Technical Guide

VRF-lite on AR-Series Firewalls

Feature Overview and Configuration Guide

Introduction

VRF-lite is used for isolating customer networks—it allows multiple secure customer routing domains, which remain completely isolated from each other, to co-exist in one physical device simultaneously.

VRF-lite also allows the re-use of IP addresses on the same physical device. An IP address range in one network interface used in one VRF domain can simultaneously be used in another network interface in a different VRF domain within the same device. While VRF-lite will segregate traffic from different customers/clients, VRF-lite can also allow access between VRF domains (inter-VRF communication), by using static inter-VRF routes. This provides controlled access from one VRF routing domain to another where the IP address ranges do not overlap.

VRF-lite can be used in conjunction with firewall features while maintaining security. Additionally, VRF routing domains can be transported across Virtual Private Networks, by associating individual VPNs with one or more VRF instances.

This document describes VRF-lite support and how to configure it on the AR-Series Firewalls. It shows how to configure the following examples:

- "Example 1: VRF-lite with firewall protection/security features"
- "Example 2: VRF-lite on a Virtual Tunnel Interface (VTI)"
- "Example 3: VRF-lite on VTIs with firewall and NAT"
- "Example 4: Multiple VRFs via IPsec VPN using Ethernet pseudowires"
- "Example 5: VRF instances from central to remote VPN sites"

These are followed by discussions on:

- "VRF-aware utilities in AlliedWare Plus"
- "Static Inter-VRF routing"
- "Configuring DNS Relay to be VRF-aware"
- "Adding a VRF-aware static ARP"
- "Local interfaces with VRF-lite"

AlliedWare Plus™ OPERATING SYSTEM

Which products and software version does it apply to?

The information provided in this document applies to AR-Series firewalls running **5.4.6-2** and above:

- AR4050S-5G from 5.5.2-0.2 onwards
- AR4050S
- AR3050S
- AR2050V
- AR2010V

Feature support may change in later software versions. For the latest information, see the product's Command Reference.

VRF-lite is also supported on several switches running AlliedWare Plus. For information about VRF-lite operation and configuration on switches, see the VRF-lite on AlliedWare Plus Switches Feature Overview and Configuration Guide.

Related documents

For information about related features, see the following documents:

- The product's Command Reference
- Getting Started with the UTM Firewall GUI Feature Overview and Configuration Guide
- Internet Protocol Security (IPsec) Feature Overview and Configuration Guide
- Interfaces Feature Overview and Configuration Guide

Contents

Introduction 1
Which products and software version does it apply to? 2
Related documents 2
Understanding VRF-lite on AR-Series firewalls 4
Routing tables 4
Interface management 4
VRF-lite support on AR-Series firewalls 5
Unsupported features 6
Supported interface types 7
Firewall security features and VRF-lite 7
Example 1: VRF-lite with firewall protection/security features 8
Virtual tunnel interfaces and VRF-lite 10
Example 2: VRF-lite on a Virtual Tunnel Interface (VTI) 10
Example 3: VRF-lite on VTIs with firewall and NAT 11
Example 4: Multiple VRFs via IPsec VPN using Ethernet pseudowires 16
Example 5: VRF instances from central to remote VPN sites 19
VRF-aware utilities in AlliedWare Plus 25
Static Inter-VRF routing 30
Configuring DNS Relay to be VRF-aware 31
Introduction 31
DNS operation with VRF-lite 31
Configuring DNS operation with VRF-lite 32
Configuring a DHCP server to be VRF aware 34
Adding a VRF-aware static ARP 37
Local interfaces with VRF-lite 37
Useful VRF-related diagnostics command list 39

Understanding VRF-lite on AR-Series firewalls

Multiple VRF instances can be defined within an AR-Series Firewall. One or more Layer 3 interfaces are associated with each VRF instance. A Layer 3 interface cannot belong to more than one VRF instance at any time. Each VRF instance is a separate routing domain.

Routing tables

Each VRF instance maintains its own IPv4 routing table independent from the routing table of the global VRF domain or other VRF instances. So, each VRF instance is a separate routing domain.

By default, before any VRF instance is configured, an AR-Series Firewall will have one route table, and routes via all IP interfaces of the AR-Series Firewall will be stored in this one table. As VRF instances are configured on the AR-Series Firewall, the original route table remains. This default route table, and its associated IP interfaces, are then referred to as the default global VRF domain.

Interface management

Each Layer 3 network interface can belong to only one VRF. Initially, every interface is in the default global VRF domain. As Layer 3 interfaces are moved to the created VRF instances, they are removed from the global VRF domain, so the global VRF domain manages a decreasing set of Layer 3 interfaces.

When a Layer 3 interface is moved to a VRF instance from the default global VRF domain, or when a Layer 3 interface is moved from one VRF instance to another via command, the interface name and ID (ifindex) are never changed as a result of the interface movement. However IP configuration on the interface in the previous VRF is unset (removed) before moving the interface to a new VRF.

VRF-lite support on AR-Series firewalls

From 5.4.6-2 onwards, VRF-lite is supported on the AR-Series Firewalls as part of the standard feature set with no additional licenses required.

Here is a summary of the features provided by the AlliedWare Plus VRF-lite implementation:

- Up to 64 VRF instances (including the default VRF) can be created.
- Multiple independent routing table instances may co-exist within the same device. The same or overlapping IP addresses can be present in different route table instances without conflicting. All routing table instances remain securely isolated from each other.
- By default, no communication occurs between VRF instances, facilitating multiple secure routing domains within the same VRF-aware device. However, inter-VRF communication between routing domains is possible by using static inter-VRF routes.
- QoS (software-based) can continue to operate within a VRF instance.
- Dynamic routing protocols (OSPF, RIP and BGP) are supported operating within a VRF. Dynamic inter-VRF routing is not supported, i.e. dynamic routing protocols cannot be used to transfer routes between VRF instances.
- Static routing can be configured between VRF instances (static inter-VRF routing) as long as the IP addresses do not overlap.
- Detailed diagnostic and debugging information is available:
  - Ability to view routing table information per VRF.
  - All appropriate VRF-related information and error messages can be viewed in the system-wide log.
- All Layer 3 interfaces and associated Layer 2 switch ports remain in the default global VRF domain until associated with a specific VRF instance.
- The default global VRF domain always exists and cannot be removed. Initially during startup, every Layer 3 interface belongs to the default global VRF domain. Also, when a Layer 3 interface is removed from a VRF, it is automatically returned to the default global VRF domain. Only one default global VRF domain exists in each physical device.

The following VRF-aware services are supported on the AR-Series Firewalls:

- Ping
- Trace Route
- Telnet Client
- SSH Server and Client
- TCP Dump
- File Copy
- System Log
- DHCP Server
- HTTP Server and Client
- TFTP Client
- SNMP Server and Client
- NTP server
- sFlow Agent

For more detail, see "VRF-aware utilities in AlliedWare Plus" on page 25.

Unsupported features

The following features are not supported on a per-VRF basis:

- IPv6
- AMF
- multicast protocols (PIM-SM/PIM-DM)
- VRRP

However, these services remain supported in the global VRF domain.

Supported interface types

The following firewall interface types can be assigned to a VRF-instance:

- VLAN interfaces
- Ethernet (WAN) interfaces
- Ethernet (WAN) sub-interfaces (802.1Q tagged interfaces)
- PPPoE interfaces
- Virtual Tunnel Interfaces (VTI) with tunnel modes GRE, IPSec, OpenVPN and unmanaged L2TPv3 Ethernet pseudowires

Note: It is not be possible to configure OpenVPN clients to be restricted to a particular VRF. Multiple OpenVPN tunnels operating in different VRFs can, however, are supported by configuring OpenVPN mode VTI tunnels using different listening ports. OpenVPN clients cannot be limited to a particular VRF instance via login credentials.

- Bridge interfaces
- Local (loopback) interfaces

Firewall security features and VRF-lite

Firewall and NAT rules can be configured as long as the IP addresses used in different VRF instances do not overlap. This includes both firewall and NAT application rules to control access between firewall entities such as firewall zones, networks and hosts. VRF instances can be configured with overlapping IP addresses if firewall and NAT rules are not used.

The software update service used for updating the GUI and stream-based UTM security signature files for UTM firewalls is not VRF-aware. Features that require access to a live update server in order to download signature files will continue to receive updates, provided that the server is reachable from the default VRF. This includes IDS/IPS, URL Filtering, Application Control, Malware Protection and IP Reputation.

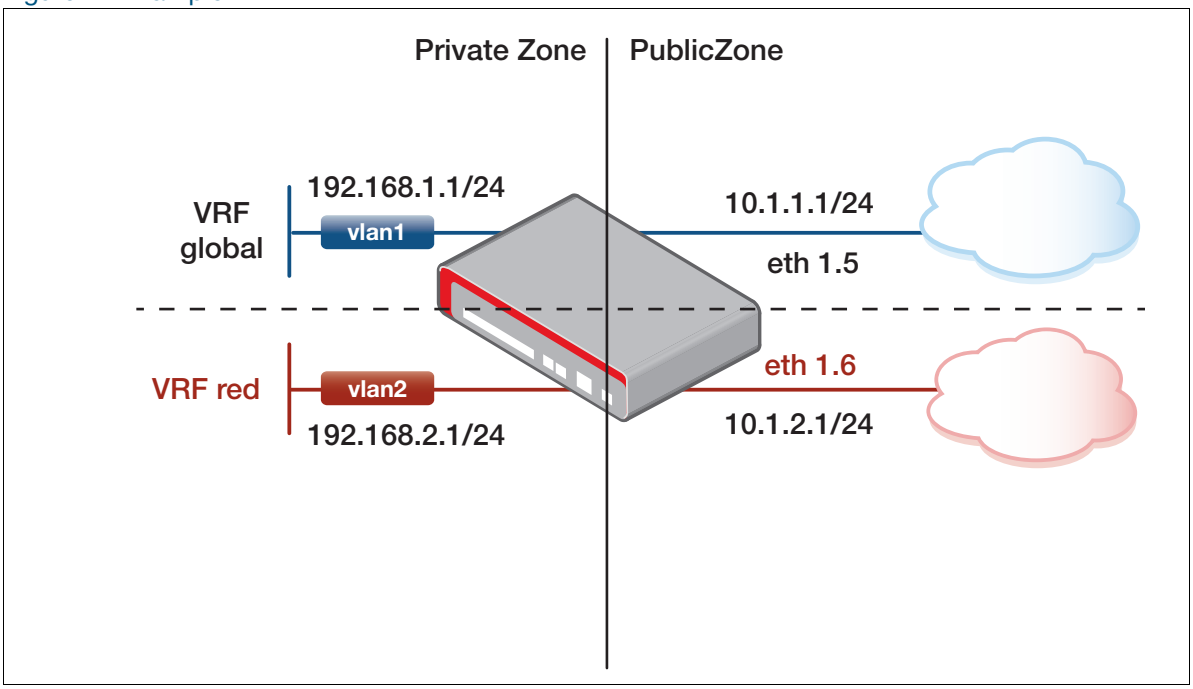
VRF can be used in conjunction with stream-based UTM features (Intrusion Detection and Prevention System, Application Control (DPI), Malware Protection, IP reputation, URL Filtering). However, the alerts, (event logging) generated by these features do not provide VRF information. This is because the stream-based security features operate on a packet level, intercepting, inspecting and processing packet data streams as they enter the device. This means that the stream inspection engine cannot know which particular Layer 3 interface the packets belong to and so cannot determine which VRF they are associated with.

VRF-lite and proxy-based UTM firewall security features (Web-control, Anti-virus) are not supported together. If VRF-lite is enabled, the proxy-based security features should be disabled.

Example 1: VRF-lite with firewall protection/security features

This example shows how to configure an AR-series firewall with a public and private zone, firewall rules and NAT rules in conjunction with VRF-lite. Traffic from the internal private zone is able to flow to the external (public) zone via the firewall rules. NAT (masquerade) rules ensure the source IP of traffic flowing from the internal (private) zone to the external (public) zone is translated, so that it appears to originate from the (public) external IP address associated with each WAN link. In this example, there is single physical Ethernet WAN interface, configured with 802.1Q Ethernet sub-interfaces.

Figure 1: Example 1



In this example:

- NAT applies the global address 10.1.1.1 to traffic from VLAN1 as it goes out onto the WAN link eth1.5.
- NAT applies the global address 10.1.2.1 to traffic from VLAN2 as it goes out onto the WAN link eth1.6.

Figure 2: Example 1—configuration

```
!
zone external
 network wan_global
  ip subnet 10.1.1.0/24
 network wan_red
  ip subnet 10.1.2.0/24
!
zone internal
 network lan_global
  ip subnet 192.168.1.0/24
 network lan_red
  ip subnet 192.168.2.0/24
!
firewall
 rule 10 permit any from internal.lan_global to external.wan_global
 rule 20 permit any from internal to internal
 rule 30 permit any from internal.lan_red to external.wan_red
 protect
!
nat
 rule 10 masq any from internal.lan_global to external.wan_global
 rule 20 masq any from internal.lan_red to external.wan_red
 enable
!
ip vrf red
!
vlan database
 vlan 2 state enable
!
interface port1.0.2
 switchport access vlan 2

interface eth1
 encapsulation dot1q 5
 encapsulation dot1q 6
!
interface eth1.5
 ip address 10.1.1.1/24
!
interface eth1.6
 ip vrf forwarding red
 ip address 10.1.2.1/24
!
interface vlan1
 ip address 192.168.1.1/24
!
interface vlan2
 ip vrf forwarding red
 ip address 192.168.2.1/24
!
ip route 0.0.0.0/0 10.1.1.254
ip route vrf red 0.0.0.0/0 10.1.2.254
!
```

Virtual tunnel interfaces and VRF-lite

It is possible to assign a Virtual Tunnel Interface (VTI) operating in GRE, IPsec, OpenVPN or unmanaged L2TPv3 modes, to a VRF instance. This ensures IPv4 traffic routed over the VTI can be associated with a VRF instance.

Note that it is **not** possible to configure a specific transport VRF within a VTI. The tunnel source and destination IP addresses used to transport the VRF-aware VPNs can only be in the global VRF. Also, IPsec tunnels associated with a VRF instance will not pass multicast packets. Therefore RIPv2 and OSPF will not discover neighbors via the reserved multicast address range. However, it is possible to configure static neighbors for the unicast routing protocols to operate within a VRF instance.

Example 2: VRF-lite on a Virtual Tunnel Interface (VTI)

A VRF-aware VPN can be formed and transported between a pair of AR-Series Firewalls that are separated by an intermediate network (located within the default global VRF), or across the Internet.

This configuration example shows how to associate a GRE-mode Virtual Tunnel Interface (VTI) Tunnel1, with VRF instance 'red'. Eth1 is the interface to the WAN link and it is used as the VPN tunnel source. Both eth1 and vlan1 are in the global VRF. Only traffic from vlan2 (also in VRF red) will flow via the VPN.

Figure 3: Example 2

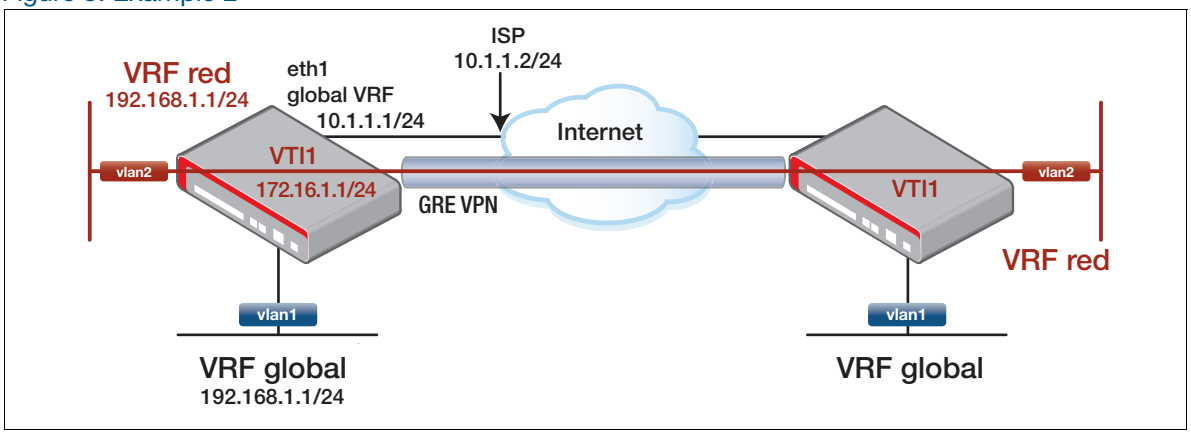


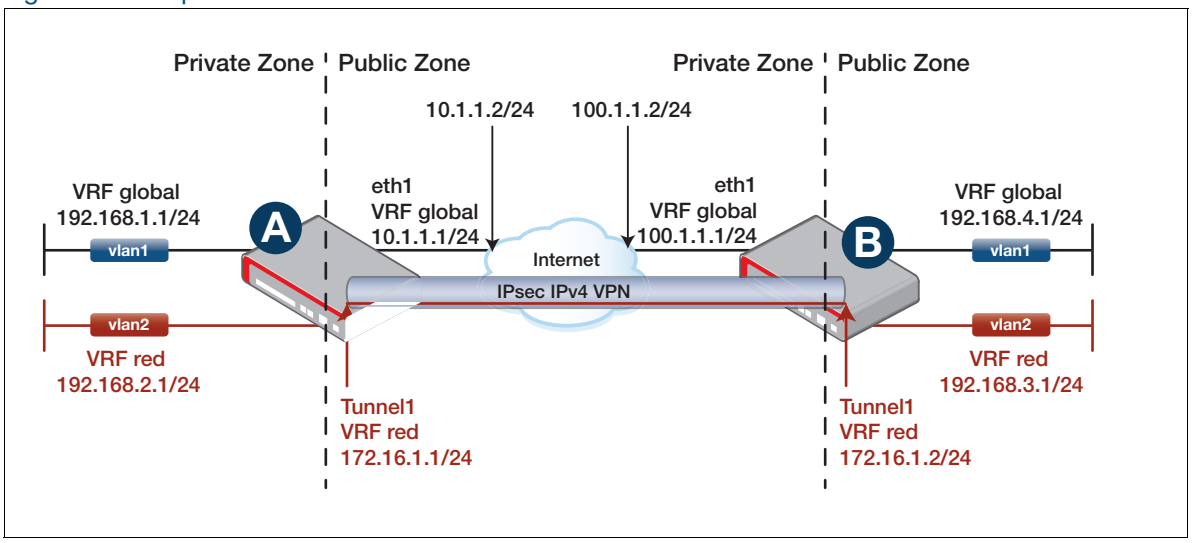
Figure 4: Example 2—configuration

```
!
ip vrf red
!
vlan database
 vlan 2 state enable
!
interface port1.0.2
 switchport access vlan 2
!
interface eth1
 ip address 10.1.1.1/24
!
interface vlan1
 ip address 192.168.1.1/24
!
interface vlan2
 ip vrf forwarding red
 ip address 192.168.1.1/24
!
interface tunnel1
 ip vrf forwarding red
 tunnel source 10.1.1.1
 tunnel destination 100.1.1.100
 tunnel mode gre
 ip address 172.16.1.1/24
!
ip route 0.0.0.0/0 10.1.1.2
ip route vrf red 192.168.2.0/24 172.16.1.2
!
```

Example 3: VRF-lite on VTIs with firewall and NAT

This example builds on the previous example, showing how to configure a virtual tunnel interface and VRF-lite. It includes the full AR-Series Firewall configurations for both sites, including the firewall and NAT rules. It also includes an intermediate Layer 3 device simulating the Internet.

Figure 5: Example 3



AR-Series Firewall A

Figure 6: Example 3—configuration for AR-Series Firewall A

```
!
hostname RouterA
!
zone private
 network local
  ip subnet 192.168.1.0/24 interface vlan1
 network local_red
  ip subnet 192.168.2.0/24 interface vlan2
 network remote_red
  ip subnet 192.168.3.0/24
 network tunnel1
  ip subnet 172.16.1.0/24
!
zone public
 network all
  ip subnet 0.0.0.0/0
 network intf
  ip subnet 10.1.1.0/24 interface eth1
  host router
   ip address 10.1.1.1
!
application esp
 protocol 50
!
application isakmp
 protocol udp
 sport 500
 dport 500
!
firewall
 rule 10 permit any from private to private
 rule 20 permit any from private.local to public
 rule 30 permit esp from public.intf.router to public
 rule 40 permit isakmp from public.intf.router to public
 rule 50 permit esp from public to public.intf.router
 rule 60 permit isakmp from public to public.intf.router
 protect
!
nat
 rule 10 masq any from private.local to public
 enable
!
crypto isakmp key 8 <samplekey> address 100.1.1.1
!
ip vrf red 1
!
interface vlan1
 ip address 192.168.1.1/24
!
interface vlan2
 ip vrf forwarding red
 ip address 192.168.2.1/24
!
```

Figure 6: Example 3—configuration for AR-Series Firewall A (continued)

```
interface tunnel1
 mtu 1438
 ip vrf forwarding red
 tunnel source 10.1.1.1
 tunnel destination 100.1.1.1
 tunnel protection ipsec
 tunnel mode ipsec ipv4
 ip address 172.16.1.1/24
!
ip route 0.0.0.0/0 10.1.1.2
ip route vrf red 192.168.3.0/24 172.16.1.2
!
```

AR-Series Firewall B

Figure 7: Example 3—configuration for AR-Series Firewall B

```
!
hostname RouterB
!
zone private
 network local
  ip subnet 192.168.4.0/24 interface vlan1
 network local_red
  ip subnet 192.168.3.0/24 interface vlan2
 network remote_red
  ip subnet 192.168.2.0/24
 network tunnel1
  ip subnet 172.16.1.0/24
!
zone public
 network all
  ip subnet 0.0.0.0/0
 network intf
  ip subnet 100.1.1.0/24 interface eth1
  host router
   ip address 100.1.1.1
!
application esp
 protocol 50
!
application isakmp
 protocol udp
 sport 500
 dport 500
!
firewall
 rule 10 permit any from private to private
 rule 20 permit any from private.local to public
 rule 30 permit esp from public.intf.router to public
 rule 40 permit isakmp from public.intf.router to public
 rule 50 permit esp from public to public.intf.router
 rule 60 permit isakmp from public to public.intf.router
 protect
!
```

Figure 7: Example 3—configuration for AR-Series Firewall B (continued)

```
nat
 rule 10 masq any from private.local to public
 enable
!
crypto isakmp key <samplekey> address 10.1.1.1
!
ip vrf red 1
!
vlan database
 vlan 2 state enable
!
interface port1.0.2
 switchport access vlan 2
!
interface eth1
 ip address 100.1.1.1/24
!
interface vlan1
 ip address 192.168.4.1/24
!
interface vlan2
 ip vrf forwarding red
 ip address 192.168.3.1/24
!
interface tunnel1
 mtu 1438
 ip vrf forwarding red
 tunnel source 100.1.1.1
 tunnel destination 10.1.1.1
 tunnel protection ipsec
 tunnel mode ipsec ipv4
 ip address 172.16.1.2/24
!
ip route 0.0.0.0/0 100.1.1.2
ip route vrf red 192.168.2.0/24 172.16.1.1
!
```

Int. device

Figure 8: Example 3—configuration for intermediate device simulating Internet

```
!
hostname Internet
!
vlan database
 vlan 2
!
interface port1.0.2
 switchport access vlan 2
!
interface vlan1
 ip address 10.1.1.2/24
!
interface vlan2
 ip address 100.1.1.2/24
!
```

Diagnostics

The outputs displayed by the following commands can be used to check that interfaces are correctly associated with VRF red, that the VPN security associations have negotiated successfully, and that ping is successful across the VRF-aware VPN.

Figure 9: Example 3— output from show ip interface brief on AR-Series Firewall A

```
RouterA#show ip interface brief
Interface              IP-Address          Status          Protocol
eth1                   10.1.1.1/24         admin up        running
eth2                   unassigned          admin up        down
lo                     unassigned          admin up        running
vlan1                  192.168.1.1/24      admin up        running

[VRF: red]
Interface              IP-Address          Status          Protocol
lo1                    unassigned          admin up        running
vlan2                  192.168.2.1/24      admin up        down
tunnel1                172.16.1.1/24       admin up        running
```

Figure 10: Example 3—output from show ipsec sa on AR-Series Firewall A

```
RouterA#show ipsec sa
------------------------------------------------------------------------
Peer              SPI (in:out)          Mode          Proto  Expires
                  Encryption            Integrity     PFS
------------------------------------------------------------------------
100.1.1.1         c8ad86ca:c99a21bf     tunnel        ESP    22801s
                  AES256                SHA256        -
```

Figure 11: Example 3—output from ping on AR-Series Firewall A

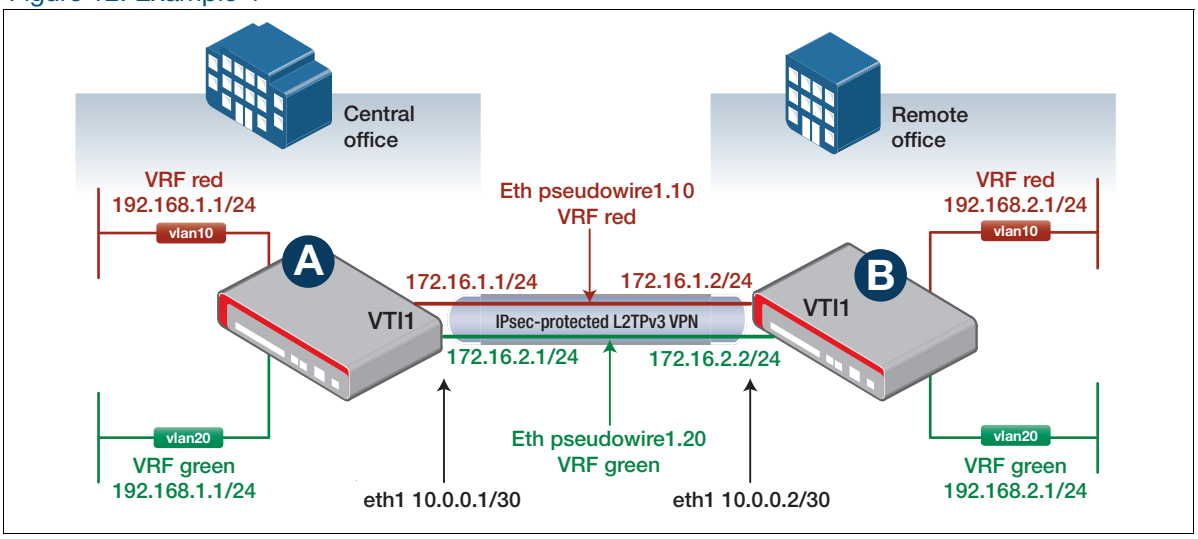
```
RouterA#ping vrf red 172.16.1.2
64 bytes from 172.16.1.2: icmp_seq=1 ttl=64 time=1.11 ms
64 bytes from 172.16.1.2: icmp_seq=2 ttl=64 time=0.762 ms
64 bytes from 172.16.1.2: icmp_seq=3 ttl=64 time=0.717 ms
64 bytes from 172.16.1.2: icmp_seq=4 ttl=64 time=0.720 ms
64 bytes from 172.16.1.2: icmp_seq=5 ttl=64 time=0.707 ms

--- 172.16.1.2 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 3998ms
rtt min/avg/max/mdev = 0.707/0.803/1.112/0.158 ms
```

Example 4: Multiple VRFs via IPsec VPN using Ethernet pseudowires

This example shows how to transport several VRFs instances via a single IPsec VPN between two sites. The single encrypted VPN operating between the pair of sites encapsulates and transports a couple of unmanaged (static) L2TPv3 Ethernet pseudowires. Each individual Ethernet pseudowire is associated with a different VRF instance.

Figure 12: Example 4



AR-Series Firewall A

Figure 13: Example 4—configuration for AR-Series Firewall A

```
!
hostname RouterA
!
crypto isakmp key <samplekey> address 10.0.0.2
!
ip vrf red 10
!
ip vrf green 20
!
vlan database
 vlan 10,20 state enable
!
interface port1.0.1
 switchport access vlan 10
!
interface port1.0.2
 switchport access vlan 20
!
interface eth1
 ip address 10.0.0.1/24
!
interface vlan10
 ip vrf forwarding red
 ip address 192.168.1.1/24
!
```

Figure 13: Example 4—configuration for AR-Series Firewall A (continued)

```
interface vlan20
 ip vrf forwarding green
 ip address 192.168.1.1/24
!
interface tunnel1
 encapsulation dot1q 10
 encapsulation dot1q 20
 mtu 1488
 tunnel source eth1
 tunnel destination 10.0.0.2
 tunnel local id 1
 tunnel remote id 2
 tunnel protection ipsec
 tunnel mode l2tp v3
!
interface tunnel1.10
 ip vrf forwarding red
 ip address 172.16.1.1/24
!
interface tunnel1.20
 ip vrf forwarding green
 ip address 172.16.2.1/24
!
ip route vrf red 192.168.2.0/24 172.16.1.2
ip route vrf green 192.168.2.0/24 172.16.2.2
!
```

AR-Series Firewall B

Figure 14: Example 4—configuration for AR-Series Firewall B

```
!
hostname RouterB
!
crypto isakmp key <samplekey> address 10.0.0.1
!
ip vrf red 10
!
ip vrf green 20
!
vlan database
 vlan 10,20 state enable
!
interface port1.0.1
 switchport access vlan 10
!
interface port1.0.2
 switchport access vlan 20
!
interface eth1
 ip address 10.0.0.2/24
!
interface vlan10
 ip vrf forwarding red
 ip address 192.168.2.1/24
!
```

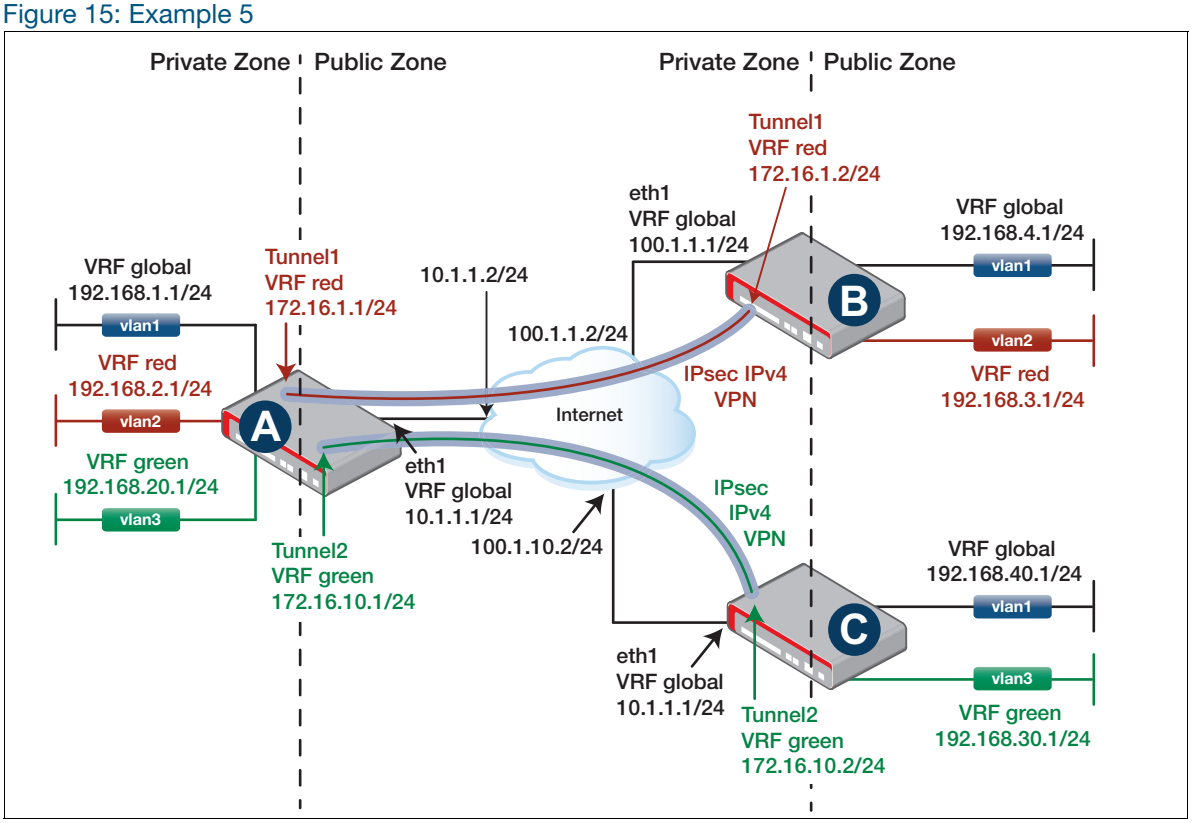
Figure 14: Example 4—configuration for AR-Series Firewall B (continued)

```
interface vlan20
 ip vrf forwarding green
 ip address 192.168.2.1/24
!
interface tunnel1
 encapsulation dot1q 10
 encapsulation dot1q 20
 mtu 1488
 tunnel source eth1
 tunnel destination 10.0.0.1
 tunnel local id 2
 tunnel remote id 1
 tunnel protection ipsec
 tunnel mode l2tp v3
!
interface tunnel1.10
 ip vrf forwarding red
 ip address 172.16.1.2/24
!
interface tunnel1.20
 ip vrf forwarding green
 ip address 172.16.2.2/24
!
ip route vrf red 192.168.1.0/24 172.16.1.1
ip route vrf green 192.168.1.0/24 172.16.2.1
!
```

Example 5: VRF instances from central to remote VPN sites

This example shows a typical hub and spoke VPN topology. The central hub site is configured with a VPN to each remote site. Each VPN from the central site to each remote site is associated with a different VRF instance.

Figure 15: Example 5



AR-Series Firewall A

Figure 16: Example 5—configuration for AR-Series Firewall A

```
!
hostname RouterA
!
zone private
 network local
  ip subnet 192.168.1.0/24 interface vlan1
 network local_red
  ip subnet 192.168.2.0/24 interface vlan2
 network local_green
ip subnet 192.168.20.0/24 interface vlan3
 network remote_red
  ip subnet 192.168.3.0/24
 network remote_green
  ip subnet 192.168.30.0/24
 network tunnel1
  ip subnet 172.16.1.0/24
 network tunnel2
  ip subnet 172.16.10.0/24
!
```

Figure 16: Example 5—configuration for AR-Series Firewall A (continued)

```
zone public
 network all
  ip subnet 0.0.0.0/0
 network intf
  ip subnet 10.1.1.0/24 interface eth1
  host router
   ip address 10.1.1.1
!
application esp
 protocol 50
!
application isakmp
 protocol udp
 sport 500
 dport 500
!
firewall
 rule 10 permit any from private to private
 rule 20 permit any from private.local to public
 rule 30 permit esp from public.intf.router to public
 rule 40 permit isakmp from public.intf.router to public
 rule 50 permit esp from public to public.intf.router
 rule 60 permit isakmp from public to public.intf.router
 protect
!
nat
 rule 10 masq any from private.local to public
 enable
!
crypto isakmp key <samplekey1> address 100.1.1.1
crypto isakmp key <samplekey2> address 100.1.10.1
!
ip vrf red 1
!
ip vrf green 2
!
vlan database
 vlan 2,3 state enable
!
interface port1.0.2
 switchport access vlan 2
!
interface port1.0.3
 switchport access vlan 3
!
interface eth1
 ip address 10.1.1.1/24
!
interface vlan1
 ip address 192.168.1.1/24
!
interface vlan2
 ip vrf forwarding red
 ip address 192.168.2.1/24
!
```

Figure 16: Example 5—configuration for AR-Series Firewall A (continued)

```
interface vlan3
 ip vrf forwarding green
 ip address 192.168.20.1/24
!
interface tunnel1
 mtu 1438
 ip vrf forwarding red
 tunnel source 10.1.1.1
 tunnel destination 100.1.1.1
 tunnel protection ipsec
 tunnel mode ipsec ipv4
 ip address 172.16.1.1/24
!
interface tunnel2
 mtu 1438
 ip vrf forwarding green
 tunnel source 10.1.1.1
 tunnel destination 100.1.10.1
 tunnel protection ipsec
 tunnel mode ipsec ipv4
 ip address 172.16.10.1/24
!
ip route 0.0.0.0/0 10.1.1.2
ip route vrf red 192.168.3.0/24 172.16.1.2
ip route vrf green 192.168.30.0/24 172.16.10.2
!
```

AR-Series Firewall B

Figure 17: Example 5—configuration for AR-Series Firewall B

```
!
hostname RouterB
!
zone private
 network local
  ip subnet 192.168.4.0/24 interface vlan1
 network local_red
  ip subnet 192.168.3.0/24 interface vlan2
 network remote_red
  ip subnet 192.168.2.0/24
 network tunnel1
  ip subnet 172.16.1.0/24
!
zone public
 network all
  ip subnet 0.0.0.0/0
 network intf
  ip subnet 100.1.1.0/24 interface eth1
  host router
   ip address 100.1.1.1
!
application esp
 protocol 50
!
```

Figure 17: Example 5—configuration for AR-Series Firewall B (continued)

```
application isakmp
 protocol udp
 sport 500
 dport 500
!
firewall
 rule 10 permit any from private to private
 rule 20 permit any from private.local to public
 rule 30 permit esp from public.intf.router to public
 rule 40 permit isakmp from public.intf.router to public
 rule 50 permit esp from public to public.intf.router
 rule 60 permit isakmp from public to public.intf.router
 protect
!
nat
 rule 10 masq any from private.local to public
 enable
!
crypto isakmp key <samplekey1> address 10.1.1.1
!
ip vrf red 1
!
vlan database
 vlan 2 state enable
!
interface port1.0.2
 switchport access vlan 2
!
interface eth1
 ip address 100.1.1.1/24
!
interface vlan1
 ip address 192.168.4.1/24
!
interface vlan2
 ip vrf forwarding red
 ip address 192.168.3.1/24
!
interface tunnel1
 mtu 1438
 ip vrf forwarding red
 tunnel source 100.1.1.1
 tunnel destination 10.1.1.1
 tunnel protection ipsec
 tunnel mode ipsec ipv4
 ip address 172.16.1.2/24
!
ip route 0.0.0.0/0 100.1.1.2
ip route vrf red 192.168.2.0/24 172.16.1.1
```

AR-Series Firewall C

Figure 18: Example 5—configuration for AR-Series Firewall C

```
!
hostname RouterC
!
zone private
 network local
  ip subnet 192.168.40.0/24 interface vlan1
 network local_green
  ip subnet 192.168.30.0/24 interface vlan3
 network remote_green
  ip subnet 192.168.20.0/24
 network tunnel1
  ip subnet 172.16.10.0/24
!
zone public
 network all
  ip subnet 0.0.0.0/0
 network intf
  ip subnet 100.1.10.0/24 interface eth1
  host router
   ip address 100.1.10.1
!
application esp
 protocol 50
!
application isakmp
 protocol udp
 sport 500
 dport 500
!
firewall
 rule 10 permit any from private to private
 rule 20 permit any from private.local to public
 rule 30 permit esp from public.intf.router to public
 rule 40 permit isakmp from public.intf.router to public
 rule 50 permit esp from public to public.intf.router
 rule 60 permit isakmp from public to public.intf.router
 protect
!
nat
 rule 10 masq any from private.local to public
 enable
!
crypto isakmp key <samplekey2> address 10.1.1.1
!
ip vrf green 1
!
vlan database
 vlan 3 state enable
!
interface port1.0.3
 switchport access vlan 3
!
interface eth1
 ip address 100.1.10.1/24
!
interface vlan1
 ip address 192.168.40.1/24
!
```

Figure 18: Example 5—configuration for AR-Series Firewall C (continued)

```
interface vlan3
 ip vrf forwarding green
 ip address 192.168.30.1/24
!
interface tunnel1
 mtu 1438
 ip vrf forwarding green
 tunnel source 100.1.10.1
 tunnel destination 10.1.1.1
 tunnel protection ipsec
 tunnel mode ipsec ipv4
 ip address 172.16.10.2/24
!
ip route 0.0.0.0/0 100.1.10.2
ip route vrf green 192.168.20.0/24 172.16.10.1
!
```

Int. device

Figure 19: Example 5—configuration for intermediate Internet device

```
!
hostname Internet
!
vlan database
 vlan 2,3 state enable
!
interface port1.0.2
 switchport access vlan 2
!
interface port1.0.3
 switchport access vlan 3
!
interface vlan1
 ip address 10.1.1.2/24
!
interface vlan2
 ip address 100.1.1.2/24
!
interface vlan3
 ip address 100.1.10.2/24
!
```

VRF-aware utilities in AlliedWare Plus

A number of network utility and management features operate in a VRF-aware manner. Whenever you use one of these utilities, you can inform the utility which VRF instance to direct its traffic into.

These include:

- Ping
- Trace Route
- Telnet Client
- SSH Server and Client
- TCP Dump
- File Copy
- System Log
- DHCP Server
- HTTP Server and Client
- TFTP Client
- SNMP Server and Client
- NTP Server and Client
- sFlow Agent

Configuration examples

The following configuration examples show how to allow a number of AlliedWare Plus management features to be constrained to a named VRF. For the servers, if the server is configured with a named VRF, clients will only be able to connect to the server from within the same VRF.

For clients, if the client is configured with a named VRF, it will only be able to connect to a server through an interface that is also contained in the same named VRF.

If a VRF is not specified in the configuration the feature will reside in the global VRF as per current behavior.

Configure an SSH Server

Configure a VRF:

```
awplus(config)#ip vrf red 1
```

Configure an SSH server for VRF:

```
awplus(config)#ssh server vrf red
awplus(config)#service ssh
```

Configure Remote Logging

Configure remote logging for VRF:

```
awplus(config)#log host 10.0.0.1 vrf red
```

Configure SNMP

Configure SNMP for VRF:

```
awplus(config)#snmp-server
awplus(config)#snmp-server community public
awplus(config)#snmp-server host 10.0.0.1 vrf red public
```

Configure a RADIUS Client

Configure a radius client for VRF:

```
awplus(config)#radius-server host 10.0.0.1 vrf red
```

Configure a TACACS+ Client

Configure TACACS+ for VRF:

```
awplus(config)#tacacs-server host 10.0.0.1 vrf red key 8 g5iqzyMUQF0=
```

Configure a TFTP Client

Configure a TFTP client for VRF:

```
awplus(config)#ip tftp vrf red
```

Configure a DHCP Server

Configure a DHCP server for VRF:

```
awplus(config)#service dhcp-server
awplus(config)#ip dhcp pool pool1
awplus(config)#vrf red
```

Ping

Example of configuration for ping

```
awplus#ping ?
  WORD  Ping destination address or hostname
  ip    IP echo
  ipv6  IPv6 echo
  vrf   VRF instance (source VRF)
  <cr>

awplus#ping vrf <name> ?
  WORD  Ping destination address or hostname
  ip    IP echo

awplus#ping vrf <name> x.x.x.x

awplus#ping vrf <name> x.x.x.x ?
  broadcast  Ping to a broadcast address
  df-bit     Enable do-not-fragment bit in IP header
  interval   Specify interval between pings
  pattern    Specify data pattern
  repeat     Specify repeat count
  size       Specify datagram size
  source     Specify source address or interface name
  timeout    Specify timeout interval
  tos        Specify type of service
  <cr>
```

Trace route

Example of configuration for trace route

```
awplus#traceroute ?
  WORD  Trace route to destination address or hostname
  ip    IP Trace
  ipv6  IPv6 trace
  vrf   VRF instance
  <cr>

awplus#traceroute vrf <name> ?
  WORD  Trace route to destination address or hostname
  ip    IP Trace

awplus#traceroute vrf <name> x.x.x.x
```

TCPdump

Example of configuration for TCPdump

```
awplus#tcpdump ?
  LINE  Execute tcpdump
  vrf   VRF instance
  <cr>

awplus#tcpdump vrf <name> ?
  LINE  Execute tcpdump
  <cr>

awplus#tcpdump vrf <name>
```

SSH client

Example of configuration for an SSH client

```
awplus#ssh ?
  HOSTNAME  IP/IPv6 address or hostname of a remote server
  client    Configure global SSH client parameters
  ip        IP SSH
  ipv6      IPv6 SSH
  port      SSH server port
  user      Login user
  version   SSH client version
  vrf       VRF instance

awplus#ssh vrf <name> ?
  HOSTNAME  IP/IPv6 address or hostname of a remote server
  ip        IP SSH
  port      SSH server port
  user      Login user
  version   SSH client version

awplus#ssh vrf <name> x.x.x.x
```

Telnet Client

Example of configuration for a Telnet client

```
awplus#telnet ?
  WORD  IPv4/IPv6 address or hostname of a remote system
  ip    IP telnet
  ipv6  IPv6 telnet
  vrf   VRF instance

awplus#telnet vrf <name> ?
  WORD  IPv4 address or hostname of a remote system
  ip    IP telnet

awplus#telnet vrf <name> ip x.x.x.x
```

SSH Server

Example of configuration for an SSH server

```
Rawplus(config)#ssh server ?
  <1-65535>             TCP port number
  allow-users           Add a user pattern to the Allow Users database
  authentication        User authentication
  deny-users            Add a user pattern to the Deny Users database
  disallow-cbc-ciphers  Disallow CBC ciphers to be used by SSH server
  login-timeout         Login grace time
  max-auth-tries        Maximum number of authentication attempts permitted
                        per connection
  max-startups          Maximum number of concurrent unauthenticated
                        connections
  resolve-hosts         Resolve host name from IP address using DNS server for
                        client host authentication
  scp                   Enable SCP service
  secure-algs           Use best current practice secure algorithms for
                        ciphers, KEX and MAC
  secure-ciphers        Use best current practice secure cipher list
  secure-kex            Use best current practice secure key exchange (kex)
                        list
  secure-mac            Use best current practice secure MAC list
  session-timeout       Session timeout
  sftp                  Enable SFTP service
  tcpforwarding         TCP forwarding
  v1v2                  Deprecated
  v2only                Deprecated
  vrf                   VRF instance

awplus(config)#ssh server vrf ?
  WORD  VRF instance name

awplus(config)#ssh server vrf <name>
```

System Log

Example of configuration for system log

```
awplus(config)#log host x.x.x.x ?
  exclude   Eliminate matched messages
  facility  Specify a syslog facility
  level     Set the minimum severity of message to accept in the log (0 -
            highest, 7 - lowest)
  msgtext   Select messages containing a certain text string
  program   Include messages from a specified program in the buffered log
  secure    Enable secure logging over TLS
  time      Specify the time offset of the remote syslog server from this
            device
  vrf       VRF instance
  <cr>

awplus(config)#log host x.x.x.x vrf ?
  WORD  VRF instance name

awplus(config)#log host x.x.x.x vrf <name> ?
  exclude   Elimite matched messages
  facility  Specify a syslog facility
  level     Set the minimum severity of message to accept in the log (0 -
            highest, 7 - lowest)
  msgtext   Select messages containing a certain text string
  program   Include messages from a specified program in the buffered log
  secure    Enable secure logging over TLS
  time      Specify the time offset of the remote syslog server from this
            device
  <cr>
```

File Copy

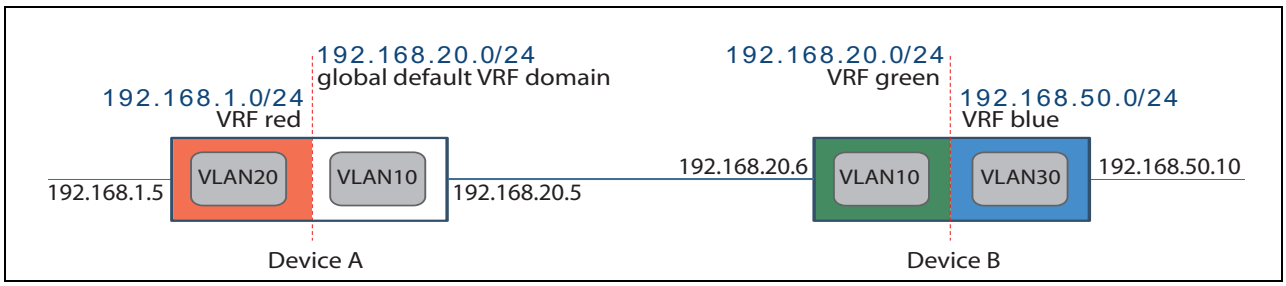
File Copy uses different underlying utilities depending on the method chosen:

Example of configuration for file copy

```
awplus#copy ?
  FILE                  URL of the Source file
  buffered-log          Copy from buffered log
  card                  Copy from SD card
  debug                 Copy from Debug
  fdb-radius-users      Copy from FDB in Radius user format
  flash                 Copy from flash memory
  force                 Overwrite destination file without prompt
  fserver               Copy from remote file server
  http                  Copy from HTTP source
  local-radius-user-db  Copy from Radius server user database
  nvs                   Copy from NVS memory
  permanent-log         Copy from buffered log
  running-config        Copy from current system configuration
  scp                   Copy from SCP source
  sftp                  Copy from SFTP source
  startup-config        Copy from startup configuration
  tftp                  Copy from TFTP source
  usb                   Copy from USB storage device
  zmodem                ZMODEM protocol
```

Static Inter-VRF routing

Whilst the prime purpose of VRF-lite is to keep routing domains separate from each other, there are cases where you do want some communication between VRFs. Static inter-VRF routing involves creating static routes in one VRF instance whose Layer 3 egress interface is in a different VRF instance. These static Inter-VRF routes must specify the egress Interface and may also require a next hop IP address to reach networks that are more than one hop away. The following figure illustrates use of static routing to achieve inter-VRF communication in VRF-lite.



DEVICE A STATIC ROUTES CONFIGURATION	DEVICE B STATIC ROUTES CONFIGURATION
ip route vrf red 192.168.20.0/24 vlan10 From source vrf red, create a static route to 192.168.20.0/24 to access target vlan10. Target VLAN is required when performing static IVR.	**ip route vrf blue 192.168.20.0/24 vlan10** From source vrf blue, create a static route to 192.168.20.0/24 to access target vlan10. Target VLAN is required when performing static IVR.
ip route 192.168.1.0/24 vlan20 From the source global VRF domain, create a static route to 192.168.1.0/24 to access target vlan20. Target VLAN is required when performing static IVR.	**ip route vrf green 192.168.50.0/24 vlan30** From the source vrf green, create a static route to 192.168.50.0/24 to access target vlan30. Target VLAN is required when performing static IVR.
ip route vrf red 192.168.50.0/24 192.168.20.6 vlan10 From source vrf red, create a static route to 192.168.50.0/24 with a next hop of 192.168.20.6 egressing target vlan10. Target VLAN is required when performing static IVR.	**ip route vrf blue 192.168.1.0/24 192.168.20.5 vlan10** From source vrf blue, create a static route to 192.168.1.0/24 with a next hop of 192.168.20.5 egressing target vlan10. Target VLAN is required when performing static IVR.
ip route 192.168.50.0/24 192.168.20.6 From the global VRF domain, create a static route to 192.168.50.0/24 with a next hop of 192.168.20.6. Static routes to networks within a VRF instance do not require target VLAN.	**ip route vrf green 192.168.1.0/24 192.168.20.5** From the source vrf green, create a static route to 192.168.1.0/24 with a next hop of 192.168.20.5. Static routes to networks within a VRF instance do not require the target VLAN.

Configuring DNS Relay to be VRF-aware

Introduction

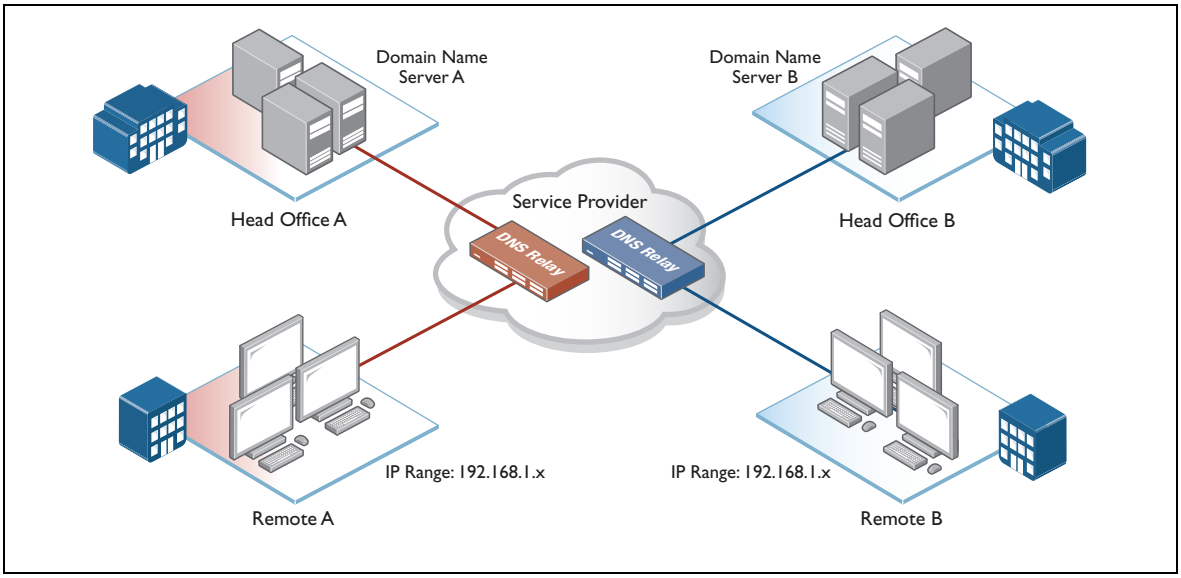
Domain Name System (DNS) is used to translate domain names into IP addresses. A DNS Relay (or proxy or forwarder) is an intermediate service that receives DNS queries from a client then forwards them to a DNS server on the client's behalf, and provides a local cache of responses. The optional DNS resolver cache provides some lookup speed advantage and avoids unnecessarily repeated requests to external DNS servers.

This section describes how to configure DNS Relay to be VRF-aware.

DNS operation with VRF-lite

When running VRF-lite, you can configure DNS Relay functionality to be VRF-aware. The switch can respond to domain name service lookup requests from external clients that are received via a VLAN interface that is associated with a VRF instance. Each VRF instance supports its own configurable set of DNS servers and DNS cache.

In a typical Service Provider setup, there may be two separate devices each connecting to two separate networks. Clients in each of the remote offices query their own DNS Relay within the Service Provider network, which in turn query their own Head Office DNS server. This can be expensive for the Service Provider as more equipment is needed to separate the networks.



By making DNS Relay VRF-aware, each network can now have its own DNS Relay that queries its own network's DNS Server and keeps its own DNS cache that is completely separate for the other networks, even if they have overlapping IP addresses.

Configuring DNS operation with VRF-lite

The command **ip name-server [vrf <name>] <ip-address>**, configures a name-server for the specified VRF. This command assigns the address of one or more name-servers to a VRF table to be used for name and address resolution. If no VRF-lite instance (**vrf <name>**) is specified, the name-server is configured for the global VRF. A name-server that is configured on the global VRF will apply to both the DNS relay and DNS client.

The DNS client can be made VRF-aware by directing all local DNS queries to the DNS Relay. To configure this use the commands **ip domain-lookup via-relay and ip dns forwarding**.

The configuration command **ip dns forwarding** applies:

A VRF specific name-cache is created within the DNS relay for every VRF instance that has a name-server configured. Once this has been configured VRF-aware client services that can utilize the DNS client such as ping, traceroute, telnet, and ssh will be able to perform DNS lookups when a VRF has been specified. For more information see the DNS Feature Overview and Configuration Guide.

A maximum of three name-servers may be defined for each DNS-relay instance.

The configuration command **ip dns forwarding** applies the DNS forwarding cache to all VRF instances configured on the device, however cached results are stored on a per VRF basis. That means if you have cached a lookup result learnt via VRF Red, then a subsequent lookup request from VRF Red will use that cached result, but a lookup request from VRF Green will not.

The configuration commands listed below apply to all VRF instances configured on the device and not on a per VRF basis. Timeouts are in seconds as per existing commands:

- `ip dns forwarding retry`
- `ip dns forwarding timeout`
- `ip dns forwarding dead-time`
- `ip dns forwarding source-interface`
- `ip dns forwarding cache`

The following **show** commands provide output information for the VRF instance specified. If a VRF instance is not specified, output is shown for all VRF instances including the global instance and the output will be formatted in a way that distinguishes the information for each VRF.

- `show ip dns [vrf <name>|global]forwarding server`
- `show ip dns [vrf <name>|global] forwarding cache`
- `show ip name-server [vrf <name>|global]`

The DNS cache can also be cleared on a per VRF instance basis by using the **clear ip dns [vrf <name>|global] forwarding** cache command.

The following commands show how to configure a DNS relay name-server for both the specified VRF instance VRF red, and the global VRF instance.

To configure a DNS relay name-server for the VRF-lite instance red:

```
awplus# configure terminal
awplus(config)# ip name-server vrf red 10.100.1.1
awplus# ip domain-lookup
```

To configure a DNS relay name-server for the global VRF instance:

```
awplus# configure terminal
awplus(config)# ip name-server 10.0.0.1
awplus# ip domain-lookup
```

To configure the IP DNS Forwarding cache:

```
awplus# ip dns forwarding
awplus# ip dns forwarding cache size 10 timeout 1800
```

Figure 20: Example extract from running configuration

```
!
ip name-server 10.0.0.1
ip name-server vrf red 10.100.1.1
ip domain-lookup
!
...
!
ip dns forwarding
ip dns forwarding cache size 10 timeout 1800
!
```

Figure 21: Example output from show ip name-server command

```
awplus#sh ip name-server
Currently learned name-servers:
10.0.0.1     static

[VRF: red]
10.100.1.1   static
```

Figure 22: Example output from show ip dns forwarding server

```
awplus#show ip dns forwarding server
Servers                                    Forwards    Fails Dead-Time
10.0.0.1                                          2        0    active

[VRF: red]
Servers                                    Forwards    Fails Dead-Time
10.100.1.1                                        2        0    active
```

Figure 23: Example output from show ip dns vrf <name> forwarding server

```
awplus#show ip dns vrf red forwarding server
[VRF: red]
Servers                                    Forwards    Fails Dead-Time
10.0.0.1                                          1        0    active
```

Figure 24: Example output from show ip dns global forwarding server

```
awplus#show ip dns global forwarding server
Servers                                    Forwards    Fails Dead-Time
10.100.1.1                                        1        0    active
```

Figure 25: Example output from show ip dns forwarding cache command

```
awplus#sh ip dns forwarding cache
Host
 Address                                       Expires Flags
www.example.com
 10.0.0.10                                        1793 REVERSE
www.example.com
 10.0.0.10                                        1793

[VRF: red]
Host
 Address                                       Expires Flags
Rwww.example_red.com
 10.100.1.10                                       645 REVERSE
www.example_red.com
 10.100.1.10                                       645
```

Configuring a DHCP server to be VRF aware

From version 5.5.2-1.1 onwards, you can configure a DHCP server to be VRF aware. This means you can associate a VRF with a DHCP address pool and (optionally) use the same DHCP lease across multiple isolated networks. You can configure DHCP pools with same or different network and address ranges associated with each.

The command: `vrf <vrf-name>` which allows you to add a VRF name to a DHCP server's address pool. For example, to add the VRF named 'red' to the DHCP address pool named 'red_pool', use the following commands:

```
awplus# configure terminal
awplus(config)# ip dhcp pool red_pool
awplus(dhcp-config)# vrf red
```

To remove the VRF named 'red' from 'red_pool' use the following commands:

```
awplus# configure terminal
awplus(config)# ip dhcp pool red_pool
```

From version 5.5.2-2.1 you can configure VRF in config mode and apply it to SMMP, NTP and sFlow.

Configure SNMP Server to only respond to requests from SNMP Managers residing within VRF 'red', use the commands:

```
awplus# configure terminal
awplus(config)# snmp server vrf red
```

Configure NTP to communicate with an NTP server residing within VRF 'red', use the commands:

```
awplus# configure terminal
awplus(config)# ntp server 10.0.0.1 vrf red
```

Configure the sFlow Agent to send samples to an sFlow Collector residing within VRF 'red', use the commands:

```
awplus# configure terminal
awplus(config)# sflow collector id 1 ip 10.0.0.1 vrf red
```

Example configuration

Here is a simple configuration for a DHCP VRF-aware server. This example shows how to configure VRF aware DHCP server address pools.

The **vrf** command must be added to the DHCP server pool before the **network** and **range** address commands. The VRF instance must exist before the VRF command is configured.

Note: When a new DHCP Lease Request is received by the server it will look up the VRF domain the request was learned on and assign a lease based off of valid pool(s) for the VRF. This also means the same pool address range can be used by multiple VRF instances.

.

```
...
ip dhcp pool blue_pool
 vrf blue
 network 192.168.4.0 255.255.255.0
 range 192.168.4.10 192.168.4.20
 host 192.168.4.3 aaaa.bbbb.cccc
!
ip dhcp pool green_pool
 vrf green
 network 192.168.4.0 255.255.255.0
 range 192.168.4.10 192.168.4.20
 host 192.168.4.3 aaaa.bbbb.cccc
 dns-server 192.168.2.1
 dns-server 192.168.3.1
!
server dhcp-server
!
ip vrf blue 1
!
ip vrf green 2
....
....
interface vlan6
 ip vrf forwarding green
 ip address 192.168.4.1/24
!
interface vlan7
 ip vrf forwarding blue
 ip address 192.168.4.1/24
!
```

Adding a VRF-aware static ARP

ARP entries associated with a given Layer 3 interface are cleared when the interface is moved from one VRF instance to another. In addition (static and dynamic) ARP entries are VRF-aware, as the same IP address can be used in other VRF instances.

Figure 26: Commands to configure static ARP entries

```
awplus(config)#arp ?  A.B.C.D  IP address of the ARP entry
  log      Arp log
  vrf      VRF instance

awplus(config)#arp vrf <name> ?
  A.B.C.D  IP address of the ARP entry
```

It is possible to show the whole ARP table, or just that section associated with a given VRF instance. The ARP table is divided into sections for each VRF instance.

Local interfaces with VRF-lite

A **local interface** (also often referred to as a **loopback interface**) is an internal interface that is always available for higher layer routing protocols to use and advertise to the network. Although a local interface is assigned an IP address, it does not have the usual requirement of connecting to a lower layer physical entity.

A local interface can be used as a reliable address via which to access a device—an address that is always accessible, irrespective of the link status of any individual external interface.

When a VRF-lite instance is created, a local interface is assigned to it either automatically, or by static assignment of a specific interface.

The following command creates a named VRF-lite instance with an optional local interface:

```
awplus(config)#ip vrf <vrf-name> [<lo>]
```

The optional local interface identifier <***lo***> is simply an integer number. This parameter specifies the number that will appear in the name of the resulting loopback interface. For example, if <***lo***> has the value 7, then the loopback interface that is created for the VRF-lite instance will have the name lo7.

If the optional local interface <***lo***> is **not** specified, a local interface will still be automatically created, and it will be assigned the next available ID number. Specifying the local interface allows the user to statically control which specific local interface ID number is associated with a particular VRF-lite instance.

Once a local interface is created, it remains assigned to the VRF (including over a reboot), unless manually changed by the user. Only a single local interface per VRF is supported.

For example, to create VRF-lite instance ‘red’ with statically assigned local interface ID 5, use the command:

```
awplus(config)#ip vrf red 5
```

Each local interface can be configured with its own IP address. For example, to assign an IP address to local interface lo5, use the commands:

```
awplus(config)# interface lo5
awplus(config-if)# ip address 192.168.200.1/32
```

Useful VRF-related diagnostics command list

You may find the following diagnostic commands helpful when troubleshooting VRF-related issues. For the complete list of VRF-aware commands, refer to the VRF-lite chapter in the Command Reference for your product.

General

Figure 27: Configuration example for general

```
awplus#show tech-support
awplus#show running-config
awplus#show running-config vrf
awplus#show system
awplus#show boot
awplus#show clock
```

VRF

Figure 28: Configuration example for general

```
awplus#show ip vrf
awplus#show ip vrf ?
  WORD       VRF instance name
  brief      Brief VRF instance information
  detail     Detailed VRF instance information
  interface  Interface information
  |          Output modifiers
  >          Output redirection
  >>         Output redirection (append)
  <cr>
awplus#show ip vrf interface
awplus#show ip vrf detail
```

Routing general

Figure 29: Configuration example for routing general

```
awplus#sh ip route
Codes: C - connected, S - static, R - RIP, B - BGP
       O - OSPF, D - DHCP, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       * - candidate default

S       0.0.0.0/0 [1/0] via 10.0.0.254, vlan1
C       10.0.0.0/24 is directly connected, vlan1

Gateway of last resort is not set

[VRF: red]
C       10.100.1.0/24 is directly connected, vlan2

Gateway of last resort is not set
No entries in route table
```

```
awplus#sh ip route vrf red
Codes: C - connected, S - static, R - RIP, B - BGP
       O - OSPF, D - DHCP, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       * - candidate default


[VRF: red]
C       10.100.1.0/24 is directly connected, vlan2

Gateway of last resort is not set
No entries in route table
```

ARP

Figure 30: Configuration example for ARP

```
awplus#sh arp
IP Address       MAC Address     Interface    Port         Type
10.0.0.254       0022.b06b.62d6  vlan1        port1.0.1    dynamic

[VRF: red]
IP Address       MAC Address     Interface    Port         Type
10.100.1.254     001a.eb94.2a44  vlan2        port1.0.2    dynamic
```

```
awplus#show arp vrf red

[VRF: red]
IP Address       MAC Address     Interface    Port         Type
10.100.1.254     001a.eb94.2a44  vlan2        port1.0.2    dynamic
```

TCPdump

Figure 31: Configuration example for TCPdump

```
awplus#tcpdump
awplus#tcpdump vrf <name>
```