



Configuring VIP and Virtual Interface Redundancy

This chapter describes how to plan for and configure virtual IP (VIP) redundancy and virtual interface redundancy on the CSS. Information in this chapter applies to all CSS models except where noted.

This chapter provides the following major sections:

- [Overview of CSS Redundancy](#)
- [Overview of VIP and Virtual Interface Redundancy](#)
- [VIP and Virtual Interface Redundancy Quick Start](#)
- [Configuring VIP and Virtual Interface Redundancy](#)
- [Displaying VIP and Virtual Interface Redundancy Configurations](#)

Overview of CSS Redundancy

Redundancy helps to ensure:

- High availability for your network applications
- Users do not experience long network delays or black holes due to a single point of failure.

A CSS provides three types of redundancy.

- Virtual IP (VIP) and virtual interface redundancy - Provides redundant VIP addresses and redundant virtual interfaces for fate sharing (the redundant interfaces and redundant VIPs fail over together to the backup CSS) and server default gateways. For details, see this chapter.
- Adaptive Session Redundancy (ASR) - Provides session-level redundancy (stateful failover) to continue active flows without interruption if the master CSS fails over to the backup CSS. For details, refer to [Chapter 7, Configuring Adaptive Session Redundancy](#).
- Box-to-box redundancy - Provides chassis-level redundancy between two identically configured CSSs. For details, refer to [Chapter 8, Configuring Box-to-Box Redundancy](#).

The following sections provide information about when and when not to use the different types of redundancy.

When to Use VIP and Virtual Interface Redundancy

Typically, you configure VIP redundancy on the public side of CSS peers that are positioned in front of a server farm. You configure virtual interface redundancy on the private-side interfaces attached to the Layer 2 device in front of the servers.

Configure VIP redundancy:

- With virtual interface redundancy to provide fate sharing
- When you have a common subnet between the two CSSs on which the VIPs reside
- As a prerequisite to configuring ASR (requires active-backup VIP redundancy)
- To provide active-active CSS behavior (both CSSs processing flows)

Configure interface redundancy:

- With VIP redundancy to provide fate sharing
- When you need a default gateway for the back-end servers
- Instead of VIP redundancy on the client side of the CSS when the VIPs are on a subnet different from the subnet of your uplinks

When to Use ASR

ASR provides session-level redundancy for applications where active flows (including TCP and UDP) must continue without interruption, even if the master CSS fails over to the backup CSS.

Configure ASR:

- If you require stateful failover for mission-critical applications (for example, enterprise applications; long-lived flows, such as HTTP or FTP file transfers; and e-commerce)
- After you have first configured active-backup VIP and virtual interface redundancy

When to Use Box-to-Box Redundancy

Configure box-to-box redundancy when you:

- Expect the behavior of the CSSs to be active/standby (only the master CSS processes flows)
- Can configure a dedicated Fast Ethernet (FE) link between the CSSs for the redundancy protocol

Do not configure box-to-box redundancy when you:

- Expect the behavior of the CSSs to be active-active (both CSSs processing flows). Use VIP redundancy instead.
- Cannot configure a dedicated FE link between the CSSs.

Overview of VIP and Virtual Interface Redundancy

This section provides information about:

- [VIP Redundancy](#)
- [Virtual Interface Redundancy](#)
- [Fate Sharing](#)
- [Examples of VIP and Virtual Interface Redundancy Configurations](#)

VIP Redundancy

When you configure a pair of CSSs to process client requests for the same VIP address, the VIP address is considered *redundant*. A typical use of VIP redundancy is with a virtual interface redundancy configuration where the master CSS processes all client requests to a VIP with a Web-server farm behind the CSSs and connected to the CSSs through a Layer 2 switch ([Figure 6-1](#)). If the master CSS becomes unavailable, the backup CSS becomes master and processes all client requests for the VIP.



Note

The CSS does not support VIP redundancy and box-to-box redundancy configurations simultaneously. For information about box-to-box redundancy, refer to [Chapter 8, Configuring Box-to-Box Redundancy](#).

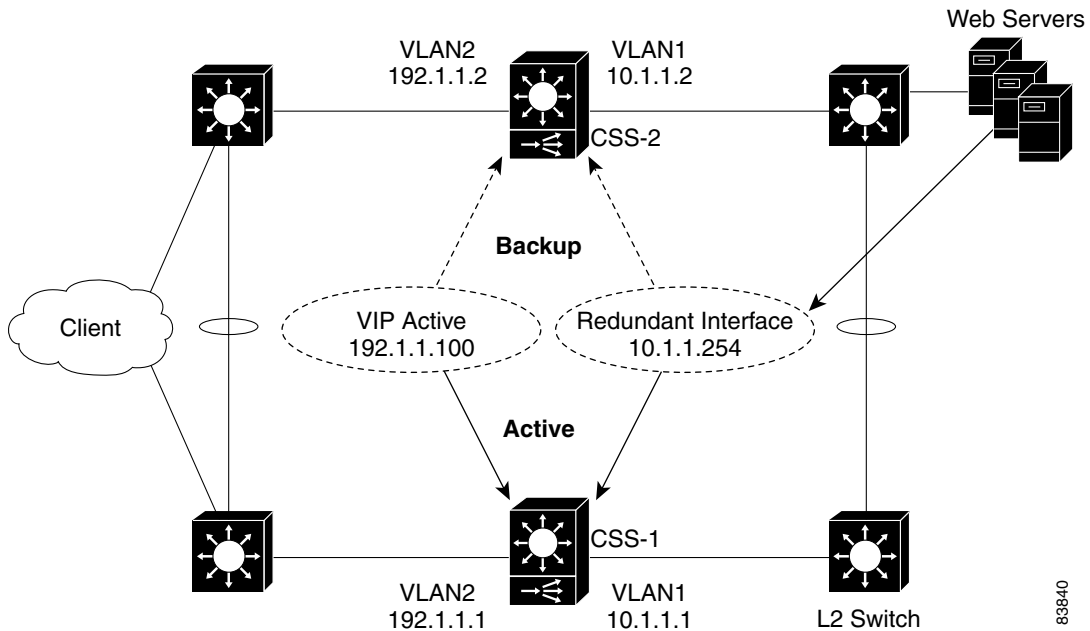
To set up CSSs for VIP redundancy, you must configure a virtual router on each CSS that will participate in the redundant configuration. A virtual router is an entity within a CSS to which you associate an existing VIP. A VIP becomes redundant when you associate it with a virtual router. You can configure a maximum of 255 virtual routers for each VLAN.



Note

The VIP address must already exist in at least one active content rule.

Figure 6-1 Example of VIP and Virtual Interface Redundancy



Virtual routers providing redundancy for a VIP address are considered *peers*. Each virtual router peer has the same virtual router identifier (VRID) and runs on the same VLAN, but runs on a different CSS. Once the virtual routers are configured, the CSSs negotiate for mastership using Virtual Router Redundancy Protocol (VRRP). A virtual router in a redundant VIP configuration that is designated as:

- Master will process all client requests directed to the VIP
- Backup may be either a:
 - Backup virtual router, which forwards all client requests directed to the VIP to the master CSS
 - Shared backup virtual router, which processes all client requests it receives and does not forward requests for the VIP to the master CSS

A CSS designated as the master of a VIP automatically sends a gratuitous ARP for the VIP when the CSS becomes the master, either at startup or upon failover. This process enables the Layer 2 switch to learn where to forward packets that are directed to the VIP from clients. The CSS transmits one ARP request packet and one ARP reply packet for every gratuitous ARP invocation.

For an example of VIP and virtual interface redundancy, see [Figure 6-1](#).

Virtual Interface Redundancy

Virtual interface redundancy is a form of IP address redundancy that applies only to IP interfaces (not VIPs). A typical interface IP address on a CSS defines the interface in use on a particular VLAN. In a virtual interface redundancy configuration, the CSS designated as master maintains control over the redundant virtual interface. Each CSS will also have its own circuit IP address that you can use for Telnet, SNMP, or the Device Management User Interface software.

The typical use for virtual interface redundancy is with a VIP redundancy configuration in which:

- Web servers are positioned behind a Layer 2 switch
- CSSs with the redundant virtual interface are positioned in front of the Layer 2 switch
- The servers are configured with a default route (gateway) pointing to the redundant virtual interface IP address

You must configure a virtual router with the same VRID on the two CSSs on the backside subnet that is common to both CSSs. This VRID must be different from the VRID configured for VIP redundancy. Once you associate the new VRID with the virtual redundant interface IP address, the CSSs uses VRRP to negotiate mastership of the virtual redundant interface.

A CSS designated as the master of a virtual interface automatically sends out gratuitous ARPs for the virtual interface's IP address when the CSS becomes the master, either at startup or upon failover. This process enables the Layer 2 switch to learn where to forward packets that are directed to the virtual interface from the servers and allows a server's default route to always point to the CSS designated as the master of the virtual interface. The CSS transmits one ARP request packet and one ARP reply packet for every gratuitous ARP invocation.

For an example of VIP and virtual interface redundancy, see [Figure 6-1](#).

**Note**

Virtual interface redundancy does not support a CSS configured as a *shared* backup.

You can also configure virtual interface redundancy on the uplinks of the CSSs when the VIPs reside on a subnet different from that of the uplinks. In this case, you cannot configure VIP redundancy on the public side of the CSSs. You will need to configure static routes on the upstream routers pointing to the redundant virtual interface on the CSS as the router's next hop gateway to the subnet where the VIPs reside.

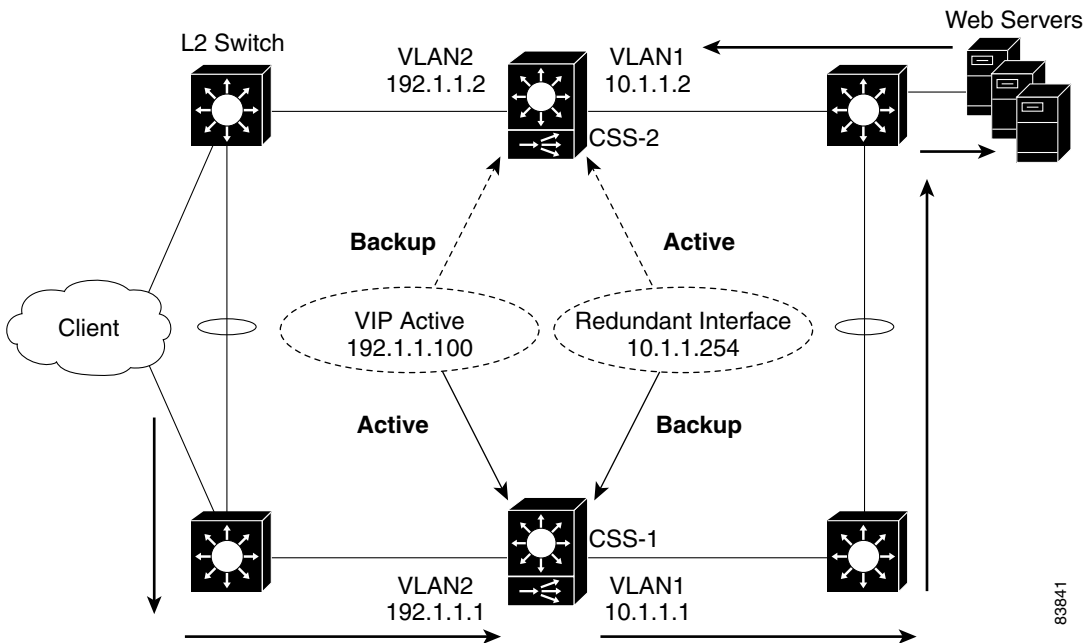
Fate Sharing

Fate sharing means that, when the redundant VIP fails over on the public side of the network from the master to the backup CSS, the redundant virtual interface on the private side of the network also fails over from the master to the backup CSS. If you do not configure virtual interface redundancy with VIP redundancy, asymmetric flows may result ([Figure 6-2](#)). Asymmetric flows occur when a CSS is master on the public side, but backup on the private side, which will break the connection between the client and the server.

To ensure that the redundant VIP and the redundant virtual interface fail over at the same time, you must bind the front and the back instances of VRRP (the virtual routers) so that the same CSS processes both inbound and outbound flows. You accomplish this by defining as critical services the IP addresses of the upstream router (the CSS default gateway) and the downstream Layer 2 switch that connects to the servers.

For example, the CSS provides a scripted keepalive (ap-kal-pinglist) that will check the health of the upstream router and the downstream Layer 2 switch. When you configure this keepalive, if either device fails, the critical service goes down and the VIP and the virtual interface fail over together to the backup CSS. For details on configuring critical services, see the [“Configuring Critical Services”](#) section.

Figure 6-2 Example of Asymmetric Flows Without Fate Sharing



83841

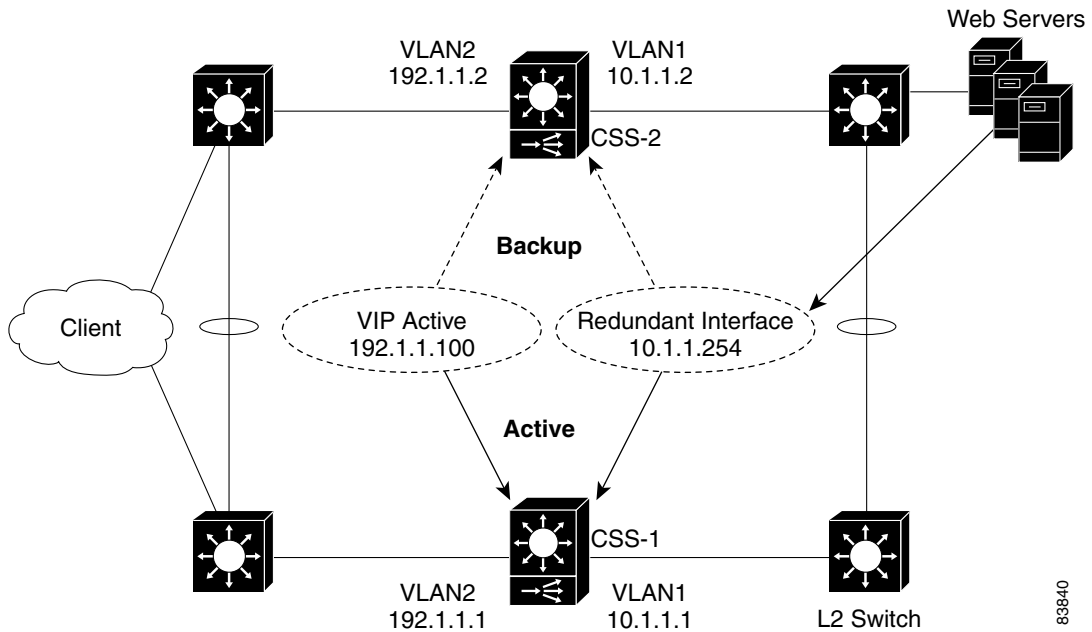
Examples of VIP and Virtual Interface Redundancy Configurations

The following sections provides examples of the most commonly used VIP and virtual interface redundancy configurations.

Active-Backup VIP and Virtual Interface Redundancy with Fate Sharing

Figure 6-3 shows an active-backup VIP and virtual interface redundancy configuration. CSS-1 is configured as the master for VIP address 192.1.1.100 and virtual interface address 10.1.1.254. If CSS-1 fails, CSS-2 (the backup CSS) will assume mastership of all flows destined to VIP address 192.1.1.100 and virtual interface address 10.1.1.254.

Figure 6-3 Example of Active-Backup VIP and Virtual Interface Redundancy



83840

CSS-1 Configuration

```

circuit VLAN1

ip address 10.1.1.1 255.255.255.0
ip virtual-router 1 priority 101 preempt
ip redundant-interface 1 10.1.1.254
ip critical-service 1 upstream_downstream
circuit VLAN2

ip address 192.1.1.1 255.255.255.0
ip virtual-router 2 priority 101 preempt
ip redundant-vip 2 192.1.1.100
ip critical-service 2 upstream_downstream

```

CSS-2 Configuration

```
circuit VLAN1

ip address 10.1.1.2 255.255.255.0
ip virtual-router 1
ip redundant-interface 1 10.1.1.254
ip critical-service 1 upstream_downstream

circuit VLAN2

ip address 192.1.1.2 255.255.255.0
ip virtual-router 2
ip redundant-vip 2 192.1.1.100
ip critical-service 2 upstream_downstream
```

Active-Active VIP and Virtual Interface Redundancy

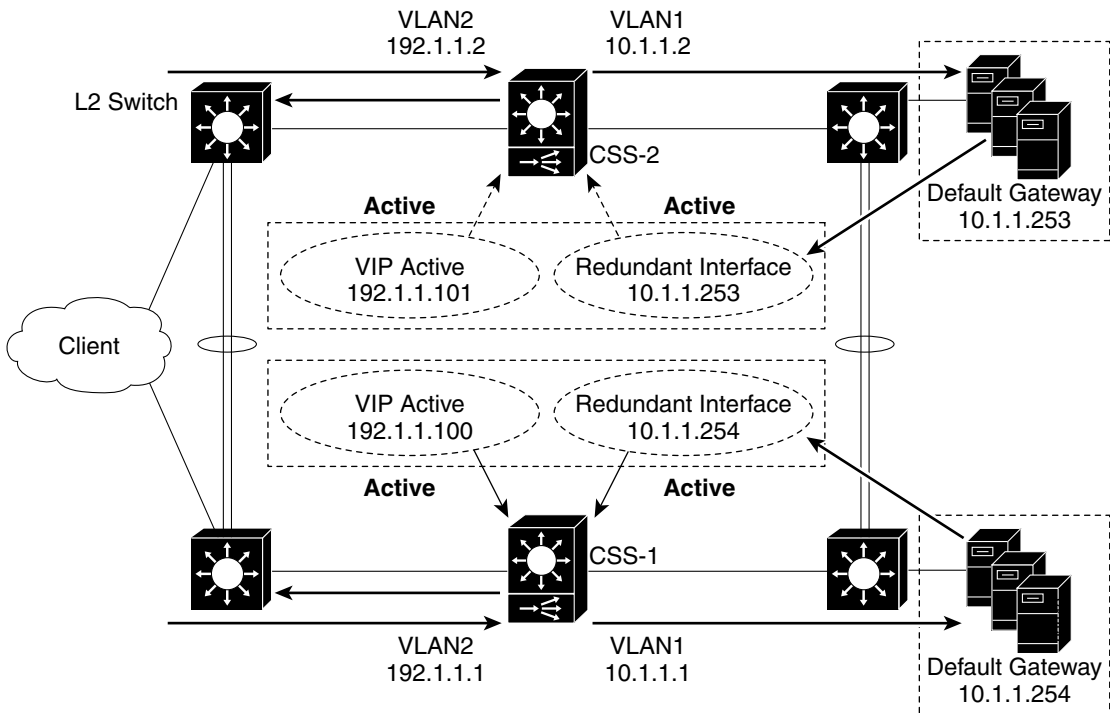
A CSS can serve simultaneously as a master to one virtual router and as a backup to a different virtual router. This is called active-active VIP and virtual interface redundancy. All redundant VIP addresses will share the state of the virtual router to which they are associated. The same virtual router cannot be active on both CSSs simultaneously.

Figure 6-4 shows an active-active VIP and virtual interface redundancy configuration with:

- CSS-1 configured as:
 - VLAN1 IP address 10.1.1.1.
 - VLAN2 IP address 192.1.1.1.
 - Master virtual router for VIP address 192.1.1.100 and virtual interface address 10.1.1.254.
 - Backup virtual router for VIP address 192.1.1.101 and virtual interface address 10.1.1.253. CSS-1 will forward all client requests it receives for VIP address 192.1.1.101 and all server requests for virtual interface address 10.1.1.253 to CSS-2.
- CSS-2 configured as:
 - VLAN1 IP address 10.1.1.2.
 - VLAN2 IP address 192.1.1.2

- Master virtual router for VIP address 192.1.1.101 and virtual interface address 10.1.1.253.
- Backup virtual router for VIP address 192.1.1.100 and virtual interface address 10.1.1.254. CSS-2 will forward all client requests it receives for VIP address 192.1.1.100 and all server requests for virtual interface address 10.1.1.254 to CSS-1.

Figure 6-4 Example of Active-Active VIP and Virtual Interface Redundancy



CSS-1 Configuration

```
circuit VLAN1

ip address 10.1.1.1 255.255.255.0
ip virtual-router 1 priority 101 preempt
ip virtual-router 2
ip redundant-interface 1 10.1.1.254
ip redundant-interface 2 10.1.1.253
ip critical-service 1 upstream_downstream
ip critical-service 2 upstream_downstream

circuit VLAN2

ip address 192.1.1.1 255.255.255.0
ip virtual-router 3 priority 101 preempt
ip virtual-router 4
ip redundant-vip 3 192.1.1.100
ip redundant-vip 4 192.1.1.101
ip critical-service 3 upstream_downstream
ip critical-service 4 upstream_downstream
```

CSS-2 Configuration

```
circuit VLAN1

ip address 10.1.1.2 255.255.255.0
ip virtual-router 1
ip virtual-router 2 priority 101 preempt
ip redundant-interface 1 10.1.1.254
ip redundant-interface 2 10.1.1.253
ip critical-service 1 upstream_downstream
ip critical-service 2 upstream_downstream

circuit VLAN2

ip address 192.1.1.2 255.255.255.0
ip virtual-router 3
ip virtual-router 4 priority 101 preempt
ip redundant-vip 3 192.1.1.100
ip redundant-vip 4 192.1.1.101
ip critical-service 3 upstream_downstream
ip critical-service 4 upstream_downstream
```

Shared VIP Redundancy

In a shared VIP redundancy configuration, both the master and the backup virtual routers process flows destined to the same VIP. However, only the master virtual router (CSS-1) responds to ARP requests for the VIP address 192.1.1.100.

A shared VIP redundancy configuration has the following requirements:

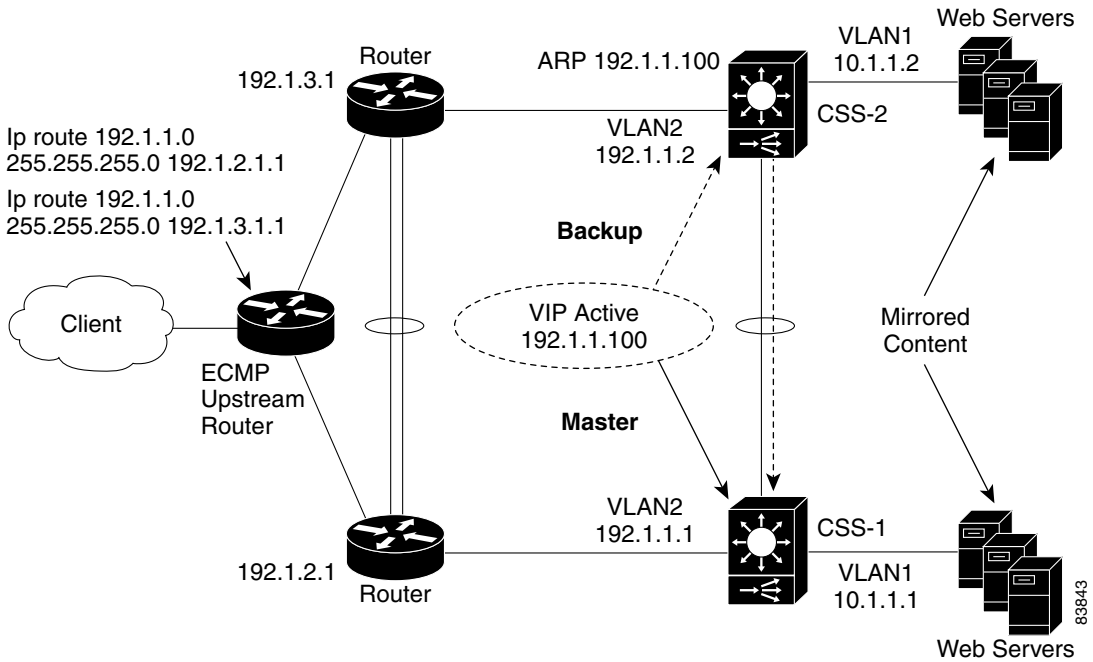
- Direct uplink connections to routers (no common Layer 2 connection between CSSs)
- Direct connection between the CSSs to enable the backup CSS to forward ARP requests to the master CSS
- Mirrored content on the servers
- Direct connection from the servers to the CSSs eliminating the need for fate sharing
- Session- or flow-based (not packet-by-packet) ECMP router upstream to preserve flow state

Figure 6-5 shows a shared VIP redundancy configuration with:

- CSS-1 configured as master virtual router for VIP address 192.1.1.100
- CSS-2 configured as shared backup for VIP address 192.1.1.100

Notice that CSS-2 (shared backup virtual router for VIP 192.1.1.100) forwards the ARP request for 192.1.1.100 to CSS-1 for a response.

Figure 6-5 Example of Shared VIP Redundancy



CSS-1 Configuration

```

circuit VLAN1

ip address 10.1.1.1 255.255.255.0

circuit VLAN2

ip address 192.1.1.1 255.255.255.0
ip virtual-router 1
ip redundant-vip 1 192.1.1.100 shared

```

CSS-2 Configuration

```
circuit VLAN1

ip address 10.1.1.2 255.255.255.0

circuit VLAN2

ip address 192.1.1.2 255.255.255.0
ip virtual-router 1
ip redundant-vip 1 192.1.1.100 shared
```

VIP and Virtual Interface Redundancy Quick Start

[Table 6-1](#) provides a quick overview of the steps required to configure active-backup VIP and virtual interface redundancy with fate sharing for CSS-1. Each step includes the CLI command required to complete the task. For a complete description of each feature and all the options associated with the CLI command, see the sections following [Table 6-1](#).

Table 6-1 *VIP and Virtual Interface Redundancy Configuration Quick Start*

Task and Command Example

1. Enter config mode.

```
# config
(config)#
```

2. Enter circuit mode for VLAN1.

```
(config)# circuit VLAN1
(config-circuit[VLAN1])#
```

3. Configure a circuit IP address.

```
(config-circuit[VLAN1])# ip address 10.1.1.1/24
(config-circuit-ip[VLAN1-10.1.1.1])#
```

Table 6-1 *VIP and Virtual Interface Redundancy Configuration Quick Start (continued)*

Task and Command Example
<p>4. Configure the virtual router. If you do not have a preference as to which router becomes master, you may leave the default priority at 100. If you have a preference, assign a higher priority to one router using the priority option. When you want a virtual router to assume mastership in all circumstances, include the preempt keyword.</p> <pre>(config-circuit-ip[VLAN1-10.1.1.1])# ip virtual-router 1 priority 101 preempt</pre>
<p>5. Configure the redundant virtual interface.</p> <pre>(config-circuit-ip[VLAN1-10.1.1.1])# ip redundant-interface 1 10.1.1.254</pre>
<p>6. Configure the critical service for the virtual router.</p> <pre>(config-circuit-ip[VLAN1-10.1.1.1])# ip critical-service 1 upstream_downstream</pre>
<p>7. Enter circuit mode for the next desired circuit VLAN.</p> <pre>(config)# circuit VLAN2 (config-circuit[VLAN2])#</pre>
<p>8. Configure a circuit IP address.</p> <pre>(config-circuit[VLAN2])# ip address 192.1.1.1/24 (config-circuit-ip[VLAN2-192.1.1.1])#</pre>
<p>9. Configure the virtual router.</p> <pre>(config-circuit-ip[VLAN2-192.1.1.1])# ip virtual-router 2 priority 101 preempt</pre>
<p>10. Configure the redundant VIP on the virtual router.</p> <pre>(config-circuit-ip[VLAN2-192.1.1.1])# ip redundant-vip 2 192.1.1.100</pre>
<p>11. Configure the critical service for the virtual router.</p> <pre>(config-circuit-ip[VLAN2-192.1.1.1])# ip critical-service 2 upstream_downstream</pre>
<p>12. Display the configuration (optional).</p> <pre>(config)# show virtual-routers</pre>

You would configure CSS-2 in a similar manner, with the exception of the **priority** and **preempt** options of the **ip virtual-router** command.

Because this chapter is dedicated to configuring VIP and virtual interface redundancy, it contains only those circuit IP commands that pertain to this feature. For a complete description of all circuit IP commands, refer to the *Cisco Content Services Switch Administration Guide*.

Configuring VIP and Virtual Interface Redundancy

You must configure each CSS that is part of a redundant oconfiguration. The following sections describe how to configure VIP and virtual interface redundancy.

- [Configuring a Circuit IP Interface](#)
- [Configuring a Virtual Router](#)
- [Configuring a Redundant VIP](#)
- [Configuring Critical Services](#)
- [Configuring a Redundant Virtual Interface](#)
- [Synchronizing a VIP Redundancy Configuration](#)

Configuring a Circuit IP Interface

Before you can configure VIP and virtual interface redundancy, you must configure a circuit IP interface and assign it an IP address. To enter a specific circuit configuration mode, enter the **circuit** command and VLAN as shown in the following example:

```
(config)# circuit VLAN2  
(config-circuit[VLAN2])#
```



Note

When you use the **circuit** command, enter the word “VLAN” in uppercase letters and *do not* include a space between VLAN and the VLAN number (for example, VLAN1).

To assign an IP address to a circuit, use the **ip address** command from the specific circuit mode. Enter the IP address and a subnet mask in CIDR bitcount notation or dotted-decimal notation. The subnet mask range is /8 to /32. For example, to configure an IP address and subnet mask for VLAN1, enter:

```
(config-circuit[VLAN2])# ip address 192.1.1.1 /24
```

When you specify an IP address, the mode changes to the specific circuit-ip-VLAN-IP address as shown:

```
(config-circuit-ip[VLAN2-192.1.1.1])#
```

Configuring a Virtual Router

Use the **ip virtual-router** command to create a virtual router on a CSS and configure its identifier and priority that is used when negotiating control of associated VIPs. You must configure the virtual router before you can configure redundant VIPs.

A virtual router's role as a master or backup is determined during negotiations between all virtual routers with the same VRID and residing on the same VLAN.

The syntax and options for the IP interface command are:

```
ip virtual-router vrid {priority number} {preempt}
```

The variables and options are:

- *vrid* - The virtual router identifier (VRID). Enter an integer between 1 and 255. You can configure 255 virtual routers per VLAN. Virtual routers are considered peers when they have the same VRID and reside on the same VLAN.
- **priority number** - The optional priority of the virtual router with its peer. The default priority value is 100. Enter an integer between 1 and 255. A virtual router with the highest priority usually becomes master. However, a higher priority virtual router will not assume mastership from a lower priority master unless you include the **preempt** option.

When a virtual router is the master, it handles the traffic directed to its associated VIPs. To set a virtual router so that it will always be master, set its priority to 255 and configure it with the **preempt** option.

- **preempt** - The optional keyword that allows a higher priority virtual router to assert mastership over a lower priority virtual router. By default, a virtual router does not become master if the current master has a lower priority.

For example, if a CSS with a virtual router that has a low priority boots before other CSSs, that virtual router becomes the master. When another CSS with a virtual router that has a higher priority boots, it will not take the mastership from the first router unless you specify the **preempt** option on the higher priority virtual router. This option does not have an effect if the priority of the two virtual routers is identical. You can use this option with or without the **priority** option. You can configure only one virtual router as the master of a particular VIP.

**Caution**

Never configure the **preempt** option on the same virtual router on both CSSs. Such a configuration may result in both CSSs becoming master, which will cause network problems.

Because a virtual router's priority is dependent on the state of the critical services, the priority field status in the **show virtual router** display may be different than the priority you configured. The priority may be different when you:

- Assign a priority of 255 to a virtual router and that virtual router gains mastership, the CSS automatically reconfigures that virtual router's priority to 254. This action ensures that you can assign a different virtual router a priority of 255.
- Configure critical services. The critical service types are:
 - **scripted** - the priority changes to 0 when one service in the scripted group goes down.
 - **redundancy uplink** - the priority changes to 0 when all of the services in the uplink group go down.
 - **local** - the priority changes to 0 when all of the services in the local group go down. Local services include all services other than scripted and uplink.

For information about configuring critical services, see the [“Configuring Critical Services”](#) section.

For example:

```
(config-circuit-ip[VLAN2-192.1.1.1])# ip virtual-router 2 priority  
1 preempt
```

To remove the virtual router from the CSS, enter:

```
(config-circuit-ip[VLAN2-192.1.1.1])# no ip virtual-router 2
```

Configuring a Redundant VIP

Use the **ip redundant-vip** command to associate an existing VIP to a virtual router and if required, configure the virtual router as a shared backup. A shared backup virtual router processes client requests. A redundant VIP configuration can consist of only two CSSs.



Note

Before you use this command, the VIP must already be configured in at least one active content rule. Additionally, if you defined the content rule VIP using the range option, you must configure an identical range for the redundant VIP. For information about configuring VIPs in content rules, refer to the *Cisco Content Services Switch Basic Configuration Guide*.

The syntax for this IP mode command is:

```
ip redundant-vip vrid vip_address {range number} {shared}
```

The variables and options are:

- *vrid* - The ID for an existing virtual router.
- *vip_address* - The address for the redundant VIP. This address must already be configured in at least one active content rule. Enter an IP address in dotted-decimal notation (for example, 192.1.1.100).
- **range number** - The optional keyword and variable if an IP address range is specified in the content rule. You cannot specify a range that differs from the range in the content rule. Also, you cannot specify address ranges that overlap. Enter a number from 0 to 65535. The default is 1.
- **shared** - The optional keyword to enable shared VIP redundancy. When you use this option, the master and backup virtual routers share the processing of traffic directed to the VIP, so the backup does not forward packets to the master. Configure each VIP identically on both CSSs.



Caution

Do not connect Layer 2 devices between the CSSs and the routers in a shared VIP redundancy configuration. In addition, each router must be connected to only one CSS. Otherwise, all traffic will go to the master CSS, thus defeating the purpose of shared VIP redundancy.

For example:

```
(config-circuit-ip[VLAN2-192.1.1.1])# ip redundant-vip 2
192.1.1.100 range 10 shared
```

To remove a VIP from a virtual router, enter:

```
(config-circuit-ip[VLAN1-192.1.1.1])# no ip redundant-vip 1
192.1.1.100
```

Configuring a Redundant Virtual Interface

Use the **ip redundant-interface** command to configure a redundant virtual interface to be used as the default route for backend servers. Servers use the IP address of the virtual interface as a default route to guarantee that packets will be sent to the CSS containing the master virtual router. Configure a redundant interface with the same virtual router of a redundant VIP that has a rule that refers to the server. This configuration ensures that the master CSS for a VIP is the same CSS that is master for the redundant virtual interface. This configuration is called *fate sharing*. If the master CSS fails over to the backup CSS, both the VIP and the redundant interface fail over together.

The syntax for this IP mode command is:

```
ip redundant-interface vrid ip_address
```

The variables are:

- *vrid* - ID for a previously-configured virtual router.
- *ip_address* - Address for the redundant interface. Enter an IP address in dotted-decimal notation (for example, 10.1.1.254).



Note

You cannot use an IP address that already exists for a VIP, redundant VIP, source group, service, log host, or IP interface address on a circuit. If you do, the following error message appears: *Address conflicts with local I/F, VIP, service, or sourcegroup.*

- **dns-server** - Keyword that enables the CSS to respond to DNS queries destined for the redundant interface IP address. For more information, see the [“Displaying VIP and Virtual Interface Redundancy Configurations”](#) section.

For example:

```
(config-circuit-ip[VLAN1-10.1.1.1])# ip redundant-interface 1
10.1.1.254
```

To remove an interface from a virtual router, enter:

```
(config-circuit-ip[VLAN1-10.1.1.1])# no ip redundant-interface 1
10.1.1.254
```



Note

The CSS does not support a traceroute of a redundant IP interface.

Configuring Critical Services

Use the **ip critical-service** command to associate a critical service with a virtual router. When a critical service goes down, the associated virtual router also goes down. There are three types of critical services that you can configure:

- A scripted critical service, as defined by the **(config-service) keepalive type script** command or the **(config-service) keepalive type named** command, that is constantly scanning for service and network availability. The keepalive sets the service to a down state whenever network or service availability is a problem. The virtual router goes down if *any* associated scripted service goes down.

The CSS provides a scripted keepalive called **ap-kal-pinglist** that you can use to check the health of, for example, an upstream router (192.1.1.254) running Hot Standby Router Protocol (HSRP) and a downstream Layer 2 switch (10.1.1.200) as critical services.

For example, create the service as follows:

```
(config)# service upstream_downstream
(config-service[upstream_downstream])# ip address 192.1.1.254
(config-service[upstream_downstream])# keepalive type script
ap-kal-pinglist "192.1.1.254 10.1.1.200"
(config-service[upstream_downstream])# active
```

- A redundancy uplink critical service, as defined by the **(config-service) type redundancy-up** command. The virtual router goes down when *all* associated redundancy uplink services go down regardless of any configured keepalive type. Refer to [Chapter 8, Configuring Box-to-Box Redundancy](#), in the “Configuring Multiple Redundant Uplink Services” section.



Note You cannot add redundant uplink services to a content rule.

- Local critical services for any service other than scripted or redundancy uplink, such as a Web service. The virtual router goes down when *all* associated local critical services go down.

The syntax for the **ip critical-service** command is:

```
ip critical-service vrid service_name
```

The variables are:

- *vrid* - The ID for an existing virtual router.
- *service_name* - The name of the service. To see a list of services, enter **ip critical-service vrid ?**.

For example:

```
(config-circuit-ip[VLAN2-192.1.1.1])# ip critical-service 1  
upstream_downstream
```

To remove a critical service from a virtual router, enter:

```
(config-circuit-ip[VLAN2-192.1.1.1])# no ip critical-service 1  
upstream_downstream
```



Note

If you configure different critical services on the two CSSs and you intend to synchronize the CSS configurations using the `commit_VipRedundConfig` script, do not use the `-a` script argument. This argument copies the master CSS configuration to the backup CSS configuration, which makes the two configs identical. For details on synchronizing a VIP and virtual interface redundancy configuration, see the [“Synchronizing a VIP Redundancy Configuration”](#) section.

The **show service** command displays the current service type only. It does, however, display the keepalive type, so you can determine from it the behavior of a configured critical service. To display critical service-specific information, use the **show critical-services** command. See the [“Displaying IP Critical Services”](#) section.

SNMP values returned for services show the current service type only. To determine the critical service behavior of a particular service, you need to examine the service keepalive type. For more information about SNMP, refer to the *Cisco Content Services Switch Administration Guide*.

Synchronizing a VIP Redundancy Configuration

To ensure that your backup CSS can perform the same tasks as your master CSS in the event of a master CSS failure, the running-config on the backup must be identical (with some modifications) to the running-config on the master. To automate this configuration synchronization process, you can run the **commit_VipRedundConfig** script on the master CSS to copy the master CSS running-config to the backup CSS running-config.

There are two types of configuration synchronization:

- **Complete** - On CSSs that have an identical chassis (the same CSS model), produces a running-config on the backup CSS that exactly matches the running-config on the master CSS.
- **Partial** (default) - On CSSs with incompatible configurations, synchronizes all parameter values in the configuration except the interface and circuit configurations. For example, the master is a CSS 11506 and the backup is a CSS 11503. The script maintains the current backup interface and circuit configurations automatically.

Script Functions

The configuration synchronization script performs all the necessary steps to update the backup CSS with the master's running configuration. The script:

- Saves the master running-config to the startup-config
- Archives the startup-config
- Copies the startup-config to a temporary file (tmp.cfg)
- Calls a function that converts the master VRRP/APP IP addresses to the backup VRRP/APP IP addresses in tmp.cfg
- Changes all VRID priorities on the master to 254
- Changes all VRID priorities in the backup's new config to 1

- Removes all preempt configs from all VRIDs in the backup's new config
- Uses the **rcmd** command to:
 - Copy tmp.cfg to a temp file on the backup (newconfig)
 - Check newconfig and copy it to the startup-config
 - Clear the backup CSS's running-config and script play newconfig

The script performs some verifications before executing the above steps. It checks to see if the local switch is a backup for any VRIDs and asks you if you want to continue, thereby changing the state on the two CSSs. The script also checks the backup to see if it is the master for any VRIDs. If the state is Interface (IF) Down, the script asks you if you want to continue without synchronizing those VRIDs on interfaces that are Down.

Before You Begin

Before you run the configuration synchronization script, ensure that you have configured VIP/interface redundancy and the Application Peering Protocol (APP). For details on configuring VIP/interface redundancy, see the [“Configuring VIP and Virtual Interface Redundancy”](#) section. For details on configuring APP, refer to [Chapter 1, Configuring the CSS Domain Name Service](#) in the [“Configuring the Application Peering Protocol”](#) section.

The synchronization script does not support the following configurations:

- Active/active shared VIP
- Any configuration where some independent VIP addresses are a master while other VIP addresses are a backup

Running the Configuration Synchronization Script

To run the configuration synchronization script, use the **script play commit_vip_redundancy** command in SuperUser mode. The syntax is:

```
script play commit_vip_redundancy “arguments”
```

You can also run the configuration synchronization script using the predefined alias that comes with all CSSs by entering:

```
# commit_VipRedundConfig “arguments”
```

You can specify the script arguments in any order. The arguments for the `commit_vip_redundancy` script are:

- *ip address* - The IP addresses of the master and backup APP sessions. This is the only required argument for this script. Use the following syntax when entering the addresses:

“local master IP address remote backup IP address”

For details on automating the entry of the IP address, see [“Setting the BACKUP_VIPR_IP Variable”](#) later in this section.

- **-a** (All) - Synchronizes the configuration completely. Use this argument only when the master and backup CSSs have identical chassis. This argument synchronizes the entire configuration and the interface mode. Do *not* use this argument if you have different critical services configured on the two CSSs.
- **-d** (Debug) - Debug switch for the `commit_vip_redundancy` script, which displays the current task being performed as the script progresses. Debug messages display even when you specify the **-s** argument.



Caution

Before you use the **-f** argument to remove a config sync lock file, ensure that no one else is running the config sync script on the CSS. Otherwise, if you remove the lock file and then run the script again while the script is in use, the resulting configurations may have some discrepancies.

- **-f** - After an abnormal script termination, removes the lock file so that you can run the script again. This argument overrides all other specified arguments and the script exits immediately after removing the lock file. For details on the lock file, see [“Setting the BACKUP_VIPR_IP Variable”](#) later in this section.
- **-nv** (No Verify) - Informs the script not to verify that the configuration synchronization was successful. However, the script does inform you if the script fails.



Note By default, the script verifies the configuration synchronization.

- **-s** (Silent) - Suppresses script progress messages and displays only the result of running the script: Commit Successful or Commit Failed. The **-d** argument overrides the **-s** argument.

For example, on the master CSS, run the following script, which uses the defaults of verify on and partial synchronization, plus the IP addresses set as variables and the script alias name:

```
CSS11506# commit_VipRedundConfig
```

The following output appears:

```
CSS11506# commit_VipRedundConfig
Verifying app and redundancy configs ...
Checking vip redundancy state ...
Working \
Verifying running-config copy success ...
Commit successful!
```

In this example, the script:

- Performs a partial configuration synchronization (default)
- Verifies that the configuration synchronization was successful (default)

For more information about scripts, refer to [Chapter 12, Using the CSS Scripting Language](#).

Config Sync Lock File

When you run the script, the software creates a lock file (vipr_config_sync_lock) in the script directory so that you cannot run the script from another session on the CSS. If the lock file exists and you run the script, the following message appears:

```
The script is in use by another session.
```

If the script terminates abnormally, the software does not remove the lock file. The next time you run the script, the above message appears. If you are certain that the script is not in use by another session, use the -f argument to remove the lock file. When you run the script with this argument, the following message appears and the script exits:

```
VIPR Config Sync lock file removed.
```

Now you can run the script again.

Setting the BACKUP_VIPR_IP Variable

To eliminate the need to specify IP addresses each time you run the configuration synchronization script, you can set the value of two variables (BACKUP_VIPR_IP and MASTER_VIPR_IP) to IP addresses and save them in your user profile. Once you set the variables and save them in your user profile, the variables will always be available after you log in to the CSS.

To set the variables, enter:

```
# set BACKUP_VIPR_IP "ip_address" {session}
# set MASTER_VIPR_IP "ip_address" {session}
```

where *ip_address* is the IP address of the backup or master CSS.

To save the variable in your user profile, enter:

```
# copy profile user-profile
```

Now you can run the configuration synchronization script without typing an IP address.

Logging Configuration Synchronization Script Result Messages

You can specify that script result messages (script success or failure messages) be sent to the current logging device automatically each time you run the configuration synchronization script. To log the script result messages, enable logging on NETMAN with level info-6 or debug-7 by entering:

```
(config)# logging subsystem netman level info-6
```



Note

Log messages are generated with or without the -s (silent) argument specified. See the [“Running the Configuration Synchronization Script”](#) section.

For example, if the APP session to the backup CSS is not running, the CSS generates the following log message:

```
vipr config sync: app session is DOWN
```

For ease of tracking, each log message contains the string “vipr config sync”.

Displaying VIP and Virtual Interface Redundancy Configurations

The CSS provides **show** commands to enable you to display redundant VIP and virtual interface configurations. The following sections describe the commands and provide tables describing the output fields.

- [Displaying IP Critical Services](#)
- [Displaying Redundant Virtual Interfaces](#)
- [Displaying Redundant VIPs](#)
- [Displaying Virtual Router Configurations](#)

Displaying IP Critical Services

Use the **show critical-services** command to display a list of all critical services configured on the CSS. You can provide an interface IP address option to display only the critical services present on a specific interface. You may also include a VRID to display only the critical service information for a specific virtual router.

The syntax for this command is:

```
show critical-services {ip_address {vrid}}
```

The optional variables are:

- *ip_address* - The address for the redundant interface. Enter an IP address in dotted-decimal notation (for example, 192.168.11.1).
- *vrid* - The ID for an existing virtual router.

For example, to view all critical services on the CSS, enter:

```
# show critical-services
```

[Table 6-2](#) describes the fields for the **show critical-services** command output.

Table 6-2 *Field Descriptions for the show critical-services Command*

Field	Description
Interface Address	The IP interface address associated with the virtual router.
VRID	The assigned identifier associated with the virtual router.
Service Name	The name of the critical service.
Service Type	The type of critical service. Possible critical service types are: <ul style="list-style-type: none"> • Scripted - A service whose state depends upon a running script or a named keepalive. • Redundancy-up - A service whose state depends upon the state of an ICMP keepalive on a router. • Local - Every type of service other than a scripted service or a redundancy uplink service. Typically, this is a Web server.
Service State	The current state of the critical service. The State field displays the service as either Alive, Dying, Down, or Suspended. The Dying state reports that a service is failing according to the parameters configured in the following service mode commands: keepalive retryperiod , keepalive frequency , and keepalive maxfailure . When a service enters the Down state, the CSS does not forward any new connections to it (the service is removed from the load balancing rotation for the content rule). However, the CSS keeps all existing connections to the service (connections to that service are not torn down).

Displaying Redundant Virtual Interfaces

Use the **show redundant-interfaces** command to display a list of all redundant virtual interfaces configured on the CSS. This command also displays the status (Enable or Disable) of the DNS server (if configured) on the virtual interface and the number of DNS packets processed by the interface. You may provide an interface IP address option to display only the virtual interfaces present on a particular interface. You may also include a VRID to display only the virtual interface information for a particular virtual router.

The syntax for this command is:

```
show redundant-interfaces {ip_address {vrid}}
```

The optional variables are:

- *ip_address* - The address for the redundant interface. Enter an IP address in dotted-decimal notation (for example, 192.168.11.1).
- *vrid* - The ID for an existing virtual router.

For example, to view all redundant interfaces on the CSS, enter:

```
(config) # show redundant-interfaces
```

[Table 6-3](#) describes the fields in the **show redundant-interfaces** command output.

Table 6-3 *Field Descriptions for the show redundant-interfaces Command*

Field	Description
Interface Address	IP interface address associated with the redundant virtual interface.
VRID	Assigned identifier associated with the virtual router.
Redundant Address	IP address of the redundant virtual interface.
Range	Not applicable. This field is always set to 1.
State	Current state of the virtual interface. Possible states are: <ul style="list-style-type: none"> • Master - The virtual interface is the master • Backup - The virtual interface is the backup • Idle - The virtual interface is

Table 6-3 *Field Descriptions for the show redundant-interfaces Command (continued)*

Field	Description
Master IP	IP address of the master virtual router.
State Changes	Number of times the redundant virtual interface state has changed.
Last Change	Date and time of the redundant virtual interface state last state change.
DNS Server	Status of the DNS server configured on the redundant interface. Possible states are Enable or Disable.
DNS Packets Processed	Number of DNS request packets that the redundant interface has processed.

Displaying Redundant VIPs

Use the **show redundant-vips** command to display a list of all redundant VIPs configured on the CSS. You could provide an interface IP address option to display only the VIPs present on a particular interface. You can also include a VRID to display only the VIP information for a particular virtual router.

The syntax for this command is:

```
show redundant-vips {ip_address {vrid}}
```

The optional variables are:

- *ip_address* - The address for the redundant interface. Enter an IP address in dotted-decimal notation (for example, 192.168.11.1).
- *vrid* - The ID for an existing virtual router.

For example, to view all redundant VIPs on the CSS, enter:

```
(config)# show redundant-vips
```

[Table 6-4](#) describes the fields in the **show redundant-vips** command output.

Table 6-4 Field Descriptions for *show redundant-vips* Command

Field	Description
Interface Address	The IP interface address associated with the redundant VIP.
VRID	The assigned identifier associated with the virtual router.
Redundant Address	The IP address of the VIP.
Range	The range associated with the VIP.
State	Current state of the redundant VIP. Possible states are: <ul style="list-style-type: none"> • Master - The redundant VIP is the master • Backup - The redundant VIP is the backup • Backup Shared - The redundant VIP is a shared backup • Idle - The redundant VIP is idle
Master IP	The IP address of the master virtual router.
State Changes	The number of times the VIP state has changed.
Last Change	The data and time of the VIP last state change.

Displaying Virtual Router Configurations

Use the **show virtual-routers** command to display a list of all virtual routers configured on the CSS. You may provide an interface IP address option to display only the virtual routers present on a particular interface. You may also include a VRID to display only the information for a particular virtual router.

The syntax for this command is:

```
show virtual-routers {ip_address {vrid}}
```

The optional variables are:

- *ip_address* - The address for the redundant interface. Enter an IP address in dotted-decimal notation (for example, 192.168.11.1).
- *vrid* - The ID for an existing virtual router.

For example, to view all virtual routers on the CSS, enter:

```
(config)# show virtual-routers
```

Table 6-5 describes the fields in the **show virtual-routers** command output.

Table 6-5 Field Descriptions for the show virtual-routers Command

Field	Description
Interface Address	The IP interface address associated with the virtual router.
VRID	The assigned identifier associated with the virtual router.
Priority	The priority currently being advertised by the virtual router. Because the priority is dependent on the state of the critical services, the priority may be different than the one configured.
Config. Priority	The configured priority.
State	Current state of the virtual router. Possible states are: <ul style="list-style-type: none"> • Master - The virtual router is master. • Backup - The virtual router is backup. • No Service - One or more critical services associated with the virtual router is down. • IF Down - The IP interface associated with the virtual router is down. • Idle - The virtual router does not have any virtual interfaces or VIPs associated with it.
Master IP	The IP address of the master virtual router.
State Changes	The number of times the virtual router state has changed.
Last Change	The data and time of the virtual router last state change.

Table 6-5 *Field Descriptions for the show virtual-routers Command (continued)*

Field	Description
Preempt	True if preemption is enabled for the virtual router; false otherwise.
Critical-Services	The names of the critical services.
State	The current condition of the critical service. Possible states are: Alive, Down, or Suspended.
Type	The type of critical service. Possible types are: Scripted, RedundancyUp, or Local.

■ **Displaying VIP and Virtual Interface Redundancy Configurations**