Tuning, Deploying and Updating Cisco IOS IPS Signature sets For Multiple-Device Deployments

Introduction

Cisco IOS IPS requires a specific sequence of actions to install a signature package on a staging router, tune and configure the signature set and distribute (copy) the resulting signature database files to a production router community. Furthermore, distribution of signature-package updates to the production-router community requires additional specific actions, particularly to assure continuity of site-specific signature set modifications.

This document describes these steps:

1. Configure IPS on a staging router with an arbitrary initial signature package (sig-1), deliver IPS database files from staging router to distribution server
2. Configure IOS IPS on production routers and install IPS database files
3. If necessary, modify configuration of a signature (sig-1) on production router.
4. Update sigs to a newer sig package (sig-2) on staging router
5. Modify configuration of signature (sig-2) on staging router.
6. Copy sig-2 IPS database files from staging router to distribution server
7. Download sig-2 IPS files to production router to staging directory on production router and compile sig-2 database into existing database
8. Verified that active sig count had changed, and modifications to sig-1 and sig-2 were both effective.

The signature packages described in the document are arbitrary, and offer no comment on the suitability of the specific packages to particular security attacks or vulnerabilities.

This document does not provide signature configuration guidance, or details on the steps needed to adjust the signatures. This information can be found in other documents indicated where appropriate.

The configuration discussion in this document is relevant to Cisco IOS Software version 12.4(11)T2 and later. Cisco IOS IPS deployments based on Cisco IOS Software versions prior to 12.4(11)T are NOT recommended, as signature updates for these software versions are no longer available.

1 Create signature database in staging environment

Create signature database files on a development router in an staging environment. Ideally, you should have a staging environment that fairly reasonably simulates your production environment, in order to test feature interaction, memory and performance requirements, and signature effectiveness for the vulnerabilities that you expect to address. To determine vulnerabilities addressed by signatures, review the signature descriptions and respective releases at: http://tools.cisco.com/security/center/search.x

1.1 Enable IOS IPS on staging router

Configure the staging router to enable IOS IPS. Cisco recommends use of Cisco Configuration Professional (CCP) version 1.1 or later to configure IPS and later update and tune signatures on the staging router. Please see http://www.cisco.com/en/US/prod/collateral/iosswrel/ps6537/ps6586/ps6634/prod_white_paper0900aecd8066d265.html for details on that task.

If you prefer to use CLI to configure Cisco IOS IPS on the staging router, please refer to: http://www.cisco.com/en/US/prod/collateral/iosswrel/ps6537/ps6586/ps6634/prod_white_paper0900aecd805c4ea8.html

Note: Cisco Security Device Manager (SDM) is not supported for IPS configuration on Cisco IOS Software version 12.4(15)T2 or later.

You can download an appropriate Cisco IOS IPS signature package files at: http://tools.cisco.com/support/downloads/go/Model.x?mdfid=281442967&mdfLevel=SoftwareFamily&treeName=Security&modelName=Cisco IOS Intrusion Prevention System Feature Software&treeMdfId=268438162

1.2 Tune signatures

After configuring Cisco IOS IPS on your router, you should adjust the staging router's signature configuration. The IOS IPS 'basic' and 'advanced' signature categories include a set of signatures that detect and mitigate a broad range of traffic that could potentially exploit various types of software vulnerabilities in server and workstation hosts, as well as network devices; other signature categories may hold some appeal for your environment, but will likely require tuning to fit into the router's available memory. You will need to be sure that IOS IPS is configured to enable signatures that specifically address the requirements of your environment. Additionally, while some signatures may offer some benefit for vulnerabilities that your network presents, IPS may recognize traffic that is not an exploit as unwanted traffic, thus affecting a "false positive". False positives must be dealt with in a manner that suits the nature of the vulnerability. If a very high-risk vulnerability must be mitigated by IPS, operational tools and staff must be able to distinguish between traffic that comprises a false positive and that which comprises a live exploit. Otherwise, signatures addressing low-risk vulnerabilities might be tuned to generate less response, or disabled entirely, to avoid the additional operational burden of dealing with the false positives. Other reasons for tuning the signature database are to reduce memory or CPU footprint, or to add custom IPS signatures that you have developed to address the security requirements of your environment.

Signature tuning is best performed with the Cisco Configuration Professional or Cisco Security Manager GUI tools. CLI can be used, although interacting with the CLI will likely present a more tedious task.

1.2.1 CCP Signature tuning is discussed in "Task 5" in the "Using CCP to configure IOS IPS in Cisco IOS 12.4(15)T4 and later releases" document, here: http://www.cisco.com/en/US/prod/collateral/iosswrel/ps6537/ps6586/ps6634/prod_white_paper0900aecd8066d265.html

1.2.2 CLI-based signature tuning for Cisco IOS IPS signatures is described in section II of http://www.cisco.com/en/US/prod/collateral/iosswrel/ps6537/ps6586/ps6634/prod_white_paper0900aecd805c4ea8.html with complete details shown at http://www.cisco.com/en/US/docs/ios/sec_data_plane/configuration/guide/sec_ips5_sig_fs_ue_ps6350_TSD_Products_Configuration_Guide_Chapter.html

Briefly, the following example describes disabling and retiring a signature which is enabled by the IOS 'Basic' category:

```
ip ips signature-definition
 signature 5733 0
 status
  enabled false
  retired true
end
```

1.3 Verify effectiveness of IPS configuration

After you have tuned your signature set that you will deploy, you will benefit from testing the signature configuration with penetration-testing ot traffic-generation tools to be certain that the signature configuration will be effective for your network environment, will not cause an excessive level of performance impact to your network's traffic, and will not degrade the traffic that you expect to carry on the network.

Many tools exist that provide the capability to test Intrusion Prevention. A commercially-available tool such as Mu Dynamics, Core Impact or ThreatEx' may be used, or low-cost open-source tools such as the Metasploit Framework can be applied.

1.4 IPS database file packaging and distribution

The signature loading and tuning populates the IOS IPS signature database, which is contained in the router's IPS configuration directory as four '.xml' or '.xmz' files that represent the signatures. These files describe the signatures, which categories they belong to, their retirement and enabled/disabled settings, and fidelity value:

On routers running IOS Releases prior to 15.0M Release:

- routername-sigdef-category.xml
- routername-sigdef-default.xml
- routername-sigdef-typedef.xml
- routername-sigdef-delta.xml

On routers running 15.0M/15,1T or later IOS Releases:

- iosips-sigdef-category.xmz
- iosips-sigdef-default.xmz
- iosips-sigdef-typedef.xmz
- iosips-sigdef-delta.xmz

The ".xmz" file extension has replaced the ".xml" extension in those releases due to IPS signature update license enforcement and indicates that the file contents are compressed. However, the purpose and function of the file is exactly the same regardless of the extension.

Additionally, the signature database holds two additional files that describe the SEAP configuration, in the event that you have adjusted the Signature Event Action Override values. If you do not modify the SEAP configuration on the staging router, you do not need to distribute these files:

On routers running IOS Releases prior to 15.0M Release:

- routername-seap-typedef.xml
- routername-seap-delta.xml

On routers running 15.0M/15,1T or later IOS Releases:

- iosips-seap-typedef.xmz
- iosips-seap-delta.xmz

If you modify the Signature Event Action Override configuration and wish to apply these changes on all production routers, you will need to distribute the SEAP configuration files with the other four IPS database files. Configuration dialogue examples in this document will not describe the additional steps necessary for distributing these files to production routers.

On routers running IOS Releases prior to 15.0M Release, the filenames include the router name, as well as a description of which component of the IPS signature database that they comprise. The router name is of little

significance, and you may wish to replace the router name in the filename with other details, such as which signature package is included, the date of database compilation, or a serial number of some significance to your operational model. This document will describe the filenames by replacing 'routername' with the signature package number used for the signature database compilation. Also, only examples using the file name format with *.xml extension are given in this document but the same steps also apply to routers running 15.0M/15,1T or later IOS Releases with the exception of slightly different file name format and extension (*.xmz) used as mentioned above.

Upload the appropriate signature database files to a host from which you will distribute the Signature database files. The name of the router where the signature database files were generated will be included in the signature database filenames. You may wish to rename the files to include the base signature package file that was used to generate the signature database, and other pertinent information such as date and memory requirements. Cisco IOS IPS database files (.xml or .xmz files) should be held on a distribution location that is supported by the Cisco IOS File System (e.g. TFTP, FTP, HTTP, SCP, etc), as described in this document:
http://www.cisco.com/en/US/docs/ios/12_0/configfun/configuration/guide/fcifs.html

The files may be bundled into a tarball to reduce the number of files that will be distributed to production routers. However, bundling files into a tarball will incur additional installation steps on production routers to unpack the tarball. If you decide to distribute the files in a tarball, you will need to archive the files into a tarball on an external host, by using the appropriate software utility to build the tarball. Cisco IOS can extract files from a tar archive, but cannot build a tar archive.

You will need to position a distribution server offering one or more of the above-listed services on your network that is accessible from production routers. It might be a good idea to put the signature-file server in the same network as the rest of your IPS operational environment.

2 IOS IPS configuration on production routers

This section describes a typical router IPS configuration, but applies the signature database files developed on the staging router. Since a Cisco signature package file will not be loaded on a router, the router will not need to be loaded with the realm-cisco key to decrypt and authenticate the Cisco signature package file. If you will download the plain signature files (as opposed to a sig-file tarball), proceed to step 2.2. If production routers will download a tarball of the signature files, follow step 2.1.

2.1 Download signatures on production router

The signatures can be downloaded directly to the router's IPS database from the staging server if they are distributed in their native format ("...xml" or "...xmz"). If you wish to follow this step, skip to step 2.2. If the files are distributed as a .tar file, the tarball must be extracted to an intermediate directory on the router, as follows. The following example describes the tarball named "S391-ios-ips_050509.tar":

2.1.1 Create a temporary "update" repository :

```
#mkdir flash:ips-update

Create directory filename [ips-update]?

Created dir flash: ips-update
```

2.1.2 Check ips signature status :

```
#show ip ips all
```

2.1.3 Create a temporary "update" repository :

- tar file download on the router :

  The tar file format is downloaded in ips-update repository

```
#copy ftp://10.11.12.13/S391-ios-ips_050509.tar ips-update
```

- tar file extraction :

  ".xml" files are extracted within ips-update repository. This example only illustrates the signature configuration files with .xml extension, and does not include the SEAP configuration:

```
# archive tar /xtract ips-update/S391-ios-ips_050509.tar /ips-update
extracting S391-sigdef-category.xml (23018 bytes)
extracting S391-sigdef-default.xml (208307 bytes)
extracting S391-sigdef-delta.xml (255 bytes)
extracting S391-sigdef-typedef.xml (6159 bytes)
[OK - 245760 bytes]
```

2.2 Configure and enable IPS. This step may be automated for large-scale router-based IPS deployments, by applying Cisco Configuration Engine or the operational support tool of your choice, which will allow the bulk download of IOS IPS signatures and configuration of IOS IPS on each production router, without tedious manual effort. The following description is included to illustrate the steps needed on each router.

Details of Cisco Configuration Engine are available at:http://www.cisco.com/en/US/products/sw/netmgtsw/ps4617/

2.2.1 Create IPS directory on Flash

Create a directory on the production router's flash where the IOS IPS signature files will be stored:

```
#mkdir flash:/iosips
Create directory filename [iosips]?
Created dir flash:iosips
```

2.2.2 Configure IPS and apply to an interface as appropriate for your router configuration:

```
(config)# ip ips name ios-ips
(config)#ip ips config location flash:/iosips
 (config)#ip ips signature-category
(config-ips-category)# category all
(config-ips-category-action)# retired true
(config-ips-category-action)# exit
(config)#interface FastEthernet 0/1
(config-if)#ip ips ios-ips in
(config-if)#ip ips ios-ips out
```

2.3 Verify initial configuration and apply signature files from staging server (or from intermediate directory, in case of tarball distribution in 2.1)

- Signature download and compilation :

  ".xml" files files are copied from distribution server and compiled, and resulted signatures are stored in the router's database directory as described by the configuration (in this case, flash:/iosips). This example copies the files from a TFTP distribution server, (e.g. 172.16.1.20). The example describes the CLI command, but not the IPS signature database compilation dialogue. You will need to apply the commands one at a time, and wait for each file's compilation before issuing the next command:

```
3845-l#copy tftp://172.16.1.20/S391-sigdef-typedef.xml idconf
```

```
3845-l#copy tftp://172.16.1.20/S391-sigdef-category.xml idconf
3845-l#copy tftp://172.16.1.20/S391-sigdef-default.xml idconf
3845-l#copy tftp://172.16.1.20/S391-sigdef-delta.xml idconf
```

- Signature compilation from local repository created from tarball:

  ".xml" files located in ips-update repository are compiled, and resulted signatures are stored in the router's database directory as described by the configuration (in this case, flash:/iosips). The example describes the CLI command, but not the IPS signature database compilation dialogue. You will need to apply the commands one at a time, and wait for each file's compilation before issuing the next command:

```
3845-l#copy flash:/ips-update/S391-sigdef-typedef.xml idconf
3845-l#copy flash:/ips-update/S391-sigdef-category.xml idconf
3845-l#copy flash:/ips-update/S391-sigdef-default.xml idconf
3845-l#copy flash:/ips-update/S391-sigdef-delta.xml idconf
```

2.4 Remove intermediate directory from sig-file download

This step is only needed if you downloaded the signature files to the router as a tarball. If you downloaded individual signature files (*.xml or *.xmz), you do not need to execute this step.

Remove the "ips-update" directory:

```
#delete flash:ips-update/* Delete filename [ips_update/*]?
Delete flash: ips-update/S391-sigdef-category.xml? [confirm]
Delete flash: ips-update/S391-sigdef-default.xml? [confirm]
Delete flash: ips-update/S391-sigdef-delta.xml? [confirm]
Delete flash: ips-update/S391-sigdef-typedef.xml? [confirm]
#rmdir ips-update
Remove directory filename [ips-update]?
Delete flash: ips-update? [confirm]
Removed dir flash:ips-update
```

3 Apply site-specific signature configuration tuning

Site-specific configuration is generally discouraged, as it incurs a substantial operation burden to maintain records of IPS configuration tuning specific to each site. However, some sites may need specific considerations for local application and network requirements that are not addressed well by organization-wide security policy and signature configuration. For these cases, Cisco Security Manager (CSM) offers the capability to deploy site-specific signature configuration, or CCP or CLI (or your preferred operational management system) can be used to adjust signature configuration based on individual sites' requirements.

This section observes the same procedures as those in the signature tuning that was applied on the staging router, but this section only applies changes at sites where specific signatures must be modified to accommodate local network activity that would otherwise cause false positives. Refer to documents described in 1.2.1 or 1.2.2 for, respectively, CCP-based or CLI-based signature configuration.

4 Update signature files on staging router

This section follows similar steps to those in the initial creation of the signature files, but this section merges an updated signature package into the existing signature database on the staging router. Thus, prior adjustments to the signature file are maintained, and new/updated sigs are added. Initial IOS IPS configuration is not required, as this was defined in 1.2.

When you compile a new signature package on a router that carries an existing signature database, the signature configuration in the new signature package will supersede the router's existing database's signature configuration. Thus, if you have made changes to the signature database on the staging router, and you compile in an updated signature package that contradicts your changes, your changes will be overwritten, and will need to be re-created. You can avoid having to re-create your changes if you copy the "routername-sigdef-delta.xml" or "iosips-sigdef-delta.xmz" file to some other location on the router's local storage (or a network storage location, if desired), and re-apply the original "routername-sigdef-delta.xml" or "iosips-sigdef-delta.xmz" to the updated signature database after you have compiled the updated signature package to the router's database.

The following steps are effectively identical to those described in Part Two, above, and are not described in detail.

4.1 If you have made any changes to the signature configuration on the staging router, and wish to maintain the changes in your global signature database, you should back up the flash:/iosips/*routername*-sigdef-delta.xml file (or "iosips-sigdef-delta.xmz" file), as you will need to re-apply it after updating the signature update.

The following steps illustrate the creation of a directory named 'ips-delta-backup', and copying the sigdef-default file. If you have modified the seap configuration, you should back up the seap-delta file, as well:

```
3845-l#mkdir ips-delta-backup
Create directory filename [ips-delta-backup]?
Created dir flash:ips-delta-backup
3845-l#dir iosips
Directory of flash:/iosips/
  5 -rw-   276237 Jun 22 2009 15:38:14 +00:00 3845-l-sigdef-default.xml
  6 -rw-      392 Jun 22 2009 16:01:08 +00:00 3845-l-sigdef-delta.xml
  7 -rw-     8509 Jun 22 2009 15:37:16 +00:00 3845-l-sigdef-typedef.xml
  8 -rw-    33966 Jun 22 2009 15:37:18 +00:00 3845-l-sigdef-category.xml
  9 -rw-      257 Jun 22 2009 15:30:42 +00:00 3845-l-seap-delta.xml
  10 -rw-     491 Jun 22 2009 15:30:42 +00:00 3845-l-seap-typedef.xml
127832064 bytes total (75780096 bytes free)
3845-l#copy iosips/3845-l-sigdef-delta.xml flash:/ips-delta-backup
Destination filename [/ips-delta-backup/3845-l-sigdef-delta.xml]?
Copy in progress...C
392 bytes copied in 0.124 secs (3161 bytes/sec)
```

4.2 Download updated signature package from Cisco.com

Download a newer signature package (IOS-Sxxx-CLI.pkg) than the sig package used in the initial configuration from the Cisco.com distribution location: http://www.cisco.com/cgi-bin/tablebuild.pl/ios-v5sigup

Additionally, you can subscribe to the Cisco IDS/IPS Signature Update Bulletin to be notified of updates in Cisco IPS signature packages, here: http://tools.cisco.com/gdrp/coiga/showsurvey.do?surveyCode=380&keyCode=123668_4

4.3 Compile updated signature package into IOS IPS database

This portion of the configuration is described for use with command-line interaction for the simplicity and clarity offered therein.

4.3.1 Check the signature configuration to determine initial number of signatures. The command 'show ip ips signature count' is the most effective command for determining the number of signatures enabled or retired for each signature micro-engine:

```
3845-l#sh ip ips sig count


Cisco SDF release version S391.0
Trend SDF release version V0.0
Signature Micro-Engine: multi-string: Total Signatures 13
   multi-string enabled signatures: 4
   multi-string retired signatures: 12
   multi-string compiled signatures: 1
Signature Micro-Engine: service-http: Total Signatures 674
   service-http enabled signatures: 58
   service-http retired signatures: 673
   service-http compiled signatures: 1
   service-http obsoleted signatures: 2
Signature Micro-Engine: string-tcp: Total Signatures 1298
   string-tcp enabled signatures: 425
   string-tcp retired signatures: 1282
   string-tcp compiled signatures: 16
   string-tcp obsoleted signatures: 21
Signature Micro-Engine: string-udp: Total Signatures 78
   string-udp enabled signatures: 8
   string-udp retired signatures: 72
   string-udp compiled signatures: 6
   string-udp obsoleted signatures: 1
Signature Micro-Engine: state: Total Signatures 33
   state enabled signatures: 7
   state retired signatures: 33
Signature Micro-Engine: atomic-ip: Total Signatures 317
```

```
   atomic-ip enabled signatures: 60
   atomic-ip retired signatures: 316
   atomic-ip compiled signatures: 1
Signature Micro-Engine: string-icmp: Total Signatures 3
   string-icmp enabled signatures: 0
   string-icmp retired signatures: 3
Signature Micro-Engine: service-ftp: Total Signatures 3
   service-ftp enabled signatures: 0
   service-ftp retired signatures: 3
Signature Micro-Engine: service-rpc: Total Signatures 75
   service-rpc enabled signatures: 18
   service-rpc retired signatures: 75
Signature Micro-Engine: service-dns: Total Signatures 39
   service-dns enabled signatures: 0
   service-dns retired signatures: 39
   service-dns obsoleted signatures: 1
Signature Micro-Engine: normalizer: Total Signatures 9
   normalizer enabled signatures: 0
   normalizer retired signatures: 9
Signature Micro-Engine: service-smb-advanced: Total Signatures 49
   service-smb-advanced enabled signatures: 25
   service-smb-advanced retired signatures: 49
Signature Micro-Engine: service-msrpc: Total Signatures 30
   service-msrpc enabled signatures: 22
   service-msrpc retired signatures: 30
   service-msrpc obsoleted signatures: 1
Total Signatures: 2621
  Total Enabled Signatures: 627
  Total Retired Signatures: 2596
  Total Compiled Signatures: 25
  Total Obsoleted Signatures: 26
```

4.3.2 On the staging router's Command-Line Interface, copy the new signature package to idconf. This will trigger a signature re-compilation:

```
3845-l(config-subif)#do copy tftp://172.16.1.20/IOS-S405-CLI.pkg idconf
Loading IOS-S405-CLI.pkg from 172.16.1.20 (via GigabitEthernet0/0.109): !!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!
[OK - 9069730 bytes]
```

```
*Jun 22 16:50:00.403: %IPS-6-ENGINE_BUILDS_STARTED: 16:50:00 UTC Jun 22 2009
*Jun 22 16:50:00.407: %IPS-6-ENGINE_BUILDING: multi-string - 13 signatures - 1 o
f 13 engines
*Jun 22 16:50:00.415: %IPS-6-ENGINE_READY: multi-string - build time 8 ms - pack
ets for this engine will be scanned
*Jun 22 16:50:00.467: %IPS-6-ENGINE_BUILDING: service-http - 700 signatures - 2
of 13 engines
*Jun 22 16:50:00.599: %IPS-6-ENGINE_READY: service-http - build time 132 ms - pa
ckets for this engine will be scanned
*Jun 22 16:50:00.759: %IPS-6-ENGINE_BUILDING: string-tcp - 1509 signatures - 3 o
f 13 engines
*Jun 22 16:50:01.279: %IPS-6-ENGINE_READY: string-tcp - build time 520 ms - pack
ets for this engine will be scanned
*Jun 22 16:50:01.411: %IPS-6-ENGINE_BUILDING: string-udp - 78 signatures - 4 of
13 engines
*Jun 22 16:50:01.419: %IPS-6-ENGINE_READY: string-udp - build time 8 ms - packet
s for this engine will be scanned
*Jun 22 16:50:01.431: %IPS-6-ENGINE_BUILDING: state - 33 signatures - 5 of 13 en
gines
*Jun 22 16:50:01.435: %IPS-6-ENGINE_READY: state - build time 4 ms - packets for
 this engine will be scanned
*Jun 22 16:50:01.499: %IPS-6-ENGINE_BUILDING: atomic-ip - 338 signatures - 6 of
13 engines
*Jun 22 16:50:01.795: %IPS-6-ENGINE_READY: atomic-ip - build time 296 ms - packe
ts for this engine will be scanned
*Jun 22 16:50:01.859: %IPS-6-ENGINE_BUILDING: string-icmp - 3 signatures - 7 of
13 engines
*Jun 22 16:50:01.859: %IPS-6-ENGINE_READY: string-icmp - build time 0 ms - packe
ts for this engine will be scanned
*Jun 22 16:50:01.859: %IPS-6-ENGINE_BUILDING: service-ftp - 3 signatures - 8 of
13 engines
*Jun 22 16:50:01.859: %IPS-6-ENGINE_READY: service-ftp - build time 0 ms - packe
ts for this engine will be scanned
*Jun 22 16:50:01.863: %IPS-6-ENGINE_BUILDING: service-rpc - 76 signatures - 9 of
 13 engines
*Jun 22 16:50:01.875: %IPS-6-ENGINE_READY: service-rpc - build time 12 ms - pack
ets for this engine will be scanned
```

```
*Jun 22 16:50:01.887: %IPS-6-ENGINE_BUILDING: service-dns - 39 signatures - 10 o
f 13 engines
*Jun 22 16:50:01.891: %IPS-6-ENGINE_READY: service-dns - build time 4 ms - packe
ts for this engine will be scanned
*Jun 22 16:50:01.895: %IPS-6-ENGINE_BUILDING: normalizer - 9 signatures - 11 of
13 engines
```

4.3.3 Re-check the signature configuration to verify that new signatures have been added to the signature database. The number of enabled, retired, compiled, and obsoleted signatures should, most likely, change from one signature package version to the next:

```
3845-l#sh ip ips sign count
Cisco SDF release version S405.0
Trend SDF release version V0.0
Signature Micro-Engine: multi-string: Total Signatures 13
   multi-string enabled signatures: 4
   multi-string retired signatures: 12
   multi-string compiled signatures: 1
Signature Micro-Engine: service-http: Total Signatures 700
   service-http enabled signatures: 58
   service-http retired signatures: 699
   service-http compiled signatures: 1
   service-http obsoleted signatures: 2
Signature Micro-Engine: string-tcp: Total Signatures 1509
   string-tcp enabled signatures: 424
   string-tcp retired signatures: 1492
   string-tcp compiled signatures: 17
   string-tcp obsoleted signatures: 21
Signature Micro-Engine: string-udp: Total Signatures 78
   string-udp enabled signatures: 8
   string-udp retired signatures: 72
   string-udp compiled signatures: 6
   string-udp obsoleted signatures: 1
Signature Micro-Engine: state: Total Signatures 33
   state enabled signatures: 7
   state retired signatures: 33
Signature Micro-Engine: atomic-ip: Total Signatures 338
   atomic-ip enabled signatures: 60
   atomic-ip retired signatures: 337
   atomic-ip compiled signatures: 1
```

```
Signature Micro-Engine: string-icmp: Total Signatures 3
   string-icmp enabled signatures: 0
   string-icmp retired signatures: 3
Signature Micro-Engine: service-ftp: Total Signatures 3
   service-ftp enabled signatures: 0
   service-ftp retired signatures: 3
Signature Micro-Engine: service-rpc: Total Signatures 76
   service-rpc enabled signatures: 18
   service-rpc retired signatures: 76
Signature Micro-Engine: service-dns: Total Signatures 39
   service-dns enabled signatures: 0
   service-dns retired signatures: 39
   service-dns obsoleted signatures: 1
Signature Micro-Engine: normalizer: Total Signatures 9
   normalizer enabled signatures: 0
   normalizer retired signatures: 9
Signature Micro-Engine: service-smb-advanced: Total Signatures 49
   service-smb-advanced enabled signatures: 25
   service-smb-advanced retired signatures: 49
Signature Micro-Engine: service-msrpc: Total Signatures 31
   service-msrpc enabled signatures: 20
   service-msrpc retired signatures: 31
   service-msrpc obsoleted signatures: 1
Total Signatures: 2881
  Total Enabled Signatures: 624
  Total Retired Signatures: 2855
  Total Compiled Signatures: 26
  Total Obsoleted Signatures: 26
```

Verify that the 'Total Signatures' section does not include a line describing signature compilation failures:

```
Total Signatures: 2881
  Total Enabled Signatures: 1164
  Total Retired Signatures: 2324
  Total Compiled Signatures: 555
  Total Signatures with compile failures: 2
  Total Obsoleted Signatures: 26
```

If you notice that signature compilation failed for any signatures, you should closely re-check the console output during signature compilation to ascertain which signatures did not compile, as in the following example:

```
*Jun 22 16:24:56.999: %IPS-6-ENGINE_BUILDING: string-tcp - 1509 signatures - 3 o
f 13 engines
*Jun 22 16:25:14.103: %SYS-2-MALLOCFAIL: Memory allocation of 65536 bytes failed
 from 0x601028EC, alignment 0
Pool: Processor Free: 1542296 Cause: Memory fragmentation
Alternate Pool: None Free: 0 Cause: No Alternate pool
 -Process= "Exec", ipl= 0, pid= 108, -Traceback= 0x616BB230 0x600F06B4 0x600F65
7C 0x600F6B4C 0x601028F4 0x601020C0 0x60103BB8 0x60101D24 0x62530BF8 0x62531858
0x62531DFC 0x62531FAC 0x6316736C 0x631691F0 0x63174A08 0x63174E0C
*Jun 22 16:25:14.135: %IPS-4-SIGNATURE_COMPILE_FAILURE: string-tcp 5916:0 - fail
ed to compile regular expression
*Jun 22 16:25:15.607: %IPS-4-SIGNATURE_COMPILE_FAILURE: string-tcp 5477:0 - fail
ed to compile regular expression
*Jun 22 16:25:16.287: %IPS-6-ENGINE_READY: string-tcp - build time 19288 ms - pa
ckets for this engine will be scanned
```

If signature compilation fails, then there may be a shortage of available memory to allow compilation of all the signatures or signature categories that you have unretired and enabled, or there may be some other problem. Verify the available memory and your signature/signature category configuration, and make adjustments as necessary.

4.4 Copy saved *routername*-sigdef-delta.xml or “iosips-sigdef-delta.xmz” to idconf to restore your signature configuration changes that were overwritten by the signature package update.

```
3845-l#copy flash://ips-delta-backup/3845-l-sigdef-delta.xml idconf
3845-l#
*Jun 22 17:08:36.707: %IPS-6-ENGINE_BUILDS_STARTED: 17:08:36 UTC Jun 22 2009
*Jun 22 17:08:36.907: %IPS-6-ENGINE_BUILDING: string-tcp - 1509 signatures - 1 o
f 13 engines
*Jun 22 17:08:37.403: %IPS-6-ENGINE_READY: string-tcp - build time 496 ms - pack
ets for this engine will be scanned
*Jun 22 17:08:37.647: %IPS-6-ENGINE_BUILDING: atomic-ip - 338 signatures - 2 of
13 engines
*Jun 22 17:08:37.879: %IPS-6-ENGINE_READY: atomic-ip - build time 232 ms - packe
ts for this engine will be scanned
*Jun 22 17:08:37.947: %IPS-6-ALL_ENGINE_BUILDS_COMPLETE: elapsed time 1240 ms
3845-l#
```

4.5 Tune and verify signatures

As during the initial signature configuration, it is a good practice to tune (i.e., add/remove/ change actions) signatures to your specific environment, and verify that the tuned signature set on the router functions as expected.

4.6 IPS database file packaging and distribution

Again, copy the signature database files from the staging router's ips directory to a distribution staging server. If you merged a new signature package into an existing IPS configuration, only modified files need to be redistributed (usually only default.xml/default.xmz and delta.xml/delta.xmz files):

- routername-sigdef-category.xml or iosips-sigdef-category.xmz
- routername-sigdef-default.xml or iosips-sigdef-default.xmz
- routername-sigdef-typedef.xml or iosips-sigdef-typedef.xmz
- routername-sigdef-delta.xml or iosips-sigdef-delta.xmz

As in section 1.4, you only need to distribute the SEAP configuration if you modified any of the Signature Event Action Override configuration:

- routername-seap-typedef.xml or iosips-seap-typedef.xmz
- routername-seap-delta.xml or iosips-seap-delta.xmz

On routers running IOS Releases prior to 15.0M Release, rename the files to your preferred filenames. You should use a different filename than you used in Part One of this document, when you generated the initial signature database. This will simplify management of archived signature databases and reduce confusion in the event that you need to revert to an older signature database.

5 Distribute updated signature files to production routers

As mentioned above in the instructions for updating the signature database on the staging router, when you compile the updated signature database on a router that carries an existing signature database, the signature configuration in the database files that you add to the router will supersede the existing database's signature configuration. Thus, if you have made device-specific changes to the signature database on the router, and you compile in an updated signature database that contradicts your device-specific (local) changes, your local changes will be overwritten, and will need to be re-created. You can avoid having to re-create the local changes if you copy the "routername-sigdef-delta.xml" or "iosips-sigdef-delta.xmz file to some other location on the router's local storage (or a network storage location, if desired), and re-apply the original "*routername*-sigdef-delta.xml" or "iosips-sigdef-delta.xmz file to the updated signature database after you have compiled the updated signature database files from the staging router.

The following steps are effectively identical to those described in Part Two, above, and are not described in detail.

5.1 If any routers have site-specific configuration, you should back up the flash:/iosips/*routername*-sigdef-delta.xml file (or "iosips-sigdef-delta.xmz" file), as you will need to re-apply it after updating the signature update.

5.2 Check signature configuration before update

This step will typically not be desired for every production router, but should be observed when the signature update is tested on one or more production router sites

5.3 Download updated IPS database files to the router if the file is distributed as a tarball, and extract the tarball to a temporary folder on the router.

5.4 Copy signature database files from temporary folder (or files in temp folder resulting from tarball extraction) to idconf to compile new sigs into local database.

5.5 Copy saved *routername*-sigdef-delta.xml or "iosips-sigdef-delta.xmz" file to idconf to restore local changes that were overwritten by signature database update.

5.6 Verify IPS signature configuration and status

5.7 Remove temporary directory after copying the files in the temp directory to ips

Americas Headquarters
Cisco Systems, Inc.
San Jose, CA

Asia Pacific Headquarters
Cisco Systems (USA) Pte. Ltd.
Singapore

Europe Headquarters
Cisco Systems International BV
Amsterdam, The Netherlands

Cisco has more than 200 offices worldwide. Addresses, phone numbers, and fax numbers are listed on the Cisco Website at www.cisco.com/go/offices.

Printed in USA C11-549300-01 02/10