System Interfaces and Ports

This chapter describes how to create a context and configure system interfaces and ports within the context. Before beginning these procedures, refer to your product-specific administration guide for configuration information for your product.

Contexts

Even though multiple contexts can be configured to perform specific functions, they are all created using the same procedure.

Creating Contexts

Important Commands used in the configuration examples in this section represent the most common or likely commands and/or keyword options. In many cases, other commands and/or keyword options are available. Refer to the *Command Line Interface Reference* for complete information regarding all commands.

To create a context, apply the following example configuration:

```
configure
   context name
      end
```

Repeat to configure additional contexts.

Viewing and Verifying Contexts

Step 1 Verify that your contexts were successfully created by entering the following command:

```
[local]host_name# show context all
```

The output is a two-column table similar to the example below. This example shows that two contexts were created: one named *source* and one named *destination*.

```
Context Name    ContextID    State
------------    ---------    -----
local           1            Active
source          2            Active
destination     3            Active
```

The left column lists the contexts that are currently configured. The center column lists the corresponding context ID for each of the configured contexts. The third column lists the current state of the context.

Step 2 Save your configuration as described in the *Verifying and Saving Your Configuration* chapter.

Step 3 Now that the context has been created, interfaces and specific functionality can be configured within the context. Proceed to other sections for instructions on configuring specific services and options.

Ethernet Interfaces and Ports

Regardless of the type of application interface, the procedure to create and configure it consists of the following:

Step 1 Create an interface and assign an IP address and subnet mask to it by applying the example configuration in Creating an Interface, on page 2.

Step 2 Assign a physical port for use by the interface and bind the port to the interface by applying the example configuration in Configuring a Port and Binding It to an Interface, on page 3.

Step 3 Optionally configure a static route for the interface by applying the example configuration in Configuring a Static Route for an Interface, on page 3.

Step 4 Repeat the above steps for each interface to be configured.

This section provides the minimum instructions for configuring interfaces and ports to allow the system to communicate on the network. Commands that configure additional interface or port properties are described in the *Ethernet Port Configuration Mode Commands* and *Ethernet Interface Configuration Mode Commands* chapters of the *Command Line Interface Reference.*

To ensure that system line card and port-level redundancy mechanisms function properly, the Spanning Tree protocol must be disabled on devices connected directly to any system port. Failure to turn off the Spanning Tree protocol may result in failures in the redundancy mechanisms or service outage.

Creating an Interface

Use the following example to create a new interface in a context:

```
configure
   context name
      interface name
         { ip | ipv6 } address address subnetmask [ secondary ]
         end
```

Notes:

- *Optional:* Add the **loopback** keyword option to the **interface** *name* command, to set the interface type as "loopback" which is always UP and not bound to any physical port.
- *Optional:* Add the **secondary** keyword to the **{ ip | ipv6 } address** command, to assign multiple IP addresses to the interface. IP addresses can be entered using IPv4 dotted-decimal or IPv6 colon-separated-hexadecimal notation.
- *Optional:* In the interface config mode, add the **port-switch-on-L3-fail address** command, to configure the interface for switchover to the port on the redundant line card if connectivity to a specified IP address is lost. This IP address can be entered using IPv4 dotted-decimal or IPv6 colon-separated-hexadecimal notation.

Configuring a Port and Binding It to an Interface

Use the following example configuration to configure and assign a port to an interface:

```
configure
  port ethernet slot#/port#
    description description
    no shutdown
    bind interface interface_name  context_name
    end
```

Notes:

- *Optional:* In the Ethernet Port configuration mode, add the preferred **slot** *slot#* command if you want to specify a port preference.
- Binding associates the port and all of its settings to the named interface.

Configuring a Static Route for an Interface

Use the following example to configure a static route for an interface:

```
configure
  context name
      { ip | ipv6 } route ip_address netmask next-hop gw_address interface_name
    end
```

Notes:

- *ip_address* and *netmask* are the IP address and subnet mask of the target network. This IP address can be entered using IPv4 dotted-decimal or IPv6 colon-separated-hexadecimal notation.
- *gw_address* is the IP address of the default gateway or next-hop route. This IP address can be entered using IPv4 dotted-decimal or IPv6 colon-separated-hexadecimal notation.
- To configure a route to the gateway router, use 0.0.0.0 for the network and mask variables.
- Repeat as needed. Multiple static routes can be configured to the same destination to provide an alternative means of communication in case the preferred route fails.

Viewing and Verifying Port Configuration

Step 1 Verify that your interface configuration settings are correct by entering the following commands:

```
[local]host_name# context context_name
[context_name]host_name# show { ip | ipv6 } interface
```

context_name represents the name of the context in which the interface was created. The output from these commands should be similar to the following example.

In this example an interface named *mgmt1* was configured in the local context.

Example:

In this example an interface named *mgmt1* was configured in the local context.

```
Intf Name:       mgmt1
Intf Type:       Broadcast
IP State:        UP (Bound to 10/11 untagged, ifIndex 285278209)
IP Address:      192.168.100.3        Subnet Mask:     255.255.255.0
Bcast Address:   192.168.100.255      MTU:          1500
Resoln Type:     ARP                  ARP timeout:     3600 secs
Number of Secondary Addresses:  0
Total interface count:         1
```

Step 2 Verify that your port configuration settings are correct by entering the following command:

```
[context_name]host_name# show configuration port slot#/port#
```

Example:

This command produces an output similar to that displayed in the following example that shows the configuration of port 11 in chassis slot 1. In this example, the port is bound to an interface called *rp1* configured in a context called *source*.

```
config
   port ethernet 1/11
      description 11_RP1
      no shutdown
      bind interface rp1 source
      end
```

Step 3 Verify that your static route(s) was configured properly by entering the following command:

```
[context_name]host_name# show ip static-route
```

Example:

This command produces an output similar to that displayed in the following example that shows a static route to a gateway with an IP address of 192.168.250.1.

```
Destination      Nexthop         Protocol  Prec  Cost  Interface
0.0.0.0/0        192.168.250.1    Static      0    0     vNIC1
0.0.0.0/0        192.168.250.1    Static      0    0     rp1 source
```

Step 4 Save the configuration as described in the *Verifying and Saving Your Configuration* chapter.

VLANs

Virtual LANs (VLANs) allow two logically separated networks to use the same physical medium. VLAN segmentation, also called 802.1q tagging, works by appending a tag identifying the VLAN ID to each Ethernet frame.

For information on how to create VLANs to handle specific packet types, see the *VLANs* chapter.

Hypervisors

VLAN usage under KVM is an extension to bridge interface sharing. The difference lies in which interface participates in the bridge set. The physical interfaces (such as eth0, eth1) are bound to the bridge, which is used by each guest. These interfaces carry unmodified packets coming externally or being generated internally, with or without a VLAN ID tag.

VMware supports the use of virtual switches that allow virtual machines on one vSphere host to communicate with each other using the same protocols as physical switches. The vSwitch emulates a traditional physical Ethernet network switch by forwarding frames at the data-link layer. A vSphere host can have numerous virtual switches, each with more than 1,000 internal virtual ports for virtual machines. The vSphere platform supports the vSphere Standard Switch virtual switch configuration at the host level and the vSphere Distributed Switch, a single virtual switch that spans multiple associated hosts.

VLANs and Management Ports

The management interface supports VLAN configuration. This support extends to the local context.

Bulkstats can be sent out an interface other than the normal management interface. This interface also supports VLANs.

You can also configure other OA&M services on separate VLANs.

Yo can assign separate source IP addresses for the OA&M services. OA&M services should not be bound to the same VLAN as service VLANs. Other services include SGi, Gi, Pi, eGTP or other packet core-specific interfaces and services.

This feature is implemented by adding support for the **vlan** command to the management port in the local context. See the example command sequence below.

```
configure
   port ethernet 1/1
      vlan 184
         no shutdown
         bind interface 19/3-UHA foo
```