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

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 2, Configuring Adaptive Session Redundancy.
- Box-to-box redundancy - Provides chassis-level redundancy between two identically configured CSSs. For details, refer to Chapter 3, 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

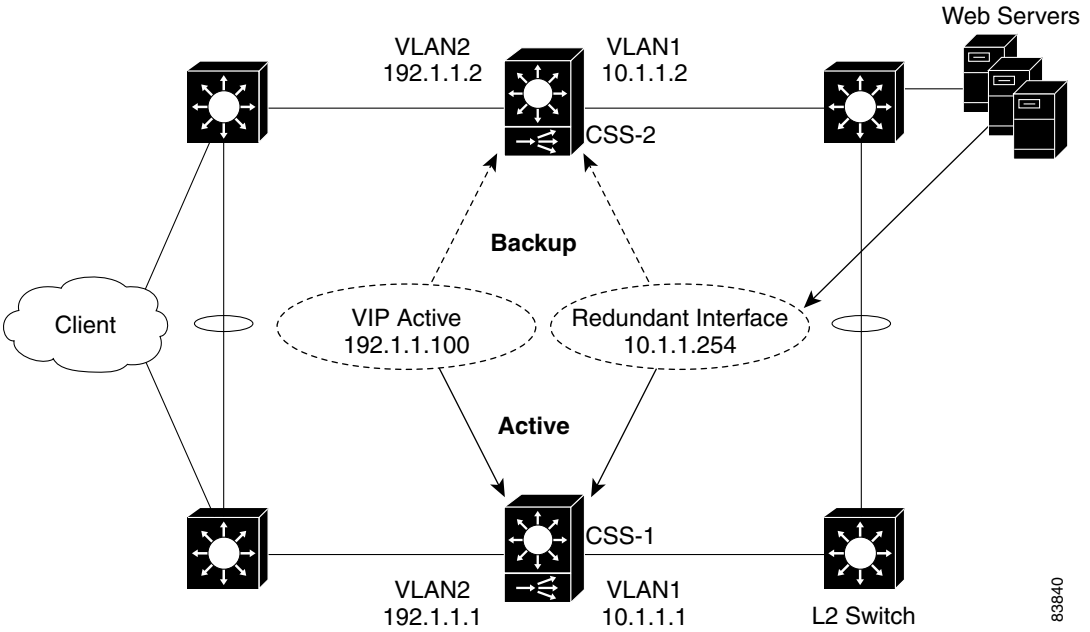
When you configure a pair of CSSs to process client requests for the same VIP address, the VIP address is considered *redundant*. A typical use of VIP redundancy is with a virtual interface redundancy configuration where the master CSS processes all client requests to a VIP with a Web-server farm behind the CSSs and connected to the CSSs through a Layer 2 switch (Figure 1-1). 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 3, Configuring Box-to-Box Redundancy.

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

Note The VIP address must already exist in at least one active content rule or source group.

Figure 1-1 Example of VIP and Virtual Interface Redundancy



Virtual routers providing redundancy for a VIP address are considered a VR pair. Each VR pair has the same VR identifier (VRID) and runs on the same VLAN, but runs on a different CSS. Once the VRs are configured, the CSSs negotiate for mastership using Virtual Router Redundancy Protocol (VRRP). A VR 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 VR, which forwards all client requests directed to the VIP to the master CSS
  - Shared backup VR, 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 gratuitious ARP invocation.

For an example of VIP and virtual interface redundancy, see Figure 1-1.

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 VR 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 redundant virtual interface IP address, the CSSs uses VRRP to negotiate mastership of the redundant virtual interface.

A CSS designated as the master of a redundant virtual interface automatically sends out gratuitous ARPs for the redundant 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 redundant virtual interface from the servers and allows a server's default route to always point to the CSS designated as the master of the redundant virtual interface. The CSS transmits one ARP request packet and one ARP reply packet for every gratuitious ARP invocation.

For an example of VIP and virtual interface redundancy, see Figure 1-1.

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.

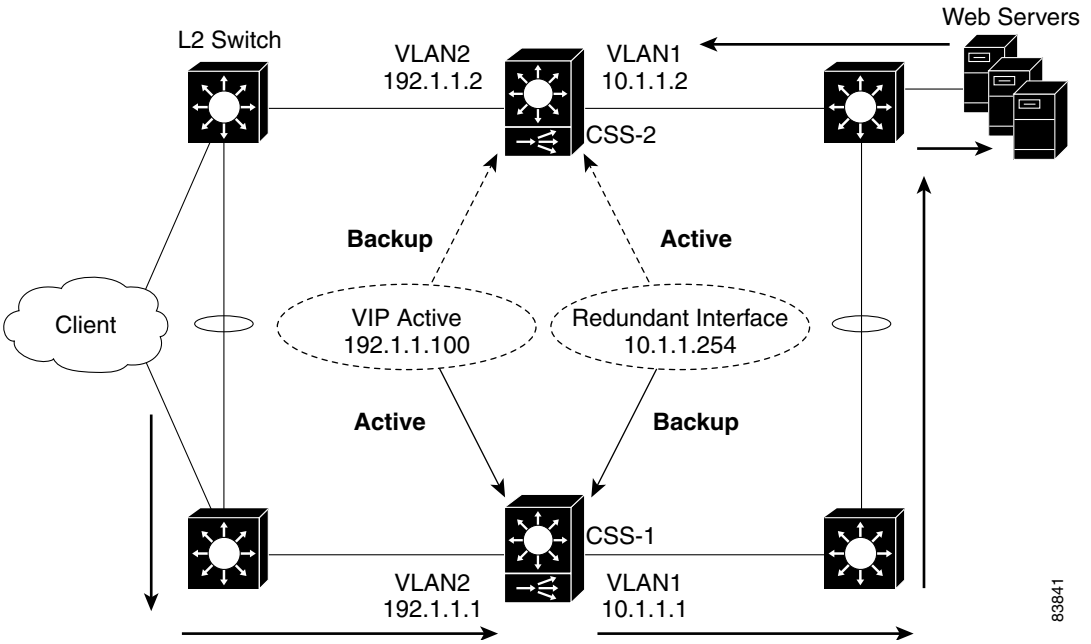
Fate Sharing

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 1-2). 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 VRs) 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 redundant VIP and the redundant virtual interface fail over together to the backup CSS. For details on configuring critical services, see the "Configuring a Critical Service" section.

Figure 1-2 Example of Asymmetric Flows Without Fate Sharing



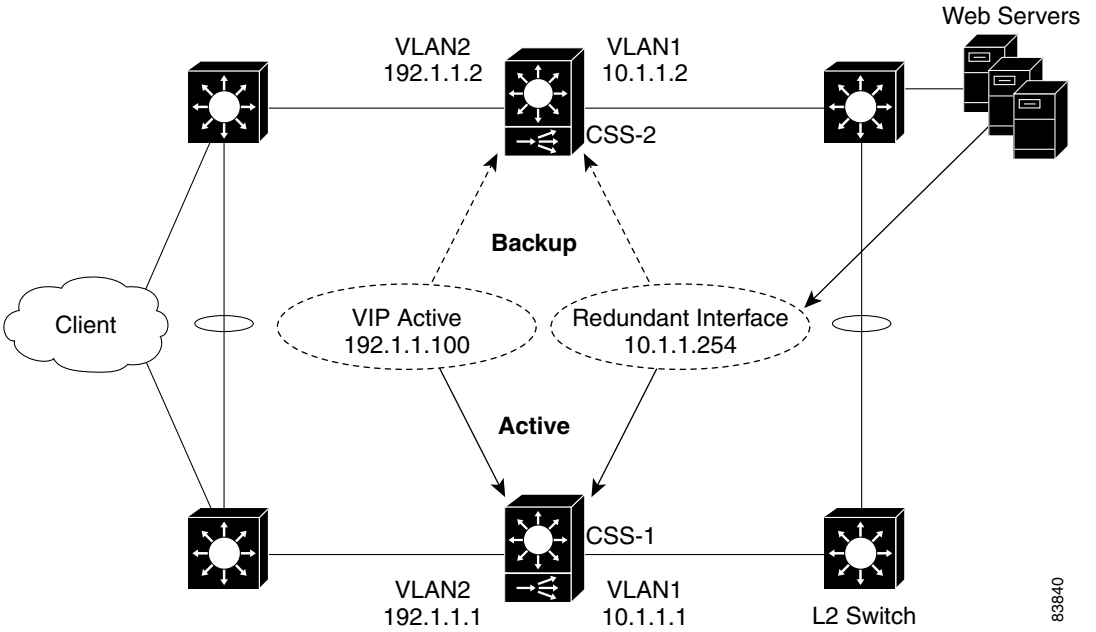
Examples of VIP and Virtual Interface Redundancy Configurations

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 1-3 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 1-3 Example of Active-Backup 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 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

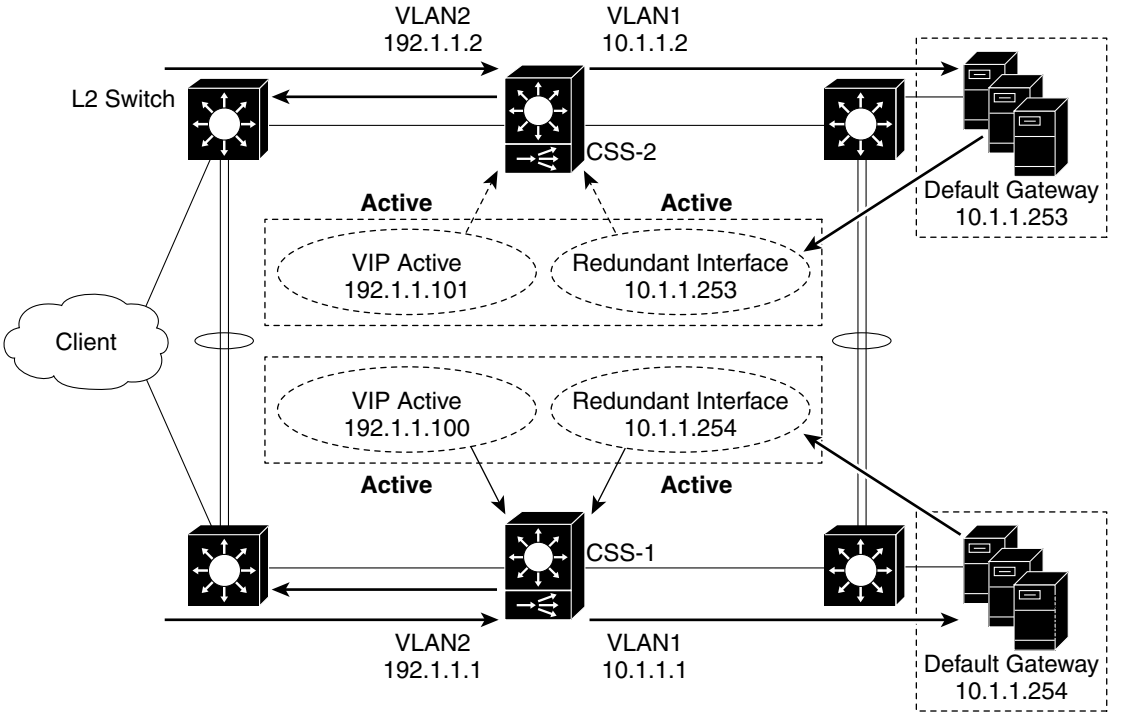
A CSS can serve simultaneously as a master to one VR and as a backup to a different VR. This is called active-active VIP and virtual interface redundancy. All redundant VIP addresses will share the state of the VR to which they are associated. The same VR cannot be active on both CSSs simultaneously.

Figure 1-4 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 VR for VIP address 192.1.1.100 and virtual interface address 10.1.1.254.
  - Backup VR 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 VR for VIP address 192.1.1.101 and virtual interface address 10.1.1.253.

– Backup VR 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 1-4 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 VRs process flows destined to the same VIP. However, only the master VR (CSS-1) responds to ARP requests for the VIP address 192.1.1.100.

Note Shared VIP redundancy is not supported with VRID peering. See the "Configuring VRID Peering" section.

A shared VIP redundancy configuration has the following requirements:

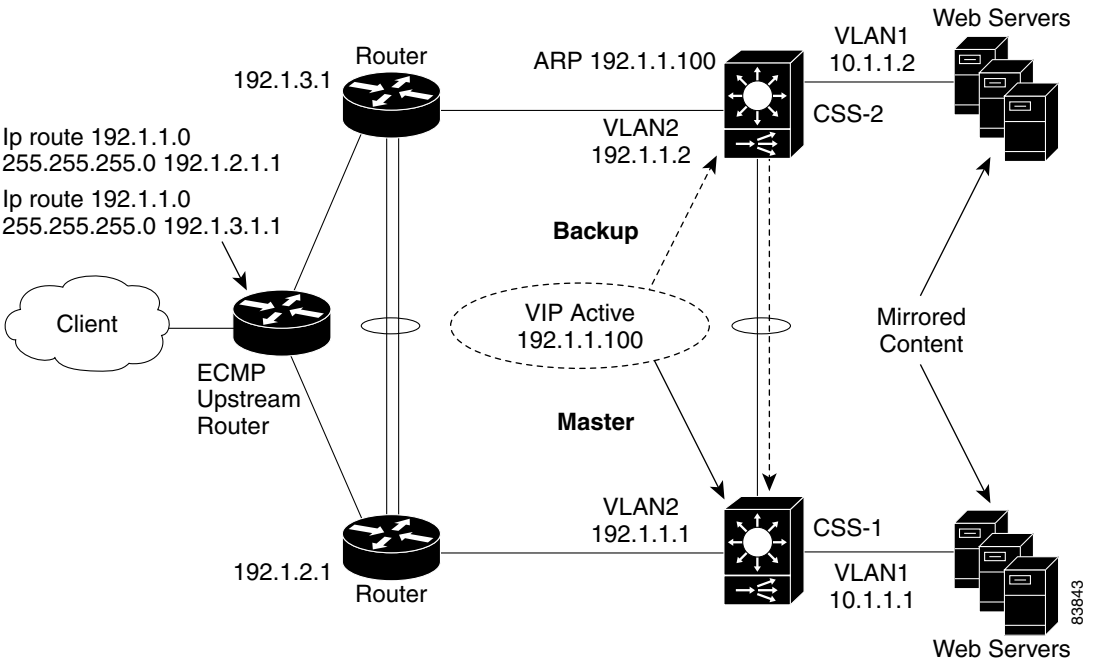
- Direct uplink connections to routers (no common Layer 2 connection between CSSs)
- Direct connection between the CSSs to enable the backup CSS to forward ARP requests to the master CSS
- Mirrored content on the servers
- Direct connection from the servers to the CSSs eliminating the need for fate sharing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

Figure 1-5 shows a shared VIP redundancy configuration with:

- CSS-1 configured as master VR 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 VR for VIP 192.1.1.100) forwards the ARP request for 192.1.1.100 to CSS-1 for a response.

Figure 1-5 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 Configuration Quick Start

Table 1-1 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 1-1.

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

Task and Command Example
1. Enter config mode. **# config** (config)#
2. Enter service mode and configure a service to be used as a critical service or use an existing service. (config)# **service upstream_downstream** (config-service[upstream_downstream])# **ip address 192.1.1.75** (config-service[upstream_downstream])# **keepalive type script ap-kal-pinglist "191.1.1.20 10.1.1.20"** (config-service[upstream_downstream])# **keepalive frequency 2** (config-service[upstream_downstream])# **keepalive maxfailure 2** (config-service[upstream_downstream])# **keepalive retryperiod 2** (config-service[upstream_downstream])# **active**

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

Task and Command Example

3. Enter circuit mode for VLAN1.
`(config)# **circuit VLAN1**`
`(config-circuit[VLAN1])#`

4. Configure a circuit IP address.
`(config-circuit[VLAN1])# **ip address 10.1.1.1/24**`
`(config-circuit-ip[VLAN1-10.1.1.1])#`

5. Configure the VR. 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 VR to assume mastership in all circumstances, include the **preempt** keyword. `(config-circuit-ip[VLAN1-10.1.1.1])# **ip virtual-router 1 priority 101 preempt**`
6. Configure the redundant virtual interface. `(config-circuit-ip[VLAN1-10.1.1.1])# **ip redundant-interface 1 10.1.1.254**`
7. Configure an existing service as a critical service for the VR. `(config-circuit-ip[VLAN1-10.1.1.1])# **ip critical-service 1 upstream_downstream**`

8. Enter circuit mode for the next desired circuit VLAN.
`(config)# **circuit VLAN2**`
`(config-circuit[VLAN2])#`

9. Configure a circuit IP address.
`(config-circuit[VLAN2])# **ip address 192.1.1.1/24**`
`(config-circuit-ip[VLAN2-192.1.1.1])#`

10. Configure the VR. `(config-circuit-ip[VLAN2-192.1.1.1])# **ip virtual-router 2 priority 101 preempt**`
11. Configure the redundant VIP on the VR. `(config-circuit-ip[VLAN2-192.1.1.1])# **ip redundant-vip 2 192.1.1.100**`

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

Task and Command Example
12. Configure the critical service for the VR. `(config-circuit-ip[VLAN2-192.1.1.1])# ip critical-service 2 upstream_downstream`
13. (Recommended) Verify the configuration. `(config)# show virtual-routers`

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

The following running-config example shows the results of entering the commands listed in Table 1-1.

```
!************************ INTERFACE ************************
interface  2/1
  bridge vlan 2

!************************* CIRCUIT *************************
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
!************************* SERVICE *************************
service upstream-downstream
  ip address 192.1.1.10
  keepalive type script ap-kal-pinglist "192.1.1.20 10.1.1.20"
  keepalive frequency 2
  keepalive maxfailure 2
  keepalive retryperiod 2
  active
```

Configuring VIP and Virtual Interface Redundancy

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

Configuring a Circuit IP Interface

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 Routing and Bridging Configuration Guide*.

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

To create a virtual router (VR) on a CSS and configure the identifier and priority that is used when negotiating control of associated VIPs, use the **ip virtual-router** command. You must configure the VR before you can configure redundant VIPs.

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

Note In a VRID peering or critical phy configuration, suspending a reporter that is configured as a critical reporter causes all VRs associated with it to go down, which causes a failover from master to backup. See the "Configuring VRID Peering" and the "Configuring a Critical Physical Interface" sections.

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 VRs per VLAN. Virtual routers are considered s when they have the same VRID and reside on the same VLAN.
- **priority** *number* - The optional priority of the VR with respect to its peers. The default priority value is 100. Enter an integer between 1 and 255. A VR with the highest priority usually becomes master. However, a higher priority VR will not assume mastership from a lower priority master unless you include the **preempt** option.

When a VR is the master, it handles the traffic directed to its associated VIPs. To set a VR 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 VR to assert mastership over a lower priority VR. By default, a VR does not become master if the current master has a lower priority.

  For example, if a CSS with a VR that has a low priority boots before other CSSs, that VR becomes the master. When another CSS with a VR 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 VR. This option does not have an effect if the priority of the two VRs is identical. You can use this option with or without the **priority** option. You can configure only one VR as the master of a particular VIP.

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

Because a VR'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 VR and that VR gains mastership, the CSS automatically reconfigures that VR's priority to 254. This action ensures that you can assign a different VR 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 a Critical Service" section.

For example:

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

To remove the VR from the CSS, enter:

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

Configuring a Redundant VIP

To associate an existing VIP with a VR and, if required, configure the VR as a shared backup, use the **ip redundant-vip** command. A shared backup VR 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 or source group. Additionally, if you defined the content rule or source group VIP using the range option, you must configure an identical range for the redundant VIP. For information about configuring VIPs in content rules and source groups, refer to the *Cisco Content Services Switch Content Load-Balancing 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 VR.
- *vip_address* - The address for the redundant VIP. This address must already be configured in at least one active content rule or source group. 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 or source group. 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 VRs 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 VR, enter:

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

Configuring a Redundant Virtual Interface

Servers use the IP address of a redundant virtual interface as a default gateway to guarantee that packets are sent to the CSS containing the master VR. To accomplish this goal, configure a redundant virtual interface with the same VR as a redundant VIP that is configured in 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. If the master CSS fails over to the backup CSS, both the VIP and the redundant interface fail over together. This configuration is called *fate sharing*. To configure a redundant virtual interface to be used as the default gateway for backend servers, use the **ip redundant-interface** command.

The syntax for this IP mode command is:

ip redundant-interface *vrid ip_address*

The variables are:

- *vrid* - ID for a previously-configured VR.
- *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 source group.`

For example:

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

To remove an interface from a VR, 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 VRID Peering

To ensure that the state of the virtual routers (VRs) on both the public and the private sides of a CSS remain synchronized, configure VRID peering in a VIP and virtual interface redundancy configuration. VRID peering groups VRs that are configured on the same CSS so that, when one VR in the group changes state (for example, from master to backup), all VRs in the group change state at the same time. This feature helps to prevent asymmetric flows, which cause network address translation (NAT) to fail and break the client-server connection.

This section contains the following topics:

Background

In a VIP and virtual interface redundancy configuration, you typically configure a pair of virtual routers (VRs) running VRRP to negotiate mastership of the redundant VIPs on the client side of the network and another pair of VRs to negotiate mastership of the redundant interfaces on the server side of the network. In addition, you can use critical services to provide fate sharing so that the redundant VIPs and the redundant interfaces change state together from master to backup. A CSS uses polling keepalives to monitor the states of critical services.

Because a link on either the public side or the private side of a CSS can go down and then up very quickly in between keepalive polling times, it is possible for a VR on one CSS to be the master for the redundant VIP and a VR on the other CSS to be the master for the redundant virtual interface. This loss of synchronization between VRs can occur even when you have configured critical services. While this unsynchronized VR state is permitted, it may not be desirable in all cases.

Unsynchronized VRs produce asymmetric flows, which cause NAT to fail. In this case, packets from a server will be routed to the CSS that is master for the redundant virtual interface (default gateway), while packets from a client will be routed to the other CSS that is master for the redundant VIP.

Overview of VRID Peering

VRID peering works by grouping two or more VR peers with a software agent called a *reporter* that monitors the states of the VRs. VRs are considered peers when they are configured on the same CSS. The reporter ensures that the states of the VR peers are synchronized by sending internal state update messages to the monitored VRs when necessary.

The internal state of a virtual router is called an independent VR state because it does not depend on the state of its VR peers. The VR peer reporter state is called a dependent VR state because it depends on the states of all VRs configured on the reporter.

Table 1-2 lists the independent VR states and describes the conditions required for each state.

Table 1-2 Independent VR States and Conditions

Independent State	Conditions
Down	Circuit where the VR runs is down or one of the critical services configured on the VR failed
Backup	Circuit where the VR runs is up, there are no failures on the critical services configured on the VR, and VRRP has determined that the VR should be in the backup state
Master	Circuit where the VR runs is up, there are no failures on the critical services configured on the VR, and VRRP has determined that the VR should be in the master state

The reporter that is responsible for monitoring the VR peers determines the dependent state of the VRs. Table 1-3 lists the dependent VR states and describes the conditions required for each state.

Table 1-3 Dependent VR States and Conditions

Dependent State	Conditions
Down	At least one of the VR peers in a reporter group is in the Down state. The VRs will hold the VIPs and redundant virtual interfaces that they control in the Down state.
Backup	At least one of the VR peers in a reporter group is in the Backup state and none is in the Down state. The VRs will hold the VIPs and redundant virtual interfaces that they control in the Backup state.
Master	All VR peers in a reporter group are in the Master state. The VRs transmit VRRP messages announcing that they are in the Master state. VIPS and redundant virtual interfaces controlled by the VRs become masters.

When two or more VRs are configured on a reporter, their dependent state is that of the lowest independent state of all the VR peers in the group. Table 1-4 shows the effects of independent VR states on reporter dependent states for two VRs.

Table 1-4 Effect of Independent VR States on Reporter Dependent States

VR1 Independent State	VR2 Independent State	Reporter Dependent State
Down	Down	Down
Down	Backup	Down
Down	Master	Down
Backup	Down	Down
Backup	Backup	Backup
Backup	Master	Backup
Master	Down	Down
Master	Backup	Backup
Master	Master	Master

Configuration Requirements and Restrictions

The following requirements and recommendations apply to the configuration and use of VRID peering on a Cisco 11500 series CSS.

- Ensure that you have configured VIP and virtual interface redundancy properly. See the "Configuring VIP and Virtual Interface Redundancy" section.
- VRID peering is not supported with shared VIP redundancy.
- Ensure that a VR exists before you attempt to configure it on a reporter. See the "Configuring a Virtual Router" section.
- All VRs associated with a VRID peering reporter must have the same priority and preempt configurations.
- Do not configure the same IP address and VRID on more than one reporter.
- You can configure a maximum of 128 reporters on a CSS.
- You can configure a maximum of four reporters of type VRID peer on a CSS.
- You can configure a maximum of eight VRIDs on a reporter of type VRID peer.

VRID Peering Quick Start

Table 1-5 provides a quick overview of the steps required to configure VRID peering. 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 1-5.

Table 1-5 VRID Peering Configuration Quick Start

Task and Command Example

1. Enter config mode.
 	
`# config`
`(config)#`

2. Enter reporter configuration mode and create a new reporter. See the "Configuring a Reporter" section.
 	
`(config)# reporter r1`
`(config-reporter[r1])#`

3. Configure the VRID peering type on the reporter. See the "Configuring the Reporter Type" section. `(config-reporter[r1])# type vrid-peering`

4. Specify the VRs that you want to monitor. See the "Configuring the Virtual Routers That You Want to Monitor" section.
 	
`(config-reporter[r1])# vrid 192.168.100.5 1`
`(config-reporter[r1])# vrid 172.16.27.12 2`

5. Activate the reporter. See the "Activating a Reporter" section. `(config-reporter[r1])# active`

6. Associate the reporter with an existing VRID. See the "Configuring a Critical Reporter" section.
 	
`(config-circuit-ip[VLAN1-192.168.100.5])# ip critical-reporter 1 r1`
`(config-circuit-ip[VLAN2-172.16.27.12])# ip critical-reporter 2 r1`

7. (Recommended) Verify the reporter configuration. See the "Displaying a Reporter Configuration in the Running-Config" section. `(config-circuit-ip[VLAN2-172.16.27.12])# show reporter r1`
8. (Recommended) Verify the VRID peering configuration. See the "Displaying Critical Reporter Information" section. `(config-circuit-ip[VLAN2-172.16.27.12])# show running-config reporter`

The following running-config example shows the results of entering the commands in Table 1-5.

```
!************************* CIRCUIT **************************
circuit VLAN1

  ip address 192.168.100.5 255.255.255.0
    ip virtual-router 1 priority 101 preempt
    ip critical-reporter 1 r1

circuit VLAN2

  ip address 172.16.27.12 255.255.255.0
    ip virtual-router 2 priority 101 preempt
    ip critical-reporter 2 r1

!************************* REPORTER *************************
reporter r1
  type vrid-peering
  vrid 192.168.100.5 1
  vrid 172.16.27.12 2
  active
```

Configuring a Reporter

A reporter is a software agent that monitors the states of all VRs associated with it. When a VR state change is necessary, the reporter sends a state update message to its associated VRs. To configure a reporter and enter reporter configuration mode, use the **reporter** command in global configuration mode. You can configure a maximum of 128 reporters on a CSS.

This command has the following syntax:

reporter *reporter_name*

The *reporter_name* variable specifies the name of the reporter you are creating. Enter an unquoted text string with no spaces from 1 to 31 characters.

For example, enter:

```
(config)# reporter r1
```

To remove an existing reporter from the running-config, enter:

```
(config-reporter[r1])# no reporter r1
```

Note If you remove a reporter from the running-config using the **no reporter** command, the CSS removes all the attributes associated with that reporter from the running-config.

Configuring the Reporter Type

A VRID peer is a type of reporter that monitors the states of associated VRs and ensures that the VR states are synchronized. To configure the reporter type, use the **type** command in reporter configuration mode. You can configure a maximum of four reporters of type **vrid-peering** on a CSS.

Note The CSS supports reporters of type **vrid-peering** only with non-shared active-backup and active-active VIP redundancy configurations.

This command has the following syntax.

type *reporter_type*

For example, to configure a reporter as a VRID peering type, enter:

```
(config-reporter[r1])# type vrid-peering
```

To remove the VRID peering type or to reconfigure the reporter type as a critical phy type:

1. Suspend the reporter. See the "Suspending a Reporter" section.

   Note If the reporter is configured as a critical reporter, suspending it causes the associated VR to transition to Backup or Down.

2. Remove the VRID peering reporter attributes using the **no vrid** *ip_address vrid* command. See the "Configuring the Virtual Routers That You Want to Monitor" section.
3. Remove the VRID peering reporter type using the **no type** command or reconfigure the reporter type as a critical phy type using the **type critical-phy-all-up** or the t**ype critical-phy-any-up** command. For details about configuring a critical phy, see the "Configuring a Critical Physical Interface" section.

4. If you reconfigured the reporter type as a critical phy type, add the physical interfaces using the **phy** *interface_name* command, then activate the reporter using the **active** command. See the "Configuring the Physical Interfaces That You Want to Monitor" section and the "Activating a Reporter" section.

Configuring the Virtual Routers That You Want to Monitor

To configure the VRs that you want a reporter to monitor, use the **vrid** command in reporter configuration mode. This command allows you to configure a maximum of eight VRIDs on a reporter of type **vrid-peering**.

This command has the following syntax:

vrid *ip_address vrid*

Note You cannot configure the same circuit IP address and VRID on more than one reporter.

The variables for this command are:

- *ip_address* - The circuit IP address of the CSS in dotted-decimal notation.
- *vrid* - The VR identifier (VRID). Enter the VRID of an existing VR. You can configure a maximum of eight VRIDs on one reporter. For details on configuring a VR, see the "Configuring a Virtual Router" section.

For example:

```
(config-reporter[r1])# vrid 192.168.12.7 1
```

Note You cannot remove the last remaining VR from an active reporter. To remove the VR, first suspend the reporter, and then remove the VR.

To remove a VR from a reporter, enter:

```
(config-reporter[r1])# no vrid 192.168.12.7 1
```

Activating a Reporter

You must activate a reporter before it can monitor the state of its configured VRIDs. To activate a reporter, use the **active** command in reporter configuration mode. A reporter remains in the Suspended state until you activate it.

For example, enter:

```
(config-reporter[r1])# active
```

Suspending a Reporter

You can suspend an active reporter to temporarily stop using the reporter or to change the reporter configuration. To suspend the reporter, enter the **suspend** command in reporter configuration mode. When you are ready to resume using the reporter again, reactivate the reporter using the **active** command. See the "Activating a Reporter" section.

Note Suspending a reporter that is configured as a critical reporter causes all VRs associated with it to go down, which causes a failover from master to backup. See the "Configuring a Critical Reporter" section.

For example, enter:

```
(config-reporter[r1])# suspend
```

Configuring a Critical Reporter

To associate a reporter with a VR, use the **ip critical-reporter** command in circuit configuration mode. You can associate more than one critical reporter with a VR provided that the critical reporters are of different types. See the "Configuring a Critical Physical Interface" section.

If any critical reporter is suspended or goes down, all VRs associated with it go down. To ensure that the VR states are synchronized, configure a critical reporter on both the front-side and the back-side VRs.

The syntax of this command is:

ip critical-reporter *vrid reporter_name*

The variables are:

- *vrid* - The virtual router identifier (VRID) of an existing VR. Enter an integer between 1 and 255.
- *reporter_name* - The name of an existing reporter. Enter an unquoted text string with no spaces from 1 to 31 characters.

For example, to associate reporter r1 with a VR that has a VRID of 1, enter:

```
(config-circuit-ip[VLAN1-192.168.7.9])# ip critical-reporter 1 r1
```

To remove a critical reporter from the running-config, enter:

```
(config-circuit-ip[VLAN1-192.168.7.9])# no ip critical-reporter 1 r1
```

Configuring a Critical Service

Configure a critical service to monitor the health of upstream and downstream devices. When one or all critical services go down (depending on the type of critical service you configure), the associated VR also goes down, which causes a failover from master CSS to backup CSS. 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 VR 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 VR goes down when *all* associated redundancy uplink services go down regardless of any configured keepalive type. Refer to Chapter 3, 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 VR goes down when *all* associated local critical services go down.

To associate a critical service with a VR, use the **ip critical-service** command. The syntax for the **ip critical-service** command is:

ip critical-service *vrid service_name*

The variables are:

- *vrid* - The ID for an existing VR.
- *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 VR, 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*.

Configuring a Critical Physical Interface

Configure a critical physical interface (critical phy) on a CSS to provide an additional failover catalyst for a virtual router (VR) in a VIP and interface redundancy configuration. A critical phy improves the failover time of a VR (as compared with a critical service) by reacting quickly to a Down state of monitored physical interfaces. This feature is intended to complement critical services, not replace them. For details on critical services, see the "Configuring a Critical Service" section.

This section contains the following topics:

Overview

A critical phy monitors the health of its associated physical interfaces and causes a VR to fail over to the backup CSS if one or all (depending on the configuration) monitored interfaces go down. Unlike a critical service, a critical phy does not depend on the state of a keepalive for its operation. Therefore, a critical phy is not susceptible to reporting delays and packet loss due to network congestion and packet storms, which, in the case of a critical service, may cause the CSS to incorrectly report a server as unavailable. When you associate a critical phy with a VR, the critical phy provides rapid VR failover in the event of a physical link failure.

To configure a critical phy, you create a software agent called a *reporter*, a general-purpose monitoring mechanism. Then you specify the reporter type of **critical-phy** and configure the reporter to monitor the health of one or more physical interfaces. To complete the configuration, you modify your VIP and virtual interface redundancy configuration to associate a critical reporter with an existing VR.

Configuration Requirements and Restrictions

The following requirements and recommendations apply to the configuration and use of critical phys on a Cisco 11500 series CSS:

- Ensure that VIP and virtual interface redundancy is configured properly (see the "Configuring VIP and Virtual Interface Redundancy" section).
- If you associate more than one critical reporter with the same VR, ensure that you do not configure the same physical interfaces (ports) on two different reporter types (for example, ports 1/1 and 1/2 on a reporter of type **critical-phy-all-up** and ports 1/1 and 1/2 on a reporter of type **critical-phy-any-up**). Otherwise, unexpected VR failovers may occur.
- Do not configure the Ethernet management port or the console port as critical-phy interfaces to be monitored by a reporter.
- Do not use critical-phy interfaces as InterSwitch Communications (ISC) ports in an ASR environment.
- Do not use critical-phy interfaces as ports to be monitored by the Switch Port Analyzer (SPAN) feature.
- Do not use critical-phy interfaces as redundancy-phy interfaces in a box-to-box redundancy configuration.

Critical Phy Quick Start

Table 1-6 provides a quick overview of the steps required to configure a critical phy on a reporter. 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 1-6.

Table 1-6 Critical Phy Configuration Quick Start

Task and Command Example

1. Enter config mode.
`# config`
`(config)#`

2. Enter reporter configuration mode and create a new reporter. See the "Configuring a Reporter" section.
`(config)# reporter r1`
`(config-reporter[r1])#`

3. Configure the reporter type. See the "Configuring the Reporter Type" section. `(config-reporter[r1])# type critical-phy-all-up`

4. Specify the physical interfaces that you want to monitor.
`(config-reporter[r1])# phy 1/1`
`(config-reporter[r1])# phy 1/2`

5. Activate the reporter. `(config-reporter[r1])# active`

6. Exit reporter mode.
`(config-reporter[r1])# exit`
`(config)#`

7. Enter circuit configuration mode.
`(config)# circuit VLAN1`
`(config-circuit[VLAN1])# ip address 192.168.7.9`
`(config-circuit-ip[VLAN1-192.168.7.9])#`

8. Associate the reporter with an existing VRID. `(config-circuit-ip[VLAN1-192.168.7.9])# ip critical-reporter 1 r1`
9. (Recommended) Verify the reporter configuration. `(config-circuit-ip[VLAN1-192.168.7.9])# show reporter r1`

Table 1-6 Critical Phy Configuration Quick Start (continued)

Task and Command Example
10. (Recommended Verify the critical phy configuration. `(config-circuit-ip[VLAN1-192.168.7.9])# show running-config reporter`
11. (Recommended) Verify the VR and critical reporter configurations. `(config-circuit-ip[VLAN1-192.168.7.9])# show running-config circuit`

The following running-config example shows the results of entering the commands in Table 1-6.

```
!************************* CIRCUIT **************************
circuit VLAN1

  ip address 192.168.7.9 255.255.255.0
    ip virtual-router 1 priority 101 preempt
    ip critical-reporter 1 r1

!************************* REPORTER *************************
reporter r1
  type vrid-peering
  phy 1/1
  phy 1/2
  active
```

Configuring a Reporter

A reporter is a software agent that the CSS uses to monitor the health of physical interfaces when you configure the reporter as a critical phy type and associate physical interfaces with it. To configure a reporter and enter reporter configuration mode, use the **reporter** command in global configuration mode. This command has the following syntax:

reporter *reporter_name*

The *reporter_name* variable specifies the name of the reporter you are creating. Enter an unquoted text string with no spaces from 1 to 31 characters.

For example, enter:

```
(config)# reporter r1
```

To remove an existing reporter and all of its attributes from the running-config, enter:

```
(config-reporter[r1])# no reporter r1
```

Configuring the Reporter Type

A critical phy is a type of reporter that determines how the reporter reacts to a Down state of the associated physical interfaces. To configure a critical phy type on a reporter, use the **type** command in reporter configuration mode. This command has the following syntax.

type *reporter_type*

Note If you associate more than one reporter with the same VR, we recommend that you do not configure the same physical interfaces (ports) on two different reporter types (for example, ports 1/1 and 1/2 on a reporter of type **critical-phy-all-up** and ports 1/1 and 1/2 on a reporter of type **critical-phy-any-up**). Otherwise, unexpected VR failovers may occur.

The *reporter_type* variable has one of the following values:

- **critical-phy-all-up** - If any critical interface goes down, the reporter goes down and mastership of the associated VR transitions from the master CSS to the backup CSS. To prevent a VR failover, all interfaces must remain up.
- **critical-phy-any-up** - If all associated critical interfaces go down, the reporter goes down and mastership of the associated VR transitions from the master CSS to the backup CSS. As long as one critical interface stays up, the reporter and the VR remain up.

You can change the critical phy reporter type without suspending the reporter. However, if you want to reconfigure a critical phy reporter as a VRID peering reporter, you must first suspend the reporter to remove the critical phy reporter attributes. See the "Suspending a Reporter" section.Then you can configure the reporter as a VRID peering reporter and activate it. For details about VRID peering, see the "Configuring VRID Peering" section.

You can configure a maximum of 128 reporters of any combination of types on a CSS, depending on available memory.

When you configure a critical phy, the states of the monitored physical interfaces affect the reporter state, which in turn affects the VR state, depending upon the type of configured reporter as shown in Table 1-7.

Table 1-7 Effect of Interface State on Reporter and Virtual Router State Based on Reporter Type

Reporter Type	Interface State	Reporter State	Virtual Router State
critical-phy-all-up	All Up	Up	Up
critical-phy-all-up	All Down	Down	Down
critical-phy-all-up	One or More Down	Down	Down
critical-phy-any-up	All Up	Up	Up
critical-phy-any-up	All Down	Down	Down
critical-phy-any-up	One or More Up	Up	Up

For example, enter:

```
(config-reporter[r1])# type critical-phy-all-up
```

To remove the critical phy type (either **critical-phy-all-up** or **critical-phy-any-up**) or to reconfigure the reporter type as a **vrid-peering** type:

1. Suspend the reporter. See the "Suspending a Reporter" section.

   Note If the reporter is configured as a critical reporter, suspending it causes the associated VR to transition to Backup or Down.

2. Remove the critical phy reporter attributes using the **no phy** *interface_name* command. See the "Configuring the Physical Interfaces That You Want to Monitor" section.
3. Remove the **critical phy** reporter type using the **no type** command or reconfigure the reporter type as a VRID peering type using the **type vrid-peering** command. For details about configuring VRID peering, see the "Configuring VRID Peering" section.

4. If you reconfigured the reporter type as a VRID peering type, add the VRIDs using the **vrid** *ip_address vrid* command, and then activate the reporter using the **active** command. See the "Configuring the Virtual Routers That You Want to Monitor" section and the "Activating a Reporter" section.

Configuring the Physical Interfaces That You Want to Monitor

To configure one or more physical interfaces that you want a reporter to monitor, use the **phy** command in reporter configuration mode. You can configure a maximum of 128 interfaces on a reporter.

Note If you associate more than one reporter with the same VR, we recommend that you do not configure the same physical interfaces (ports) on two different reporter types (for example, ports 1/1 and 1/2 on a reporter of type **critical-phy-all-up** and ports 1/1 and 1/2 on a reporter of type **critical-phy-any-up**). Otherwise, unexpected VR failovers may occur.

This command has the following syntax:

phy *interface_name*

The *interface_name* variable is the name of the physical interface that you want to monitor. Enter an interface name in interface port format (for example, e1 on a CSS 11501) or slot/port format (for example, 1/1 on a CSS 11503 and CSS 11506). See the following configuration examples.

To configure Ethernet port 1 on a CSS 11501, enter:

```
(config-reporter[r1])# phy e1
```

To configure Ethernet port 1 on a CSS 11503 or 11506, enter:

```
(config-reporter[r1])# phy 1/1
```

Note You cannot remove the last remaining physical interface from an active reporter. To remove the interface, first suspend the reporter, and then remove the interface.

To remove Ethernet port 1 from the list of interfaces to monitor on a CSS 11501, enter:

```
(config-reporter[r1])# no phy e1
```

To remove Ethernet port 1 from the list of interfaces to monitor on a CSS 11503 or 11506, enter:

```
(config-reporter[r1])# no phy 1/1
```

Activating a Reporter

Before a CSS can use a reporter to monitor the health of the configured critical interfaces, you must activate the reporter using the **active** command. A reporter remains in a suspended state until you activate it.

For example, enter:

```
(config-reporter[r1])# active
```

Suspending a Reporter

You can suspend an active reporter to temporarily stop using the reporter or to change the reporter configuration. To suspend the reporter, enter the **suspend** command in reporter configuration mode. When you are ready to resume using the reporter again, reactivate the reporter using the **active** command. See the "Activating a Reporter" section.

For example, enter:

```
(config-reporter[r1])# suspend
```

Note Suspending a reporter that is configured as a critical reporter causes all VRs associated with it to go down, which causes a failover from master to backup. See the "Configuring a Critical Reporter" section.

Configuring a Critical Reporter

To associate a reporter with a VR, use the **ip critical-reporter** command in circuit configuration mode. You can associate more than one critical reporter with a VR. If any critical reporter is suspended or goes down, all VRs associated with it go down and cause a failover from master to backup.

Note If you associate more than one reporter with the same VR, we recommend that you do not configure the same physical interfaces (ports) on two different reporter types (for example, ports 1/1 and 1/2 on a reporter of type **critical-phy-all-up** and ports 1/1 and 1/2 on a reporter of type **critical-phy-any-up**). Otherwise, unexpected VR failovers may occur.

The syntax of this command is:

ip critical-reporter *vrid reporter_name*

The variables for this command are:

- *vrid* - The virtual router identifier (VRID) of an existing VR. Enter an integer between 1 and 255. Virtual routers are considered peers when they have the same VRID and reside on the same VLAN.
- *reporter_name* - The name of an existing reporter. Enter an unquoted text string with no spaces from 1 to 31 characters.

For example, enter:

```
(config-circuit-ip[VLAN1-192.168.7.9])# ip critical-reporter 1 r1
```

To remove a critical reporter, enter:

```
(config-circuit-ip[VLAN1-192.168.7.9])# no ip critical-reporter 1 r1
```

Synchronizing a VIP Redundancy Configuration

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

This type of synchronization copies reporter configurations for both VRID peering (see the "Configuring VRID Peering" section) and critical phy (see the "Configuring a Critical Physical Interface" section) with the following exceptions:

- If you configure a critical physical interface on a reporter on the local CSS and that interface does not exist on the remote CSS or is of a different interface type (Gigabit Ethernet or Fast Ethernet), the script exits and the synchronization does not complete.
- If you configure one or more VRID IP addresses on a reporter on the local CSS for VRID peering, the script preserves any configured VRID IP addresses on the remote CSS. If the remote CSS configuration does not contain a VRID IP address that corresponds with one in the local CSS configuration, the remote CSS lacks that VRID IP address when the script finishes. The script does copy the reporter configuration from the local CSS to the remote CSS. The script exits with the `File copy Vipr Config Sync Complete` message and indicates that any byte differences between the local and remote configurations exist because the script did not find a corresponding VRID IP address on the remote 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 remote interface and circuit configurations automatically. For CSSs with configured reporters, the script does not copy the reporter configurations to the remote CSS regardless of chassis types.

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

- Preserves all existing VRID priorities and preempt settings on both the master and the backup CSSs
- 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 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 the *Cisco Content Services Switch Global Server Load-Balancing Configuration Guide*.

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 LOCAL_VIPR_IP and REMOTE_VIPR_IP Variables" 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 or critical reporters 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 LOCAL_VIPR_IP and REMOTE_VIPR_IP Variables" 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:

```
# commit_VipRedundConfig
```

The following output appears:

```
# 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 the *Cisco Content Services Switch Administration Guide*.

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 LOCAL_VIPR_IP and REMOTE_VIPR_IP Variables

To eliminate the need to specify IP addresses each time you run the configuration synchronization script, you can set the value of two variables (LOCAL_VIPR_IP and REMOTE_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. Set the LOCAL_VIPR_IP variable to the IP address of the master CSS and set the REMOTE_VIPR_IP variable to the IP address of the backup CSS.

To set the variables, enter:

```
# set LOCAL_VIPR_IP "master_ip_address" session
# set REMOTE_VIPR_IP "backup_ip_address" session
```

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.

Note If you already created the MASTER_VIPR_IP and BACKUP_VIPR_IP variables in an earlier release, the script will use the new variables instead, if present.

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 VIP and virtual interface redundancy configurations. The following sections describe the commands and provide tables describing the output fields.

- Displaying Redundant Virtual Interfaces
- Displaying Redundant VIPs
- Displaying Virtual Router Configurations
- Displaying IP Critical Services
- Displaying Reporter Configurations

Displaying Redundant Virtual Interfaces

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

The syntax for this command is:

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

The optional variables are:

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

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

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

Table 1-8 describes the fields in the **show redundant-interfaces** command output.

Table 1-8 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 VR.
Redundant Address	IP address of the redundant virtual interface.
Range	Not applicable. This field is always set to 1.
State	Current state of the redundant virtual interface. Possible states are: • **Master** - The redundant virtual interface is the master of the flows between the CSS and the network device connected to the redundant virtual interface • **Backup** - The redundant virtual interface is the backup for the flows between the CSS and the network device connected to the redundant virtual interface • **Idle** - The redundant virtual interface is idle
Master IP	IP address of the master VR.
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.

Table 1-8 Field Descriptions for the show redundant-interfaces Command (continued)

Field	Description
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

To display a list of all redundant VIPs configured on the CSS, use the **show redundant-vips** command. 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 VR.

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 VR.

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

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

Table 1-9 describes the fields in the **show redundant-vips** command output.

Table 1-9 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 VR.
Redundant Address	The IP address of the VIP.

Table 1-9 Field Descriptions for show redundant-vips Command (continued)

Field	Description
Range	The range associated with the VIP.
State	Current state of the redundant VIP. Possible states are: • **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 VR.
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

To display a list of all VRs configured on the CSS, including their configuration and state information, use the **show virtual-routers** command. If VRID peering (see the "Configuring VRID Peering" section) or critical phy (see the "Configuring a Critical Physical Interface" section) is configured on the CSS, this command also displays any critical reporters associated with the VRs.

You may provide an interface IP address option to display only the VRs present on a particular interface. You may also include a VRID to display only the information for a particular VR.

The syntax for this command is:

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

The optional variables are:

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

For example, to view all VRs on the CSS, enter:

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

Table 1-10 describes the fields in the **show virtual-routers** command output.

Table 1-10 Field Descriptions for the show virtual-routers Command

Field	Description
Interface Address	The interface IP address associated with the VR.
VRID	The configured identifier of the VR.
Priority	The priority currently being advertised by the VR. 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 operational state of the VR. Possible states are: • **Master** - The VR is the master. • **Backup** - The VR is the backup. • **Idle** - The VR does not have any redundant virtual interfaces, VIPs, or reporters associated with it. • **Down** - The VR is down. See the Fail Reason field for an explanation of the failure.
Master IP	The IP address of the master VR.
State Changes	The number of times the VR state has changed since the CSS was booted.
Last Change	The data and time of the last VR state change.
Preempt	The state of the preempt option on the VR. If enabled, the state is True; if disabled, the state is False.

Table 1-10 Field Descriptions for the show virtual-routers Command (continued)

Field	Description
Fail Reason	The reason that the failover occurred. Possible reasons are: • **IF Down** - The IP interface associated with the VR is down. • **No Service, Reporter is Down** - One or more critical services and critical reporters associated with the VR are down. • **Critical Phy Down**, **VRID Peering Down** - A critical-phy reporter for the VR and VRID peering are down. • **No Service** - One or more critical services associated with the VR are down. • **Critical Phy Down** - A critical-phy reporter for the VR is down. • **VRID Peering Down** - VRID peering is down.
Critical-Services	The names of the critical services associated with the VR.
State	The current condition of the critical service. Possible states are Up, Down, or Suspended.
Type	The type of critical service. Possible types are Scripted, RedundancyUp, or Local.
Critical-Reporters	The names of the critical reporters associated with the VR.
State	The current condition of the critical reporters associated with the VR. Possible states for VRID peering are Master, Backup, Down, or Suspended. Possible states for critical phy are Up, Down, or Suspended.
Type	The type of critical reporter. Possible types are vrid-peering, critical-phy-all-up, or critical-phy-any-up.

Resetting the Virtual Router State Changes Counter

The **show virtual-routers** command displays a State Changes field that records the number of times that a VR changed state since the CSS was booted. To set this counter to zero, use the **zero virtual-router state-changes** command in any mode.

The syntax for this command is:

zero virtual-router state-changes [all|circuit *ip_address* [**all|vrid** *number*]]

The variables and options for this command are:

- **all** - Zeroes the State Changes counter of all VRs configured on the CSS
- **circuit** *ip_address* - Specifies a circuit IP address where VRs are configured
- **all** - Zeroes the State Changes counter of all VRs on the specified circuit
- **vrid** *number* - Zeroes the State Changes counter of the specified VR on the specified circuit

For example, to reset the State Changes counter for all VRs configured on the circuit with an IP address of 192.168.1.7, enter:

```
(config)# zero virtual-router state-changes circuit 192.168.1.7 all
```

Displaying IP Critical Services

To display a list of all critical services configured on the CSS, use the **show critical-services** command. 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 VR.

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 VR.

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

 # show critical-services

Table 1-11 describes the fields for the **show critical-services** command output.

Table 1-11 Field Descriptions for the show critical-services Command

Field	Description
Interface Address	The IP interface address associated with the VR.
VRID	The assigned identifier associated with the VR.
Service Name	The name of the critical service.
Service Type	The type of critical service. Possible critical service types are: • **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 Reporter Configurations

To display reporter configurations for VRID peering and critical phy, use the following commands:

- **show reporter** {*reporter_name*|**summary**} - Displays the reporter name, type, state, and the critical interfaces and their link states.
- **show running-config reporter** - Displays the configurations of all reporters configured on the CSS.
- **show running-config reporter** *reporter_name* - Displays the configuration of the specified reporter.
- **show virtual-routers** - Displays the configured virtual router information and the critical reporter state and type. See the "Displaying Virtual Router Configurations" section.
- **show critical-reporters** - Displays the critical reporters configured for critical phy.
- **show running-config circuit** - Displays the critical reporter associated with a circuit VLAN for a critical phy.

Using the show reporter Command

To display reporter configuration and statistics for VRID peering or critical phy, use the **show reporter** command. The syntax of this command is:

show reporter {*reporter_name*|**summary**}

The variables and options for this command are:

- *reporter_name* - Name of an existing reporter
- **summary** - Displays summary statistics for all configured reporters

Table 1-12 describes the fields for the **show reporter** command output.

Table 1-12 Field Descriptions for the show reporter Command

Field	Description
Name	Name of the reporter whose configuration you are displaying.

Table 1-12 Field Descriptions for the show reporter Command (continued)

Field	Description
State	Current state of the reporter. Possible reporter states for VRID peering are Master, Backup, Suspended, or Down. Possible reporter states for critical phy are Up, Suspended, or Down.
Type	Type of reporter. Possible reporter types are: • vrid-peering (see the "Configuring VRID Peering" section) • critical-phy-all-up (see the "Configuring a Critical Physical Interface" section) • critical-phy-any-up (see the "Configuring a Critical Physical Interface" section)
State Transitions	Number of times the reporter state changed since the last time the CSS was booted. If the State Transitions field is 0, the 0 value can be due to a counter reset through the global configuration mode **zero reporter state-transitions** command. The counter can also be 0 if the reporter is down.
Circuit (VRID peering only)	Circuit IP address of the CSS.
VRID (VRID peering only)	Identifier of the VR associated with the reporter.
State (VRID peering only)	Current state of the VR associated with the reporter. Possible states of the VR are Master, Backup, Down, or Unknown.
Interface (critical phy only)	Interfaces associated with the reporter.
Link (critical phy only)	Current state of the physical link interface. Possible link states are Up or Down.

Resetting the Reporter State Transitions Counter

The **show reporter** command displays a State Transitions field that records the number of times that a reporter changed state since the CSS was booted. To set this counter to zero, use the **zero reporter state-transitions** command in any command mode.

The syntax for this command is:

zero reporter state-transitions [**all**|**reporter** *reporter_name*]

The variables and options for this command are:

- **all** - Zeroes the State Transitions counter of all reporters configured on the CSS
- **reporter** *reporter_name* - Zeroes the State Transitions counter of the specified reporter

For example, to reset the State Transitions counter for reporter r1, enter:

```
(config)# zero reporter state-transitions reporter r1
```

Displaying a Reporter Configuration in the Running-Config

To display a reporter configuration in the running-config, use the **show running-config reporter** command in any mode. This command displays all reporter configurations on the CSS. To display a specific reporter configuration, use the **show running-config reporter** *reporter_name* command.

Displaying Critical Reporter Information

To display critical reporter configuration information for VRID peering and critical phy, use the **show critical-reporter** command in any mode. The syntax for this command is:

show critical-reporters *ip_address vrid*

The variables for this command are:

- *ip_address* - Specifies the interface address of the virtual router associated with the critical-reporter
- *vrid* - Specifies the VRID of the VR associated with the critical reporter

Table 1-13 describes the fields for the **show critical-reporters** command output.

Table 1-13 Field Descriptions for the show critical-reporters Command

Field	Description
Interface Address	IP address of the VR interface.
VRID	Identifier of the VR associated with the reporter.
Reporter Name	Name of the reporter with which the VR is associated.
Reporter Type	Configured type of the reporter. Possible types are vrid-peering, critical-phy-all-up, or critical-phy-any-up.
State	State of the reporter. Possible states are Master, Backup, Up, Down, or Suspended.

Displaying Critical Reporters in the Running-Config

To display configured critical reporters in the running-config, use the **show running-config circuit** command. This command displays all configured circuits and any critical reporters associated with a virtual router.