VRF Aware Cisco IOS IPS

Virtual Route Forwarding or Virtual Private Network (VPN) Route Forwarding (VRF), is a mechanism that allows multiple instances of a routing table to exist on a router and work simultaneously. This mechanism allows for network paths to be segregated without using multiple devices, thereby increasing network security and eliminating the need for encryption and authentication. VRFs are generally used to create separate VPNs. Allowing Intrusion Prevention System (IPS) to be configured on a per-VRF basis means global parameters will be shared by multiple VPNs, providing VRF related information on the Security Device Event Exchange (SDEE) and syslog alerts.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for VRF Aware Cisco IOS IPS

- Understand Cisco IOS IPS.
- Configure VRFs.
- Verify that the VRFs are operational.
- Verify IPS is supported.
- Capability to send SDEE alarms and syslog with VRF information.
- If two VPN networks have overlapping addresses, VRF-aware network address translation (NAT) is required for them to support VRF Aware Cisco IOS IPS.
- Every VRF instance requires a new IPS rule.

Restrictions for VRF Aware Cisco IOS IPS

- VRF Aware Cisco IOS IPS is not supported on Multiprotocol Label Switching (MPLS) interfaces.

Information About VRF Aware Cisco IOS IPS

Cisco IOS IPS

The Cisco IOS IPS acts as an in-line intrusion detection sensor, watching packets and sessions as they flow through the router and scanning each packet to match any of the Cisco IOS IPS signatures. When Cisco IOS IPS detects suspicious activity, it responds before network security can be compromised and logs the event through Cisco IOS syslog messages or SDEE. The network administrator can configure Cisco IOS IPS to choose the appropriate response to various threats. When packets in a session match a signature, Cisco IOS IPS can take any of the following actions, as appropriate:

- Send an alarm to a syslog server or a centralized management interface
- Drop the packet
- Reset the connection
- Deny traffic from the source IP address of the attacker for a specified amount of time
- Deny traffic on the connection for which the signature was seen for a specified amount of time

Cisco IOS software-based intrusion-prevention capabilities and Cisco IOS firewall are developed with flexibility in mind, so that individual signatures could be disabled in case of false positives. Generally, it is preferable to enable both the firewall and Cisco IOS IPS to support network security policies. However, each of these features may be enabled independently and on different router interfaces.

VRF

Each VPN has its own routing and forwarding table in the router, so any customer or site that belongs to a VPN is provided access only to the set of routes contained within that table. Any Provider Edge (PE) router in the MPLS VPN network therefore contains a number of per-VPN routing tables and a global routing table that is used to reach other routers in the service provider network. Effectively, virtual routers are created in a single physical router.

VRF is a Cisco IOS route table instance for connecting a set of sites to a VPN service. A VRF contains a template of a VPN Routing/Forwarding table in a PE router.

The overlapping addresses, usually resulting from the use of private IP addresses in customer networks, are one of the major obstacles to successful deployment of peer-to-peer VPN implementation. The MPLS VPN technology provides a solution to this dilemma.

VRF Lite

VRF-lite is a feature that enables a service provider to support two or more VPNs, where IP addresses can be overlapped among the VPNs. VRF Lite uses input interfaces to distinguish routes for different VPNs and forms virtual packet-forwarding tables by associating one or more Layer 3 interfaces with each VRF. Interfaces in a VRF can be physical, such as Ethernet ports, or logical, such as VLAN switched virtual interfaces (SVIs). However, a Layer 3 interface cannot belong to more than one VRF at a time.

Note VRF Lite interfaces must be Layer 3 interfaces.

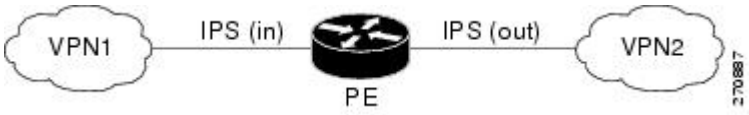
VRF-lite includes these devices:

- Customer edge (CE) devices provide customer access to the service provider network over a data link to one or more PE routers. The CE device advertises the site's local routes to the PE router and learns the remote VPN routes from it. A Catalyst 4500 switch can be a CE device.
- PE routers exchange routing information with CE devices by using static routing or a routing protocol such as BGP, RIPv1, or RIPv2.
- Provider routers (or core routers) are any routers in the service provider network that do not attach to CE devices.
- The PE is only required to maintain VPN routes for those VPNs to which it is directly attached, eliminating the need for the PE to maintain all of the service provider VPN routes. Each PE router maintains a VRF for each of its directly connected sites. Multiple interfaces on a PE router can be associated with a single VRF if all of these sites participate in the same VPN. Each VPN is mapped to a specified VRF. After learning local VPN routes from CEs, a PE router exchanges VPN routing information with other PE routers by using Internal BGP (IBPG).

With VRF Lite, multiple customers can share one CE, and only one physical link is used between the CE and the PE. The shared CE maintains separate VRF tables for each customer and switches or routes packets for each customer based on its own routing table. VRF Lite extends limited PE functionality to a CE device, giving the CE device the ability to maintain separate VRF tables to extend the privacy and security of a VPN to the branch office.

If VRF is configured on an interface and then IPS is attached, an IPS control block is created as shown in the figure below, with an appropriate name for the VRF instead of the existing default value.VRF support for IPS allows customization of IPS parameters, settings and statistics per VRF interface and customization of IPS signature sets for each VRF user. The IPS VRF code takes care of the VRF support including overlapping addresses.

Figure 1: IPS in a VRF-to-VRF Scenario



Applying IPS Directly to a VRF

Virtual Route Forwarding (VRF) is a mechanism that allows multiple instances of a routing table to exist on a router and work simultaneously. This mechanism allows for network paths to be segregated without using multiple devices which increases network security and eliminates the need for encryption and authentication. VRFs are generally used to create separate Virtual Private Networks (VPNs). If VRF is configured on an interface and IPS is attached, an IPS control block is created with the appropriate name for the VRF instead of the existing default value. VRF support for IPS allows customization of IPS parameters, settings and statistics per VRF interface and customization of IPS signature sets for each VRF user. All interfaces share the same global parameters for IPS, but alarms and event log information on the SDEE and syslog alerts carry respective VRF information.

How to VRF Aware Cisco IOS IPS

Configuring a VRF and Applying IPS Directly to the VRF

The following steps are used to configure VRF routing and forwarding tables, configuring VRF on an interface and attach IPS to this interface:

Note If a global VRF is removed, the IPS configuration on the interfaces belonging to that VRF is cleaned and removed, including any sessions and statistics created on the VRF. The user must reconfigure IPS on the affected interfaces if they are to be used again.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip vrf vrfname**
4. **rd route-distinguisher**
5. **route-target export target VPN extended community**
6. **route-target import target VPN extended community**
7. **exit**
8. **interface FastEthernet port**
9. **ip vrf forwarding vrfname**
10. **ip address range**
11. **ip ips** *ips-name* **in**
12. **exit**
13. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ip vrf vrfname** **Example:** `Router# ip vrf VRF600`	Configures a VRF table and enters VRF configuration mode.
Step 4	**rd route-distinguisher** **Example:** `Router# rd 100:600`	Creates routing and forwarding tables for the VRF instance. **Note** There are two formats for configuring the route distinguisher argument. It can be configured in the as-number:network number (ASN:nn) format, as shown in the example, or it can be configured in the IP address:network number format (IP-address:nn).

	Command or Action	Purpose
Step 5	**route-target export target VPN extended community** **Example:** `Router(config-vrf)# route-target export 100:600`	Creates lists of export route-target extended communities for the specified VRF.
Step 6	**route-target import target VPN extended community** **Example:** `Router(config-vrf)#route-target import 100:600`	Creates lists of import route-target extended communities for the specified VRF.
Step 7	**exit** **Example:** `Router(config-vrf)# exit`	Exits VRF configuration mode.
Step 8	**interface FastEthernet port** **Example:** `Router(config)# interface FastEthernet0/1.600`	Enters subinterface configuration mode and specifies a subinterface that is associated with a VRF.
Step 9	**ip vrf forwarding vrfname** **Example:** `Router(config-subif)# ip vrf forwarding VRF600`	Configures the forwarding details for the respective interfaces.
Step 10	**ip address range** **Example:** `Router(config-subif)# ip address 192.168.25.3 255.255.255.0`	Sets a primary or secondary IP address for an interface.
Step 11	**ip ips** *ips-name* **in** **Example:** `Router(config-subif)# ip ips ips_policy600 in`	Applies an IPS rule at an interface and automatically loads the signatures and builds the signature engines.

	Command or Action	Purpose

Step 12	**exit**
	
	Example:
	`Router(config-subif)# exit`
	
	Example:
	`Router(config-if)# end`
		Exits subinterface mode, and enters interface configuration mode.

Step 13	**end** **Example:** `Router(config-if)# end`	Exits interface configuration mode.

Configuration Examples for VRF Aware Cisco IOS IPS

Example Cisco IOS IPS Configuration

The following example shows how to enable and verify Cisco IOS IPS on your router:

```
Router# mkdir
flash:/ips5
Create directory filename [ips5]?
Created dir flash:/ips5
Router#
Router#
Router#
Router# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)# ip ips name MYIPS
Router(config)# ip ips config location
flash:/ips5
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 advanced
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]
Router(config)# d
*Nov 14 2006 17:16:42 MST: Applying Category configuration to signatures ..
Router(config)#
Router(config)# do show ip interface brief
Interface                  IP-Address      OK?    Method  Status                  Protocol
GigabitEthernet0/0         10.0.20.120     YES    NVRAM   up                      up
GigabitEthernet0/1         10.12.100.120   YES    NVRAM   administratively down  down
NVI0                       unassigned      NO     unset   up                      up
```

```
Router(config)#
Router(config)# interface gigabits 0/0
Router(config-if)# ip ips MYIPS in
Router(config-if)#
*Nov 14 2006 17:17:07 MST: %IPS-6-ENGINE_BUILDS_STARTEf: 17:17:07 MST Nov 14 2006
*Nov 14 2006 17:17:07 MST: %IPS-6-ENGINE_BUILDING: atomic-ip - 3 signatures - 1 of 13 engines
*Nov 14 2006 17:17:07 MST: %IPS-6-ENGINE_READY: atomic-ip - build time 0 ms - packets for
this engine will be scanned
*Nov 14 2006 17:17:07 MST: %IPS-6-ALL_ENGINE_BUILDS_COMPLETE: elapsed time 0 ms
Router(config-if)#
Router(config-if)# ip ips MYIPS out
Router(config-if)#
Router(config-if)#
Router(config-if)#^Z
Router#
*Nov 14 2006 17:17:23 MST: %SYS-5-CONFIG_I: Configured from console by cisco on console
Router# wr
Building configuration...
[OK]
Router#
Router# show ip ips signature count
Cisco SDF release version S0.0
Signature Micro-Engine: multi-string (INACTIVE)
Signature Micro-Engine: service-http (INACTIVE)
Signature Micro-Engine: string-tcp (INACTIVE)
Signature Micro-Engine: string-udp (INACTIVE)
Signature Micro-Engine: state (INACTIVE)
Signature Micro-Engine: atomic-ip
     Total Signatures: 3
       Enablef: 0
       Compilef: 3
Signature Micro-Engine: string-icmp (INACTIVE)
Signature Micro-Engine: service-ftp (INACTIVE)
Signature Micro-Engine: service-rpc (INACTIVE)
Signature Micro-Engine: service-dns (INACTIVE)
Signature Micro-Engine: normalizer (INACTIVE)
Signature Micro-Engine: service-smb-advanced (INACTIVE)
Signature Micro-Engine: service-msrpc (INACTIVE)
     Total Signatures: 3
     Total Enabled Signatures: 0
     Total Retired Signatures: 0
     Total Compiled Signatures: 3
Router#
Router# copy flash:IOS-S258-CLI-kd.pkg idconf
*Nov 14 2006 17:19:47 MST: %IPS-6-ENGINE_BUILDS_STARTEf: 17:19:47 MST Nov 14 2006
*Nov 14 2006 17:19:47 MST: %IPS-6-ENGINE_BUILDING: multi-string - 3 signatures - 1 of 13
engines
*Nov 14 2006 17:19:47 MST: %IPS-6-ENGINE_READY: multi-string - build time 4 ms - packets
for this engine will be scanned
*Nov 14 2006 17:19:47 MST: %IPS-6-ENGINE_BUILDING: service-http - 611 signatures - 2 of 13
 engines
*Nov 14 2006 17:20:00 MST: %IPS-6-ENGINE_READY: service-http - build time 12932 ms - packets
 for this engine will be scanned
*Nov 14 2006 17:20:00 MST: %IPS-6-ENGINE_BUILDING: string-tcp - 864 signatures - 3 of 13
engines
*Nov 14 2006 17:20:02 MST: %IPS-6-ENGINE_READY: string-tcp - build time 2692 ms - packets
for this engine will be scanned
*Nov 14 2006 17:20:02 MST: %IPS-6-ENGINE_BUILDING: string-udp - 74 signatures - 4 of 13
engines
*Nov 14 2006 17:20:03 MST: %IPS-6-ENGINE_READY: string-udp - build time 316 ms - packets
for this engine will be scanned
*Nov 14 2006 17:20:03 MST: %IPS-6-ENGINE_BUILDING: state - 28 signatures - 5 of 13 engines
*Nov 14 2006 17:20:03 MST: %IPS-6-ENGINE_READY: state - build time 24 ms - packets for this
 engine will be scanned
*Nov 14 2006 17:20:03 MST: %IPS-6-ENGINE_BUILDING: atomic-ip - 252 signatures - 6 of 13
engines
*Nov 14 2006 17:20:03 MST: %IPS-4-META_ENGINE_UNSUPPORTEf: atomic-ip 2154:0 - this signature
 is a component of the unsupported META engine
*Nov 14 2006 17:20:03 MST: %IPS-6-ENGINE_READY: atomic-ip - build time 232 ms - packets for
 this engine will be scanned
*Nov 14 2006 17:20:03 MST: %IPS-6-ENGINE_BUILDING: string-icmp - 3 signatures - 7 of 13 e
Router# engines
*Nov 14 2006 17:20:03 MST: %IPS-6-ENGINE_READY: string-icmp - build time 12 ms - packets
```

```
for this engine will be scanned
*Nov 14 2006 17:20:03 MST: %IPS-6-ENGINE_BUILDING: service-ftp - 3 signatures - 8 of 13
engines
*Nov 14 2006 17:20:03 MST: %IPS-6-ENGINE_READY: service-ftp - build time 8 ms - packets for
 this engine will be scanned
*Nov 14 2006 17:20:03 MST: %IPS-6-ENGINE_BUILDING: service-rpc - 75 signatures - 9 of 13
engines
*Nov 14 2006 17:20:03 MST: %IPS-6-ENGINE_READY: service-rpc - build time 80 ms - packets
for this engine will be scanned
*Nov 14 2006 17:20:03 MST: %IPS-6-ENGINE_BUILDING: service-dns - 38 signatures - 10 of 13
engines
*Nov 14 2006 17:20:03 MST: %IPS-6-ENGINE_READY: service-dns - build time 20 ms - packets
for this engine will be scanned
*Nov 14 2006 17:20:03 MST: %IPS-6-ENGINE_BUILDING: normalizer - 9 signatures - 11 of 13
engines
*Nov 14 2006 17:20:03 MST: %IPS-6-ENGINE_READY: normalizer - build time 0 ms - packets for
 this engine will be scanned
*Nov 14 2006 17:20:03 MST: %IPS-6-ENGINE_BUILDING: service-msrpc - 22 signatures - 12 of
13 engines
*Nov 14 2006 17:20:03 MST: %IPS-6-ENGINE_READY: service-msrpc - build time 8 ms - packets
for this engine will be scanned
*Nov 14 2006 17:20:03 MST: %IPS-6-ALL_ENGINE_BUILDS_COMPLETE: elapsed time 16344 ms
Router#
Router#
Router# show ip ips signature count
Cisco SDF release version S258.0
Signature Micro-Engine: multi-string
     Total Signatures: 3
       Enablef: 3
       Retiref: 3
Signature Micro-Engine: service-http
     Total Signatures: 611
       Enablef: 159
       Retiref: 428
       Compilef: 183
Signature Micro-Engine: string-tcp
     Total Signatures: 864
       Enablef: 414
       Retiref: 753
       Compilef: 111
Signature Micro-Engine: string-udp
     Total Signatures: 74
       Enablef: 1
       Retiref: 44
       Compilef: 30
Signature Micro-Engine: state
     Total Signatures: 28
       Enablef: 16
       Retiref: 25
       Compilef: 3
Signature Micro-Engine: atomic-ip
     Total Signatures: 252
       Enablef: 56
       Retiref: 148
       Compilef: 103
       Inactive - invalid params: 1
Signature Micro-Engine: string-icmp
     Total Signatures: 3
       Enablef: 0
       Retiref: 2
       Compilef: 1
Signature Micro-Engine: service-ftp
     Total Signatures: 3
       Enablef: 1
       Compilef: 3
Signature Micro-Engine: service-rpc
            Total Signatures: 75
       Enablef: 44
       Retiref: 44
       Compilef: 31
Signature Micro-Engine: service-dns
     Total Signatures: 38
       Enablef: 30
```

```
       Retiref: 5
       Compilef: 33
Signature Micro-Engine: normalizer
     Total Signatures: 9
       Enablef: 8
       Retiref: 5
       Compilef: 4
Signature Micro-Engine: service-smb-advanced (INACTIVE)
Signature Micro-Engine: service-msrpc
     Total Signatures: 22
       Enablef: 22
       Retiref: 22
```

Example VRF Aware Cisco IOS IPS Configuration Without Subinterfaces

The following example shows a physical interface supporting a VRF forwarding table with IPS enabled on VRF600:

```
Router(config)# ip vrf
 VRF600
Router(config-vrf)# rd
 100:600
Router(config-vrf)# route-target export
 100:600
Router(config-vrf)# route-target import
100:600
Router(config-vrf)# exit
Router(config)# interface FastEthernet
0/1
Router(config-subif)# ip vrf forwarding
 VRF600
Router(config-subif)# ip address
 192.168.00.3 192.168.255.225
Router(config-subif)# ip ips
 ips_policy600 in

router(config-subif)# exit

router(config-if)# end
```

Example VRF Aware Cisco IOS IPS Configuration with Subinterfaces

The following example shows two physical interfaces supporting two VRF forwarding tables, VRF600 and VRF601, with IPS enabled on only VRF600:

```
config terminal
Router1(config)# ip vrf
 VRF600
Router1(config-vrf)# rd
 100:600
Router1(config-vrf)# route-target export
 100:600
Router1(config-vrf)# route-target import
 100:600
Router1(config-vrf)# exit
Router1(config)# ip vrf
VRF601
Router1(config-vrf)# rd
 100:601
Router1(config-vrf)# route-target export
100:601
Router1(config-vrf)# route-target import
```

```
100:601
Router1(config-vrf)# exit
Router1(config)# interface FastEthernet
0/0.600
Router1(config-subif)# encapsulation dot1Q
600
Router1(config-subif)# ip vrf forwarding
VRF600
Router1(config-subif)# ip address
 192.168.00.0 192.168.255.0
Router1(config-subif)# exit
Router1(config)# interface FastEthernet
0/0.601
Router1(config-subif)# encapsulation dot1Q
601
Router1(config-subif)# ip vrf forwarding
 VRF601
Rrouter1(config-subif)# ip address
192.168.00.0 192.168.255.0
Router1(config-subif)# end


config terminal

Router2(config)# ip ips name
 ips_policy600

Router2(config)# ip vrf
 VRF600
Router2(config-vrf)# rd
 100:600
Router2(config-vrf)# route-target export
 100:600
Router2(config-vrf)# route-target import
 100:600
Router2(config-vrf)# exit

Router2(config)# ip vrf
VRF601
Router2(config-vrf)# rd
 100:601
Router2(config-vrf)# route-target export
 100:601
Router2(config-vrf)# route-target import
 100:601
Router2(config-vrf)# exit

Router2(config)# interface FastEthernet
0/1.600
Router2(config-subif)# encapsulation dot1Q
 600
Router2(config-subif)# ip vrf forwarding
 VRF600
Router2(config-subif)# ip address
192.168.00.0 192.168.255.0
Router2(config-subif)# ip ips
ips_policy600 in
Router2(config-subif)# exit

Router2(config)# interface FastEthernet
0/1.601
Router2(config-subif)# encapsulation dot1Q
 601
Router2(config-subif)# ip vrf forwarding
 VRF601
Router2(config-subif)# ip address
 192.168.00.0 192.168.255.0
Router2(config-subif)# end
```

Example Multi VRF with IPS and Zone Based Policy (ZBP) Firewall

The following example shows Multiple VRFs configured with IPS and ZBP firewalls:

```
ip cef
!
ip vrf VRF_600
 rd 100:110
 route-target export 100:1000
 route-target import 100:1000
!
ip vrf VRF_601
 rd 100:120
 route-target export 100:2000
 route-target import 100:2000
!
ip ips config location flash:ips5/ retries 1
ip ips name IPS_POLICY_201
ip ips name IPS_POLICY_VRF_600
ip ips name IPS_POLICY_VRF_601
!
ip ips signature-category
  category all
   retired true
!
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
!
class-map type inspect match-any L4-cmti
 match protocol tcp
 match protocol udp
 match protocol icmp
!
policy-map type inspect inside201-outside-pmti
 class type inspect L4-cmti
  inspect
!
policy-map type inspect inside600-outside-pmti
 class type inspect L4-cmti
  inspect
!
policy-map type inspect inside602-outside-pmti
 class type inspect L4-cmti
  inspect
!
zone security inside201
zone security inside600
zone security inside601
zone security outside
!
zone-pair security inside201-outside source inside201 destination outside
 service-policy type inspect inside201-outside-pmti
!
zone-pair security inside600-outside source inside600 destination outside
 service-policy type inspect inside600-outside-pmti
!
zone-pair security inside601-outside source inside602 destination outside
```

```
 service-policy type inspect inside602-outside-pmti
!
interface Loopback0
 ip address 10.10.10.4 10.255.255.255
 ip router isis
!
interface GigabitEthernet0/0
 no ip address
 duplex auto
 speed auto
 media-type rj45
 no keepalive
!
interface GigabitEthernet0/0.201
 encapsulation dot1Q 201
 ip address 192.168.00.0 192.168.255.0
 zone-member security inside201
 ip ips myips201 in
 ip ips myips201 out
!
interface GigabitEthernet0/0.600
 encapsulation dot1Q 600
 ip vrf forwarding VRF_600
 ip address 10.0.0.0 10.255.255.255
 zone-member security inside600
 ip ips IPS_POLICY_VRF_600 in
 ip ips IPS_POLICY_VRF_600 out
!
interface GigabitEthernet0/0.601
 encapsulation dot1Q 601
 ip vrf forwarding IPS_POLICY_VRF_601
 ip address 10.0.0.0 10.255.255.255
 zone-member security inside602
 ip ips IPS_POLICY_VRF_601 in
 ip ips IPS_POLICY_VRF_601 out
!
!
interface FastEthernet2/0
 ip address 10.1.1.14 10.255.255.255
 ip router isis
 duplex auto
 speed auto
 mpls ip
!
router isis
 net 10.225.225.225
 is-type level-1
!
router bgp 100
 no synchronization
 bgp log-neighbor-changes
 neighbor 10.10.10.6 remote-as 100
 neighbor 10.10.10.6 update-source Loopback0
 no auto-summary
 !
 address-family vpnv4
 neighbor 10.10.10.6 activate
 neighbor 10.10.10.6 send-community both
 exit-address-family
 !
 address-family ipv4 vrf VRF_600
 redistribute connected
 no synchronization
 exit-address-family
 !
 address-family ipv4 vrf VRF_601
 redistribute connected
 no synchronization
 exit-address-family
```

Examples VRF Aware Cisco IOS IPS Output and Error Message

Note All VRFs will share the same global IPS configurations, therefore, some show commands which show the information of the global items are shown for every VRF irrespective of whether the event happened on that particular VRF.

Examples VRF Aware Cisco IOS IPS Output

The following is sample output from the **show ip ips statistics**command. The output provides statistics that may not necessarily be the ones that fired on VRF 600.

```
Router# show ip ips statistics vrf VRF_600
Signature statistics [process switch:fast switch]
signature 5170:1 packets checkef: [0:4]
Interfaces configured for ips 4
Session creations since subsystem startup or last reset 0
Current session counts (estab/half-open/terminating) [0:0:0]
Maxever session counts (estab/half-open/terminating) [0:0:0]
Last session created never
Last statistic reset never
TCP reassembly statistics
received 0 packets out-of-order; dropped 0
peak memory usage 0 KB; current usage: 0 KB
peak queue length 0
```

The following is sample output from the **show ip ips interfaces command.**

```
Router# show ip ips interface
Interface Configuration
Interface GigabitEthernet0/0
VRF name: vrf1
Inbound IPS rule is tst
Outgoing IPS rule is not set
```

The following is sample output from the **show ip ips session** command.

```
Router# show ip ips session vrf vrf1
Established Sessions
Session 485EBEE8 (172.16.0.0:10001)=>(172.31.255.255:80) tcp SIS_OPEN
```

The following is sample output from the **clear ip ips statistics**command**.**

```
Router# clear ip ips statistics vrf vrf1
Router# show ip ips stat vrf vrf1

Interfaces configured for ips 1
Session creations since subsystem startup or last reset 0
Current session counts (estab/half-open/terminating) [1:0:0]
Maxever session counts (estab/half-open/terminating) [1:0:0]
Last session created 00:00:26
Last statistic reset 00:00:01
```

Examples ErrMSG with VRF Name Output

The following is sample error message output with the VRF name.

```
%IPS-4-SIGNATURE: Sig:5405 Subsig:0 Sev:100  [192.168.103.1:51129 -> 192.168.3.4:80]
VRF:vrf600 RiskRating:100
```

Examples SDEE Messages with VRF Name

The SDEE messages have been enhanced to show the VRF name. An example of output from the browser is as follows:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
  <env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope">
     <env:Body>
         <sf:events xmlns:cid="http://www.cisco.com/cids/2003/08/cidee"
xmlns:sd="http://example.org/2003/08/sdee"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://example.org/2003/08/sdee sdee.xsd
http://www.cisco.com/cids/2003/08/cidee cidee.xsd">
              <sf:evIdsAlert eventId="11630069224" vendor="Cisco" severity="unknown">
              <sf:originator>
                  <sf:hostId>iosfw-28a</sf:hostId>
               </sf:originator>
                <sf:time offset="0" timeZone="UTC">1163006922225223338</sf:time>
                 <sf:signature description="" id="5123" version="S4">
                    <cif:subsigId>0</cif:subsigId>
                    <cif:sigDetails>Host:\x3c250+ chars></cif:sigDetails>
                 </sf:signature>
                 <cif:protocol>tcp</cif:protocol>
                 <cif:riskRatingValue>85</cif:riskRatingValue>
                 <sf:participants>
                    <sf:attacker>
       <sf:addr>10.1.0.1</sf:addr>
                               <sf:port>10001</sf:port>
                     </sf:attacker>
                     <sf:target>
                            <sf:addr>10.2.0.1</sf:addr>
                            <sf:port>80</sf:port>
                     </sf:target>
               </sf:participants>
               <sf:actions></sf:actions>
                <cif:interface>Gi0/1</cif:interface>
                <cif:vrf_name>vrf1</cif:vrf_name>
              </sf:evIdsAlert>
      </sf:events>
    </env:Body>
</env:Envelope>
```

Examples SDEE show Commands

The SDEE show commands have been enhanced to include VRF specific information, the following is sample output from the **show ip sdee alerts**commanf:

```
Router# show ip sdee alerts

Alert storage: 200 alerts using 56000 bytes of memory
                                    SDEE Alerts
      SigID    Sig Name                    SrcIP:SrcPort         DstIP:DstPort     VRF
                                                 or Summary Info
   1:  5170:1                              192.162.4.0:3692       192.162.6.0:80    NONE
```

```
   2:  5170:1                                192.162.4.0:3692       192.162.6.0:80 VRF_600
```

The following is sample output from the **show ip sdee events** command.

```
Router# show ip sdee events

Alert storage:   200 alerts using 56000 bytes of memory
Message storage: 200 messages using 84800 bytes of memory
                                          SDEE Events
      Time                          Type    Description
1: 10:25:55 MST Jan 22 2007  ALERT   Sig ID     5170:1
2: 10:25:57 MST Jan 22 2007  ALERT   Sig ID     5170:1
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
Security commands	• Cisco IOS Security Command Reference: Commands A to C • Cisco IOS Security Command Reference: Commands D to L • Cisco IOS Security Command Reference: Commands M to R • Cisco IOS Security Command Reference: Commands S to Z

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for VRF Aware Cisco IOS IPS

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 1: Feature Information for VRF Aware Cisco IOS IPS

Feature Name	Releases	Feature Information
VRF aware Cisco IOS IPS	12.4(20)T	Virtual Route Forwarding or Virtual Private Network (VPN) Route Forwarding (VRF), is a mechanism that allows multiple instances of a routing table to exist on a router and work simultaneously. This mechanism allows for network paths to be segregated without using multiple devices, thereby increasing network security and eliminating the need for encryption and authentication. VRFs are generally used to create separate VPNs. Allowing Intrusion Prevention System (IPS) to be configured on a per-VRF basis means global parameters will be shared by multiple VPNs, providing VRF related information on the Security Device Event Exchange (SDEE) and syslog alerts. The following commands were introduced or modifief: **clear ip ips statistics**, **show ip ips**