Detecting IPv6 Tunnels in an Enterprise Network

Introduction

The ongoing depletion of unique and global IPv4 addresses is creating an increased focus on IPv6 technology. The regional registries run ongoing statistics on the available IPv4 address pool and realize that they will be unable to meet IPv4 address block requests at some point in the future (http://www.potaroo.net/tools/ipv4/index.html). Most major computer operating system vendors support IPv6 and have it enabled by default. They will even try to use IPv6 through dynamic tunneling technologies if the enterprise network is capable of supporting IPv4 only.

An important consideration is that IPv6 is quite likely to be already running on the enterprise network, whether that implementation was planned or not. Some important characteristics of IPv6 include:

- IPv6 has a mechanism to automatically assign addresses so that end systems can easily establish communications.
- IPv6 has several mechanisms available to ease the integration of the protocol into the network.
- Automatic tunneling mechanisms can take advantage of the underlying IPv4 network and connect it to the IPv6 Internet.

For an IPv4 enterprise network, the existence of an IPv6 overlay network has several of implications:

- The IPv4 firewalls can be bypassed by the IPv6 traffic, and leave the security door wide open.
- Intrusion detection mechanisms not expecting IPv6 traffic may be confused and allow intrusionIn some cases (for example, with the IPv6 transition technology known as 6to4), an internal PC can communicate directly with another internal PC and evade all intrusion protection and detection systems (IPS/IDS). Botnet command and control channels are known to use these kind of tunnels.

For all these reasons, it is recommended that enterprise IT departments do their utmost to detect overlay networks and generally create awareness of their existence. Using a managed implementation strategy, IT can help build a plan to control the integration of IPv6 into the network.

Network Tools for IPv6 Intelligence Enterprise IT organizations can use existing network tools to gather information about IPv6 activity in the network. Some of the existing network infrastructure tools that can be used include: Netflow, ACL (access control lists) (on routers and firewalls) and special rules on intrusion detection/prevention systems. The usage of these tools can detect the most frequently used tunnels; however it cannot detect all of them (i.e. IPv6 tunneling over https used by Windows 7).

Netflow

Netflow is a technology used to collect information about the data flows in a network. These flows carry a set of tuples of information (for example the source and destination IP address, the used source and destination port numbers and the protocol carried). As a direct result, Netflow can also be used to identify IPv6 traffic within a network. It can identify traffic that is being automatically tunneled and it can identify native IPv6 traffic. In the manual configuration below, the interface has been set up to capture flow information that is leaving the interface.

```
!
interface GigabitEthernet0/1
 ip address 192.4.1.1 255.255.255.0
 ip flow egress
!
```

Once this feature is enabled, one can look into the captured information. To view this information on the router, run the “show ip cache flow” command. The example below shows an example of the output from this command

```
Router#show ip cache flow
IP packet size distribution (112297 total packets):
  1-32  64  96 128 160 192 224 256 288 320 352 384 416 448 480
  .000 .790 .209 .000 .000 .000 .000 .000 .000 .000 .000 .000 .000 .000

  512 544 576 1024 1536 2048 2560 3072 3584 4096 4608
  .000 .000 .000 .000 .000 .000 .000 .000 .000 .000 .000

IP Flow Switching Cache, 278544 bytes
 4094 active, 2 inactive, 46879 added
 96780 ager polls, 0 flow alloc failures
 Active flows timeout in 30 minutes
 Inactive flows timeout in 15 seconds
IP Sub Flow Cache, 34056 bytes
 0 active, 1024 inactive, 0 added, 0 added to flow
 0 alloc failures, 0 force free
 1 chunk, 0 chunks added
 last clearing of statistics 00:00:02
Protocol     Total  Flows  Packets Bytes Packets Active(Sec) Idle(Sec)
--------      Flows   /Sec   /Flow  /Pkt   /Sec    /Flow     /Flow
TCP-other   23440  2344.0     1   60  2356.8    0.0      0.2
UDP-other   23426  2342.6     2   48  6566.3    0.0      0.2
IPv6INIP     31   3.1      1025 82  3177.5    0.1      0.0
Total:       46897  4689.7     2   68  12100.6    0.0      0.2

SrcIf     SrcIPaddress  DstIf     DstIPaddress  Pr SrcP DstP Pkts
Gi0/0     192.3.1.174   Gi0/1*    192.3.9.1    11 281A 0050   1
Gi0/0     192.3.1.163   Gi0/1*    192.3.73.1   11 2812 0050   1
Gi0/0     192.3.1.170   Gi0/1*    192.3.41.1   11 281A 0050   1
Gi0/0     192.3.1.147   Gi0/1*    192.3.41.1   11 2826 0050   1
```

There is a lot of information that can be gleaned from this output. The top table identifies the packet size distribution of the flows that have been captured. Then there’s an indication of the overall flow activity and a summary table of the flows going through the configured interface. The key information in the flow summary table is the line that starts with “IPv6INIP”. This entry means that IPv6 is being tunneled in the network using one of the automatic tunneling mechanisms. The last part of the output shows actual flow information.

Because there might be a lot of flow information to sift through, the output can be piped with the help of a filter to only include certain flows. In the example below, only the output lines that include “29” in them will be displayed. “29” was used because it is the hexadecimal representation of the decimal number “41”. “41” is the protocol number assigned to be used for automatic IPv6 in IPv4 tunneling mechanisms such as 6to4 (RFC3056) or ISATAP (RFC5214).

```
Router#show ip cache flow | incl 29
Gi0/0     192.3.1.129   Gi0/1*    192.3.64.1   11 282F 0050   1
Gi0/0     192.3.1.23   Gi0/1*    192.3.32.1   11 2829 0050   1
Gi0/0     192.3.1.2    Gi0/1*    192.168.1.1   29 0000 0000  32
Gi0/0     192.3.1.229   Gi0/1*    192.3.69.1   06 2B34 01BB   1
Gi0/0     192.3.1.248   Gi0/1*    192.3.69.1   06 2B29 01BB   1
```

This information can be used to track down the end systems that are using these mechanisms to connect to the IPv6 Internet.

Another popular automatic tunneling mechanism is Teredo (RFC4380). Teredo encapsulates IPv6 in IPv4 but it uses UDP to accomplish this functionality. A side impact of having IPv6 through Teredo is that it can bypass NAT and firewall devices. The default UDP port number used by Teredo is 3544. The example below shows the netflow cache filtered to only show lines that have DD8 in the output (DD8 in hex = 3544 in decimal).

```
Router#show ip cache flow | incl DD8
Gi0/0     192.3.1.2    Gi0/1*    192.168.1.91  11 2710 0DD8  136
```

This information can be used to track down the end systems that are using this mechanism to connect to the IPv6 Internet. A caveat here is that end users can change the default port number for Teredo traffic, and as result it becomes much harder to identify the Teredo tunnels.

IPv6 traffic can flow on a network even though it is not enabled on the network infrastructure. The existing Cisco switching infrastructure can be used to observe that traffic and track those hosts down. In this example, a Catatlyst 6500 is used to identify IPv6 traffic. The switch itself is not configured for IPv6. It is acting simply as a L2 switch and forwarding IPv6 frames between hosts in a single VLAN. To identify the traffic the following configuration commands are used:

```
mls flow ipv6 full
ip flow ingress layer2-switched vlan X
```

With these commands enabled, IPv6 traffic can be observed that is possibly traversing the switch. In the output below you can see several IPv6 flows. This output along with the MAC address table can be used to identify hosts in the network that are using IPv6.

```
Switch6500 #show mls netflow ipv6
Displaying Netflow entries in Active Supervisor EARL in module 1
DstIP                 SrcIP
-------------------------------------------------------------------------------
Prot:SrcPort:DstPort  Src i/f     :AdjPtr
Pkts    Bytes    Age LastSeen  Attributes
-------------------------------------------------------------------------------
-------
FF02::1                FE80::21F:CAFF:FE0E:8270
icmp:136  :0     Vl5       :0x0
1      72     172 14:06:28  Mcast(IPv6)-Dynamic
::                   ::
0  :0   :0     --        :0x0
2852047   193937146  1599 14:09:20  L3 (IPv6) - Dynamic
FF02::1:FF0E:8270           ::
icmp:135  :0     Vl5       :0x0
1      64     173 14:06:27  Mcast(IPv6)-Dynamic
FF02::1:FF63:3BF8           ::
icmp:135  :0     Vl5       :0x0
1      64     159 14:06:41  Mcast(IPv6)-Dynamic
FF02::1:FF63:3BF8           ::
icmp:135  :0     Vl5       :0x0
0      0      159 14:06:41  Mcast(IPv6)-Dynamic
FF02::1                FE80::223:33FF:FE63:3BF8
icmp:136  :0     Vl5       :0x0
1      72     158 14:06:42  Mcast(IPv6)-Dynamic
```

```
FF02::1                 FE80::223:33FF:FE63:3BF8
icmp:136  :0     Vl5       :0x0
0      0      158 14:06:42  Mcast(IPv6)-Dynamic
FF02::1:FF0E:8270            FE80::223:33FF:FE63:3BF8
icmp:135  :0     Vl5       :0x0
1      72     18  14:09:02  Mcast(IPv6)-Dynamic
FF02::1:FF0E:8270            FE80::223:33FF:FE63:3BF8
icmp:135  :0     Vl5       :0x0
0      0      18  14:09:02  Mcast(IPv6)-Dynamic
```

In the above example output it can be seen that there are two nodes in the VLAN sending IPv6 ICMPv6 packets. The IPv6 source addresses used are FE80::223:33FF:FE63:3BF8 and FE80::21F:CAFF:FE0E:8270. It is very common that nodes which run IPv6 use IPv6 Stateless Address Autoconfiguration (SLAAC) (RFC4862) to automatically number its IPv6 enabled interfaces. On a media as Ethernet the L2 MAC is used within this procedure. Hence, by extracting the last 64 bits of this address the MAC address can be identified rather easily.

For example lets have a look at FE80::223:33FF:FE63:3BF8. The last 64 bits in this address are **223:33**FF:FE**63:3BF8**. From these 64 bits the device 48 bit MAC address can be deducted as 02:23:33:63:3B:F8. Due to the SLAAC operation there is one additional consideration that is that it executes a logical OR function on the 7th bit of the MAC address. Taking that into account, the original MAC address could have been either 02:23:33:63:3B:F8 or 00:23:33:63:3B:F8.

Knowing these potential MAC addresses, one could investigate the L2 MAC address table on the switch and correlate the MAC address with a port number on the switch.

```
Switch6500#show mac-address-table vlan 5
Legend: * - primary entry
    age - seconds since last seen
    n/a - not available

 vlan  mac address   type  learn   age       ports
------+---------------+--------+-----+----------+--------------------------
Active Supervisor:
*  5 001f.ca0e.8270  dynamic Yes     0  Fa3/45
*  5 0023.3363.3bf8  dynamic Yes     0  Fa3/47
*  5 001d.7172.cb00  static No      -  Router
*  5 3333.0000.000d  static Yes     -  Fa3/1,Fa3/3,Fa3/33,Fa3/35
                          Fa3/45,Fa3/47,Router,Switch
*  5 0000.0900.0000  dynamic Yes     0  Fa3/1
*  5 3333.0000.0001  static Yes     -  Switch
*  5 3333.0000.0016  static Yes     -  Switch
```

Access Control Lists (ACL)

The primary use for ACLs is to permit or deny access to resources on the network. ACLs can also be used to gather information about what is happening on the network. The example below shows an interface that is configured to deny all IPv6 automatic tunneling mechanisms, deny the Teredo tunneling mechanism (or at least the default port for Teredo) and to permit everything else. In this example, the “log” keyword has been added so that the router captures the source and destination IP addresses and port numbers. Note that the “log” keyword has a severe impact on the device CPU is not needed to get an indication that IPv6 traffic is flowing in the network, but it is needed to capture specific information about the flows. An ACL like this can be applied specifically to find out if IPv6 is being used in a certain area of the network without having any significant impact to the network. Note that ‘permit’ is used in the

ACL. The motivation is that to detect tunnels should not have an impact on the existing traffic. The prescribed technique can be used on routers, switches and firewalls throughout the network infrastructure.

```
interface GigabitEthernet0/1
 ip address 192.4.1.1 255.255.255.0
 ip access-group DetectIPv6 out

ip access-list extended DetectIPv6
 permit 41 any any log
 permit udp any any eq 3544
 permit ip any any
```

The configuration for this filter on an ASA firewall looks as follows

```
access-list DetectIPv6 extended permit 41 any any
access-list DetectIPv6 extended permit udp any any eq 3544
access-list DetectIPv6 extended permit ip any any
```

Apply the access list to the inside interface of the ASA. The Internal interface is that interface pointing towards those PCs trying to tunnel out:

```
access-group DetectIPv6 in interface inside
```

To look at the statistics for the ACL, run the “show access-list” command. Note that if the “log” keyword was not in use the only information available would be the number of matches for each entry.

```
Router#show access-list DetectIPv6
Extended IP access list DetectIPv6
  10 permit 41 any any log (63110 matches)
  20 permit udp any any eq 3544 log (107813 matches)
  30 permit ip any any (2250006 matches)
```

Because the “log” keyword is used for each ACL entry, a syslog message is generated when a match occurs on an ACL entry.

```
Router# show log | incl IPACCESS
*May 14 18:44:25.822: %SEC-6-IPACCESSLOGNP: list DetectIPv6 allowed 41 192.3.1.2 ->
192.168.1.1, 93533 packets
*May 14 18:44:25.822: %SEC-6-IPACCESSLOGP: list DetectIPv6 allowed udp
192.3.1.2(10000) -> 192.168.1.91(3544), 159784 packets
```

From this output on a router, the devices that are attempting to use automatic IPv6 tunneling mechanisms can be tracked down.

The results of the “show access-list DetectIPv6” on the ASA firewall provides the following output:

```
ASA# show access-list DetectIPv6
access-list DetectIPv6; 3 elements; name hash: 0x544680dd
access-list DetectIPv6 line 1 extended permit 41 any any (hitcnt=8) 0x0d08f5ea
access-list DetectIPv6 line 2 extended permit udp any any eq 3544 (hitcnt=4)
0xe4bf6382
access-list DetectIPv6 line 3 extended permit ip any any (hitcnt=19730) 0x64eb8a3c
ASA#
```

Note, that with the ASA you actually do not necessarily need the “log” keyword on the deny statements. The Syslog messages will appear regardless. And, without the “log” statement they will be actually per-denied-packet. With low volume of the tunneled traffic it might be easier to use.

If you include the “log” it will use a slightly different mechanism, whereas it will try to memorize the “deny flow” and will act a bit different:

```
%ASA-6-106100: access-list DetectIPv6 allowed 41 inside/192.168.2.10(0) ->
outside/192.0.4.126(0) hit-cnt 1 first hit [0xd08f5ea, 0x0]
%ASA-6-106100: access-list DetectIPv6 allowed udp inside/192.168.2.10(1280) ->
outside/192.0.4.126(3544) hit-cnt 1 first hit [0xe4bf6382, 0x0]
%ASA-6-106100: access-list DetectIPv6 allowed udp inside/192.168.2.10(1281) ->
outside/192.0.4.126(3544) hit-cnt 1 first hit [0xe4bf6382, 0x0]
%ASA-6-106100: access-list DetectIPv6 allowed udp inside/192.168.2.10(1282) ->
outside/192.0.4.126(3544) hit-cnt 1 first hit [0xe4bf6382, 0x0]
%ASA-6-106100: access-list DetectIPv6 allowed udp inside/192.168.2.10(1283) ->
outside/192.0.4.126(3544) hit-cnt 1 first hit [0xe4bf6382, 0x0]
```

Note how it shows the “first hit”.

This may be nicer in higher-bandwidth requirements because it does not send as much syslog entries, however, there is a consequence that needs to be taken into account.

```
ASA(config)# access-list deny-flow-max ?

configure mode commands/options:
 <1-4096> Maximum number of concurrent deny flows that can be created,
      default is 4096
ASA(config)# access-list deny-flow-max
```

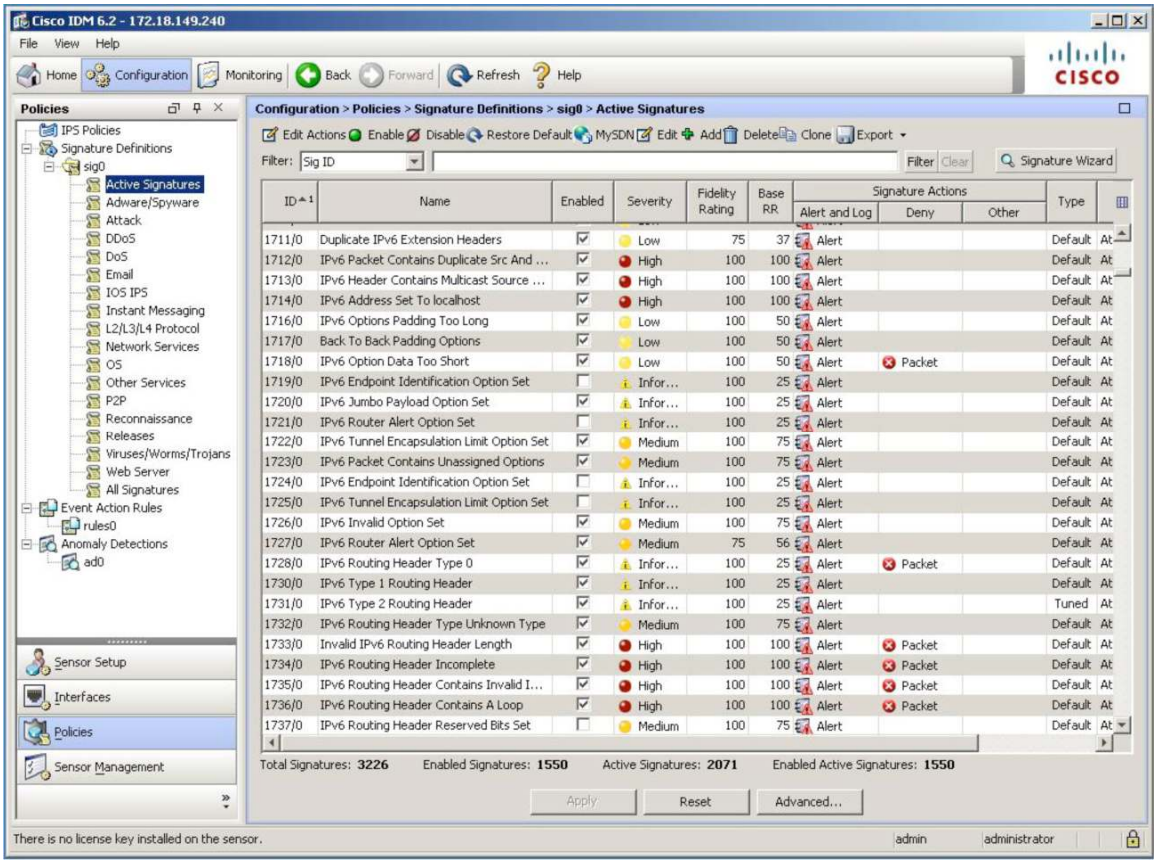
After 4096 flows, it falls back to “non-flow-based” mode to avoid overload.

Below is the output of the “show access-list” for this scenario:

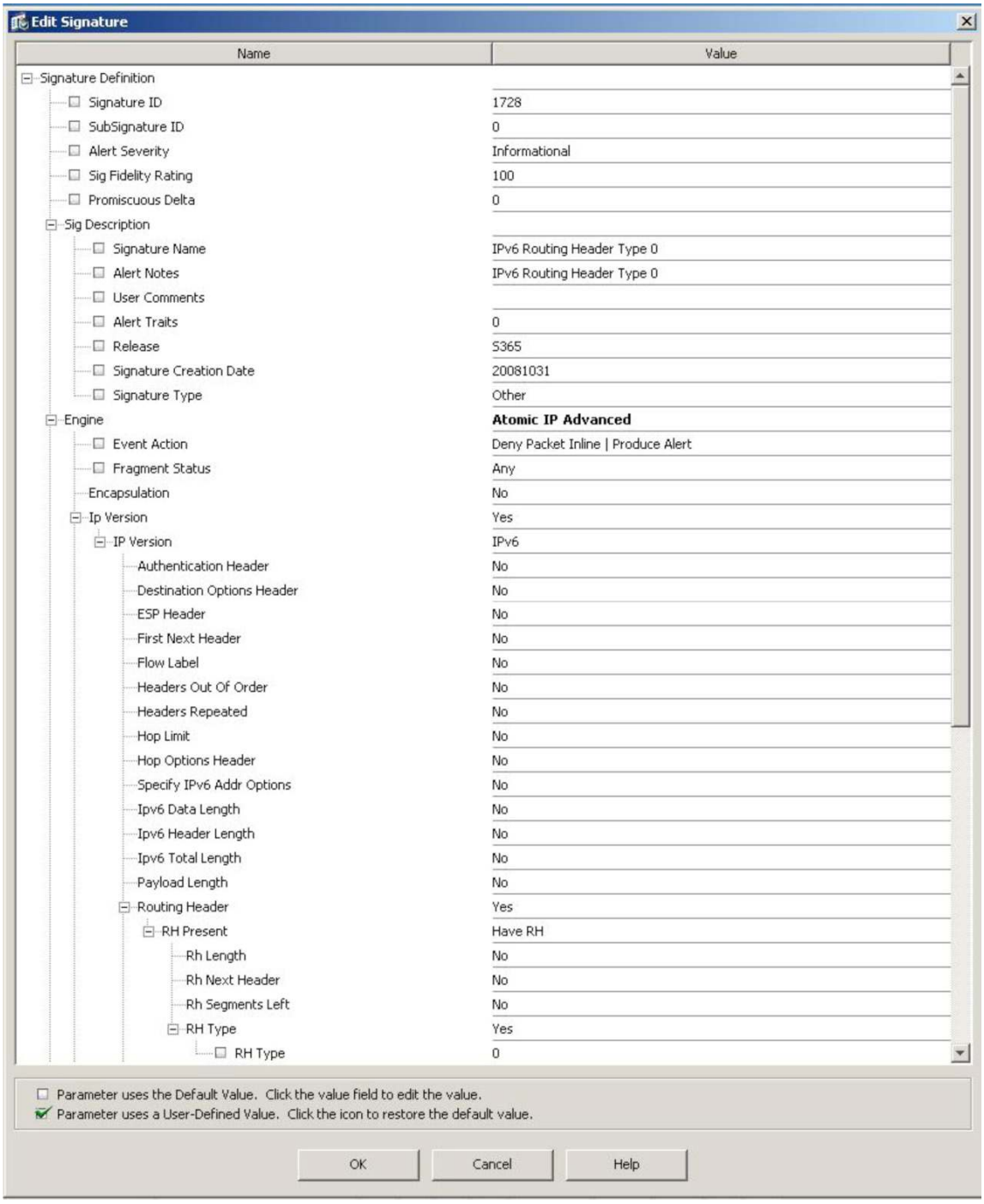
```
ASA(config)# show access-list DetectIPv6
access-list DetectIPv6; 3 elements; name hash: 0x544680dd
access-list DetectIPv6 line 1 extended permit 41 any any log informational interval
300 (hitcnt=185) 0x0d08f5ea
access-list DetectIPv6 line 2 extended permit udp any any eq 3544 log informational
interval 300 (hitcnt=12) 0xe4bf6382
```

Intrusion Detection/Prevention Systems

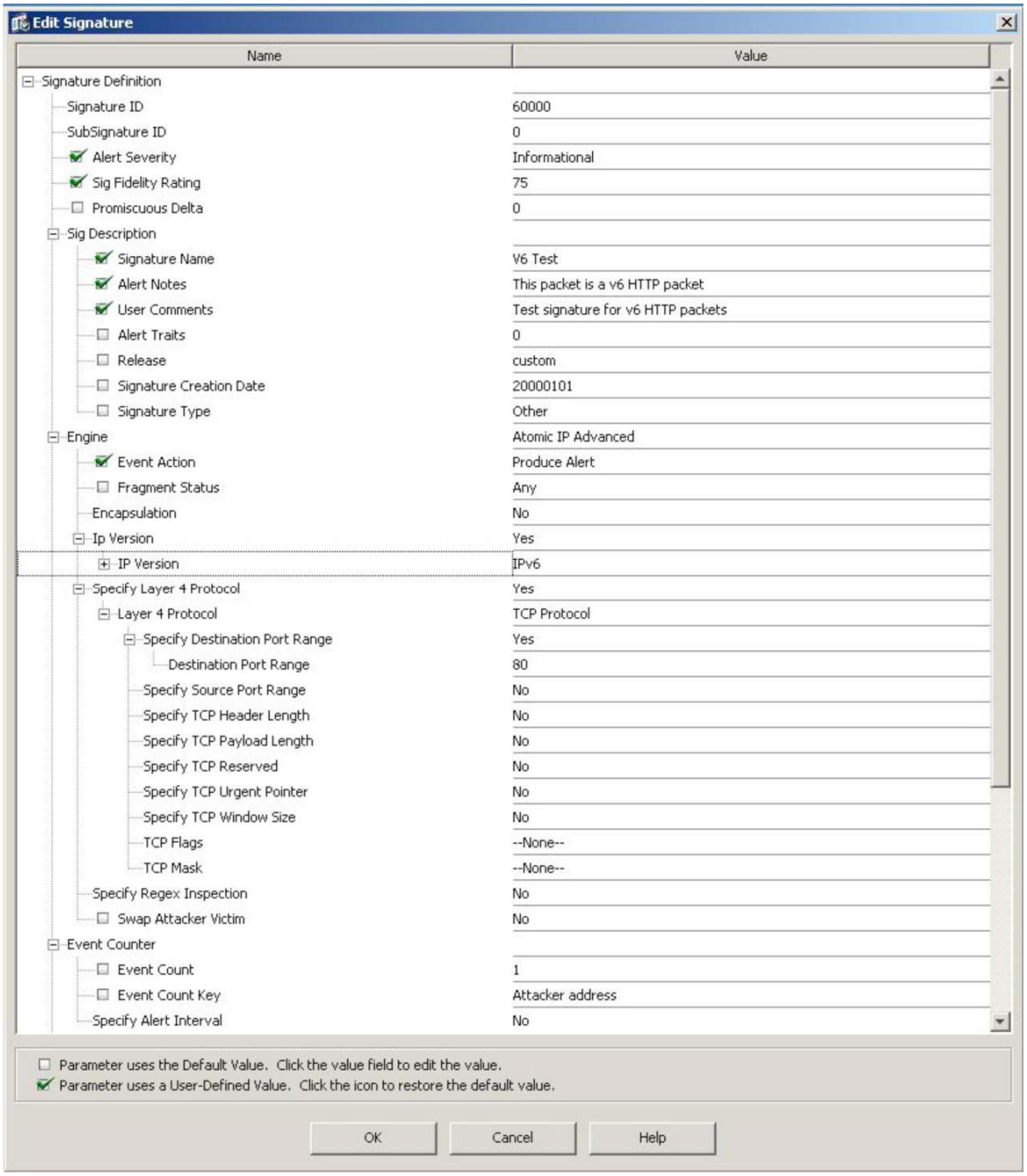
The existing intrusion detection and prevention systems (IDS/IPS) can also be used to identify IPv6 traffic that might be flowing in a network. Cisco IDS appliances that run software release 6.2 or later can be used to detect native IPv6 traffic and tunneled IPv6 traffic. The figure below shows a sample of some of the IPv6 signatures that are available in the 6.2 release.



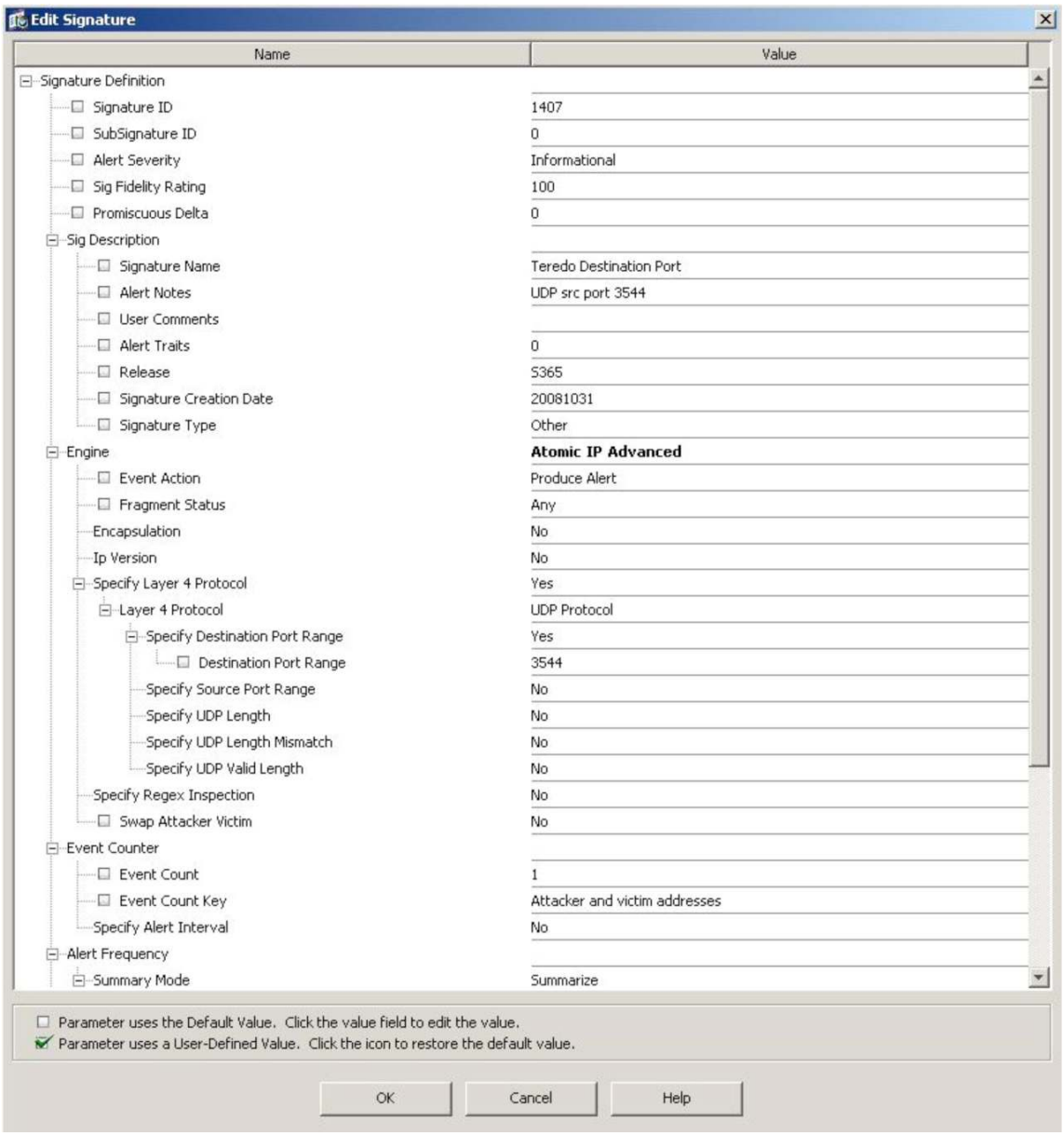
The figure below shows some details behind an IPv6 signature. In this case, the signature is identifying IPv6 packets that have a routing extension header type 0.



Custom IPv6 signatures can also be created. In the figure below a signature was created to send an alert when an IPv6 HTTP packet is detected.



The IDS/IPS running older versions of code can also be used to detect IPv6 tunneled traffic. By building custom signatures that detect protocol 41 and Teredo tunnels the IDS/IPS can be used to find this traffic in the network. The figure below shows a signature for recognizing Teredo tunneled traffic.



Tunnel Well-known addresses

Dynamic IPv6 tunnel mechanisms often use well-known addresses, and hence monitoring these parameters will also provide data-points on tunnel usage in the network.

By default a Windows 7 device will try to use ISATAP, which is an intra-site dynamic IPv6 tunnel mechanism. One of the steps a Windows 7 device will do in its efforts to enable IPv6, is a DNS request to isatap.MyCompany.com. As result it is important to monitor on the DNS server requests to this name, and to understand that if this entry is added to the DNS server that ISATAP will be enabled without any further manual configuration of the Windows 7 device.

6to4 is another dynamic IPv6 tunnel technology. For 6to4 to work autonomous it uses a well known 6to4 relay tunnel address. This well-known IPv4 address is 192.88.99.*. So any traffic directed to this IPv4 address will have a high probability of carrying encapsulated 6to4 IPv6 traffic.

Conclusion

IPv6 is out there on the network whether the network infrastructure is configured to support it or not. Several techniques have been identified in this paper to help identify this IPv6 traffic in the network. Using these tools, both tunneled and native IPv6 traffic can be identified and corrective action can be taken to improve the health of the network. These tools can also be used to help seamlessly integrate IPv6 into the network. With the integration of IPv6 into the network, more enhanced control and oversight can be utilized.

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 C11-629391-00 10/10