6-ENGINE_BUILDS_STARTED: 16:44:47 PST Feb 14 2008
   *Feb 14 16:44:47 PST: %IPS-6-ENGINE_BUILDING: multi-string - 8 signatures -
                         1 of 13 engines
   *Feb 14 16:44:47 PST: %IPS-6-ENGINE_READY: multi-string - build time 4 ms -
                         packets for this engine will be scanned
   *Feb 14 16:44:47 PST: %IPS-6-ENGINE_BUILDING: service-http - 622 signatures -
                         2 of 13 engines
   *Feb 14 16:44:53 PST: %IPS-6-ENGINE_READY: service-http - build time 6024 ms -
                         packets for this engine will be scanned
   |
   output snipped
   |
   *Feb 14 16:45:18 PST: %IPS-6-ENGINE_BUILDING: service-smb-advanced - 35 signatures -
                         12 of 13 engines
   *Feb 14 16:45:18 PST: %IPS-6-ENGINE_READY: service-smb-advanced - build time 16 ms -
                         packets for this engine will be scanned
   *Feb 14 16:45:18 PST: %IPS-6-ENGINE_BUILDING: service-msrpc - 25 signatures -
                         13 of 13 engines
   *Feb 14 16:45:18 PST: %IPS-6-ENGINE_READY: service-msrpc - build time 32 ms -
                         packets for this engine will be scanned
   *Feb 14 16:45:18 PST: %IPS-6-ALL_ENGINE_BUILDS_COMPLETE: elapsed time 31628 ms
   ```

2. Use the **show ip ips signature count** command in order to verify the signature package is properly compiled.

   For example:

   ```
   router#show ip ips signature count
   Cisco SDF release version S310.0   signature package release version
   Trend SDF release version V0.0
   Signature Micro-Engine: multi-string: Total Signatures 8
   multi-string enabled signatures: 8
   multi-string retired signatures: 8
   |
   ```

```
outpt snipped
|
Signature Micro-Engine: service-msrpc: Total Signatures 25
service-msrpc enabled signatures: 25
service-msrpc retired signatures: 18
service-msrpc compiled signatures: 1
service-msrpc inactive signatures - invalid params: 6
Total Signatures: 2136
Total Enabled Signatures: 807
Total Retired Signatures: 1779
Total Compiled Signatures:
      351   total compiled signatures for the IOS IPS Basic category
Total Signatures with invalid parameters: 6
Total Obsoleted Signatures: 11
router#
```

Additional Commands and References

The public crypto key is invalid if you receive an error message at the time of signature compilation similar to this error message:

```
%IPS-3-INVALID_DIGITAL_SIGNATURE: Invalid Digital Signature found (key not found)
```

Refer to Step 3 for more information.

If you do not have access to an FTP or TFTP server, you can use a USB flash drive in order to load signature package to the router. First, copy the signature package onto the USB drive, connect the USB drive to one of the USB ports on the router, and then use the **copy** command with the *idconf* parameter in order to copy the signature package to the router.

For example:

```
router#copy usbflash1:IOS-S310-CLI.pkg idconf
```

There are six files in the configured IOS IPS storage directory. These files use this name format: <*router-name*>-sigdef-xxx.xml or <*router-name*>-seap-xxx.xml.

```
router#dir ips
Directory of flash:/ips/
7 -rw- 203419 Feb 14 2008 16:45:24 -08:00 router-sigdef-default.xml
8 -rw- 271 Feb 14 2008 16:43:36 -08:00 router-sigdef-delta.xml
9 -rw- 6159 Feb 14 2008 16:44:24 -08:00 router-sigdef-typedef.xml
10 -rw- 22873 Feb 14 2008 16:44:26 -08:00 router-sigdef-category.xml
11 -rw- 257 Feb 14 2008 16:43:36 -08:00 router-seap-delta.xml
12 -rw- 491 Feb 14 2008 16:43:36 -08:00 router-seap-typedef.xml
64016384 bytes total (12693504 bytes free)
router#
```

These files are stored in compressed format and are not directly editable or viewable. The contents of each file are described below:

- *router-sigdef-default.xml* contains all the factory default signature definitions.
- *router-sigdef-delta.xml* contains signature definitions that have been changed from the default.
- *router-sigdef-typedef.xml* contains all the signature parameter definitions.
- *router-sigdef-category.xml* contains the signature category information, such as category ios_ips basic and advanced.
- *router-seap-delta.xml* contains changes made to the default SEAP parameters.
- *router-seap-typedef.xml* contains all the SEAP parameter definitions.

Section II. Advanced Configuration Options

This section provides instructions and examples on advanced IOS IPS options for signature tuning.

Retire or Unretire signatures

To retire or unretire a signature means to select or deselect the signatures that are used by IOS IPS in order to scan traffic.

- **Retiring** a signature means IOS IPS will *NOT* compile that signature into memory for scanning.
- **Unretiring** a signature instructs IOS IPS to compile the signature into memory and use the signature to scan traffic.

You can use IOS command−line interface (CLI) in order to retire or unretire individual signatures or a group of signatures that belong to a signature category. When you retire or unretire a group of signatures, all signatures in that category are retired or unretired.

Note: Some unretired signatures (either unretired as individual signature or within an unretired category) may not compile due to insufficient memory or invalid parameters or if the signature has been obsoleted.

This example shows how to retire individual signatures. For example, signature 6130 with subsig ID of 10:

```
router#configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
router(config)#ip ips signature−definition
router(config−sigdef)#signature 6130 10
router(config−sigdef−sig)#status
router(config−sigdef−sig−status)#retired true
router(config−sigdef−sig−status)#exit
router(config−sigdef−sig)#exit
router(config−sigdef)#exit
Do you want to accept these changes? [confirm]y
router(config)#
```

This example shows how to unretire all signatures that belong to the IOS IPS Basic category:

```
router#configure terminal
Enter configuration commands, one per line. End with CNTL/Z
router(config)#ip ips signature−category
router(config−ips−category)#category ios_ips basic
router(config−ips−category−action)#retired false
router(config−ips−category−action)#exit
router(config−ips−category)#exit
Do you want to accept these changes? [confirm]y
```

Note: When signatures in categories other than IOS IPS Basic and IOS IPS Advanced are unretired as a category, compilation of some signatures or engines could fail because certain signatures in those categories are not supported by IOS IPS (see example below). All other successfully compiled (unretired) signatures are used by IOS IPS to scan traffic.

```
Router(config)#ip ips signature−category
router(config−ips−category)#category os
router(config−ips−category−action)#retired false
router(config−ips−category−action)#exit
router(config−ips−category)#exit
Do you want to accept these changes? [confirm]y
*Feb 14 18:10:46 PST: Applying Category configuration to signatures ...
*Feb 14 18:10:49 PST: %IPS−6−ENGINE_BUILDS_STARTED: 08:10:49 PST Feb 18 2008
```

```
*Feb 14 18:10:49 PST: %IPS-6-ENGINE_BUILDING: multi-string - 8 signatures -
                      1 of 13 engines
*Feb 14 18:10:49 PST: %IPS-6-ENGINE_READY: multi-string - build time 136 ms -
                      packets for this engine will be scanned
*Feb 14 18:10:49 PST: %IPS-6-ENGINE_BUILDING: service-http - 622 signatures -
                      2 of 13 engines
*Feb 14 18:10:50 PST: %IPS-4-META_ENGINE_UNSUPPORTED: service-http 5903:1 -
                      this signature is a component of the unsupported META engine
*Feb 14 18:24:42 PST: %IPS-4-SIGNATURE_COMPILE_FAILURE: service-http 5754:0 -
                      compilation of regular expression failed
*Feb 14 18:24:49 PST: %IPS-4-SIGNATURE_COMPILE_FAILURE: service-http 5729:1 -
                      compilation of regular expression failed
```

Enable or Disable Signatures

To enable or disable a signature is to enforce or disregard the action(s) associated with the signatures by IOS IPS when packet or packet flow matches the signatures.

Note: Enable and disable does NOT select and deselect signatures to be used by IOS IPS.

- To **Enable** a signature means that when triggered by a matching packet (or packet flow), the signature takes the appropriate action associated with it. However, only unretired AND successfully compiled signatures will take the action when they are enabled. In other words, if a signature is retired, even though it is enabled, it will not be compiled (because it is retired) and it will not take the action associated with it.
- To **Disable** a signature means that when triggered by a matching packet (or packet flow), the signature DOES NOT take the appropriate action associated with it. In other words, when a signature is disabled, even though it is unretired and successfully compiled, it will not take the action associated with it.

You can use IOS command-line interface (CLI) in order to enable or disable individual signatures or a group of signatures based on signature categories. This example shows how to disable signature 6130 with subsig ID of 10.

```
router#configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
router(config)#ip ips signature-definition
router(config-sigdef)#signature 6130 10
router(config-sigdef-sig)#status
router(config-sigdef-sig-status)#enabled false
router(config-sigdef-sig-status)#exit
router(config-sigdef-sig)#exit
router(config-sigdef)#exit
Do you want to accept these changes? [confirm]y
router(config)#
```

This example shows how to enable all signatures that belong to the IOS IPS Basic category.

```
router#configure terminal
Enter configuration commands, one per line. End with CNTL/Z
router(config)#ip ips signature-category
router(config-ips-category)#category ios_ips basic
router(config-ips-category-action)#enabled true
router(config-ips-category-action)#exit
router(config-ips-category)#exit
Do you want to accept these changes? [confirm]y
router(config)#
```

Change Signature Actions

You can use IOS command−line interface (CLI) in order to change signature actions for one signature or a group of signatures based on signature categories. This example shows how to change signature actions to alert, drop, and reset for signature 6130 with subsig ID of 10.

```
router#configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
router(config)#ip ips signature−definition
router(config−sigdef)#signature 6130 10
router(config−sigdef−sig)#engine
router(config−sigdef−sig−engine)#event−action produce−alert
router(config−sigdef−sig−engine)#event−action deny−packet−inline
router(config−sigdef−sig−engine)#event−action reset−tcp−connection
router(config−sigdef−sig−engine)#exit
router(config−sigdef−sig)#exit
router(config−sigdef)#exit
Do you want to accept these changes? [confirm]y
router(config)#
```

This example shows how to change event actions for all signatures that belong to the signature IOS IPS Basic category.

```
router#configure terminal
Enter configuration commands, one per line. End with CNTL/Z
router(config)#ip ips signature−category
router(config−ips−category)#category ios_ips basic
router(config−ips−category−action)#event−action produce−alert
router(config−ips−category−action)#event−action deny−packet−inline
router(config−ips−category−action)#event−action reset−tcp−connection
router(config−ips−category−action)#exit
router(config−ips−category)#exit
Do you want to accept these changes? [confirm]y
router(config)#
```

Related Information

- **Cisco IOS Intrusion Prevention System (IPS) Products & Services Page**
- **Cisco IOS IPS − Version 5 Signatures Software Download**
- **IPS 5.x Signature Format Support and Usability Enhancements**
- **Cisco Security Device Manager Software Download**
- **How to Use CCP to Configure IOS IPS**
- **Cisco Intrusion Detection System Event Viewer 3DES Cryptographic Software Download**
- **Technical Support & Documentation − Cisco Systems**

Contacts & Feedback | Help | Site Map
© 2013 − 2014 Cisco Systems, Inc. All rights reserved. Terms & Conditions | Privacy Statement | Cookie Policy | Trademarks of Cisco Systems, Inc.

Updated: May 17, 2008 Document ID: 105625