Configuring TCP/IP for High Availability

Matt Muggeridge
TCP/IP for OpenVMS Engineering

Overview

High availability of the network complements other high availability features associated with OpenVMS clustering. In the OpenVMS network, several high-availability technologies can be used in isolation or combined to provide a high-availability solution to meet a multitude of requirements.

The key to configuring a high availability solution of any kind is careful planning with an ethos of *"keep it simple."* Understand where failures might reasonably occur, what types of failures can be tolerated and what failures cannot be tolerated. It is wise to consider the impact of a catastrophic failure and how taking the appropriate precautions can mitigate the impact on the system.

TCP/IP high availability solutions include:

- failSAFE IP[1] – address failover to alternate interfaces
- IP Cluster Alias – superseded by failSAFE IP
- Load Broker/Metric Server – DNS alias name dynamically updated with available addresses
- LAN Failover[2,3]

This paper describes the key difference between these technologies and the environments that best suit their application.

[1] failSAFE IP is introduced with TCP/IP V5.4, which is in field test at the time of writing.
[2] LAN Failover is introduced with OpenVMS V7.3-2, which is in field test at the time of writing.
[3] LAN Failover is not discussed in detail in this paper. Refer to the OpenVMS V7.3-2 documentation.

Comparing High Availability Technologies

Table 1 briefly describes each of the technologies to help the reader compare the features and choose which solution or combination of solutions is best suited to their environment. Each of the technologies is described in depth in subsequent sections.

Table 1 High Availability Network Technologies

	failSAFE IP	IP Cluster Alias	DNS Alias (Load Broker / Metric Server)	LAN Failover
Protects	All IP addresses	Single IP address designated as the cluster address	DNS Alias with list of most available IP addresses	MAC Address
Protocols	IP only	IP only	IP only	All LAN protocols
Scope	Interfaces within a node or cluster	Single interface per node in a cluster	DNS name lookup	Interfaces within a node
NIC	Independent of interface type	Independent of interface type	Not applicable	DE600 and DEGXA
Load Balancing	All interfaces active, balance outgoing connections, higher throughput	One interface in a cluster is assigned the cluster address, no load balancing	Load share inbound connections across DNS alias addresses	One interface in a node is active others are standby, no load balancing
Detects	Failure and recovery: interface, cable, switch, node	Node failure	Most available nodes	Failed interface, cable, and switch
Addressing	May require additional IP addresses per cluster	Requires an additional address per cluster	Multiple addresses listed for DNS alias	LAN virtual interface address automatically generated
Notes	Monitor at least 3 interfaces on a LAN to avoid phantom failures	Superseded by failSAFE IP	Does not protect against all interface failures in multihomed hosts	
Availability	Introduced with TCP/IP V5.4	Long-time feature of TCP/IP Services	Long-time feature of TCP/IP Services	Introduced with OpenVMS V7.3-2

failSAFE IP

The network interface controller (NIC) is often regarded as a single point of failure (SPOF) in a network. Typical failures include NIC failure, disconnected or broken cable, or a dead port on the switch. failSAFE IP removes the NIC as a SPOF. (failSAFE IP is introduced starting with Version 5.4 of HP TCP/IP Services for OpenVMS.)

Introduction to failSAFE IP

failSAFE IP provides IP address redundancy when the same IP address is configured on multiple interfaces. Only one instance of each IP address is active at any time; the other duplicate IP addresses are in standby mode[4]. Standby IP addresses may be configured on multiple interfaces within the same node or across a cluster. The failSAFE service monitors the health of each interface and takes appropriate action upon detecting interface failure or recovery.

When an interface fails, each active IP address on the failed interface is removed and the standby IP address becomes active. If an address is not configured with a standby, then the address is removed from the failed interface until it recovers. Static routes on the failed interface are also removed and migrated to any interface where their network is reachable.

When an interface recovers, it may request the return of its IP addresses. The IP address is returned when the recovering interface is configured as the *home* interface for one or more addresses. When the *home* interface recovers, it requests that the current holder of the address give it up[5]. (The concept of a *home* interface is discussed in Home Interfaces.)

The current holder of an address will not release an address if it would result in dropped connections, nor if the current holder is also designated as a *home* interface for that address. Management intervention can force the removal of an address.

failSAFE IP Configuration Requirements

Configuring failSAFE IP requires two steps:

1. *Assign the same IP address to multiple interfaces*. Only one instance of that address will be active; all other instances will be in standby mode. For simple configurations, use the TCPIP$CONFIG Core Environment menu to assign an IP address to multiple interfaces; see the TCP/IP Services for OpenVMS *Installation and Configuration* guide for more information. Alternately, use the *ifconfig* utility, which provides a greater degree of management control; see Table 2 for more information.
2. *Enable the failSAFE IP service*, which monitors the health of interfaces and takes appropriate action upon detecting interface failure or recovery. This service is enabled using the TCPIP$CONFIG Optional Components menu.

failSAFE IP Service – Interface Health Monitor

The failSAFE IP service monitors the health of interfaces and upon detecting a failure or recovery will take the appropriate action. The service is enabled using TCPIP$CONFIG and is started and stopped with the TCP/IP Services startup and shutdown procedures. Alternately, it may be started or stopped using:

[4] The OpenVMS distributed lock manager is used to ensure only one instance of an IP address is active across a cluster. An exception to this is any address assigned to the loopback interface *'LO0'.* For instance, the localhost address, 127.0.0.1 must be configured on every node in a cluster. See Management Utilities for more information.
[5] An IP address may be configured with multiple home interfaces. By default, the primary address is configured with its interface marked as a home interface.

```
SYS$STARTUP:TCPIP$FAILSAFE_STARTUP.COM

SYS$STARTUP:TCPIP$FAILSAFE_SHUTDOWN.COM
```

The failSAFE IP service:

- Monitors the health of interfaces by periodically reading their *"Bytes received"* counter.
- When required, marks an interface as failed or recovered.
- Maintains static routes to ensure they are preserved after interface failure or recovery.
- Logs all messages to TCPIP$FAILSAFE_RUN.LOG. Important events are additionally sent to OPCOM.
- Generates traffic to help avoid *phantom failures,* (see Avoiding Phantom Failures).
- Invokes a customer written command procedure at the transitions marked by an asterisk in Figure 1 below. (Refer to Site-Specific Customization of failSAFE IP for more detail on site-specific command procedures).

The finite state machine for the failSAFE IP service is shown in Figure 1.

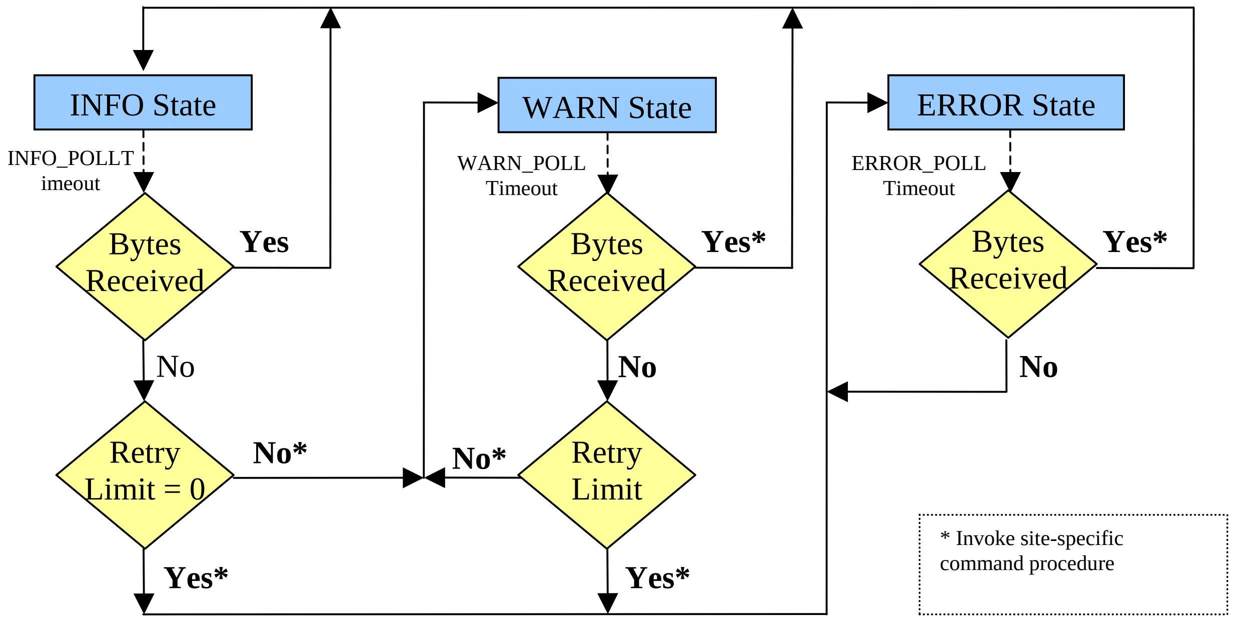


Figure 1 Finite State Machine for failSAFE IP Service – Interface Health Monitor

If the failSAFE IP service is not enabled, then configuring a failSAFE IP address across nodes provides identical functionality to the IP Cluster Alias, described in IP Cluster Alias.

Configuring failSAFE IP Service

By default, the failSAFE IP service monitors all TCP/IP interfaces on a system and periodically polls each interface using default polling intervals. The defaults may be overridden by editing the configuration file defined by the logical name TCPIP$FAILSAFE, which by default is:

SYS$SYSDEVICE:[TCPIP$FSAFE]TCPIP$FAILSAFE.CONF[6]

The configurable parameters are:

Parameter	Description
INTERFACE_LIST [Default: ALL interfaces]	The list of interfaces that failSAFE monitors.
INFO_POLL [Default: 3 seconds]	The polling interval used when the interface is known to be functional. Two INFO_POLL timeouts are required to determine that an interface is not responding, at which time the polling frequency is set to the WARN_POLL period.
WARN_POLL [Default: 2 seconds]	The polling interval used when the interface first stops responding. Polling will continue for RETRY_WARN attempts before the interface is deemed dead, at which time the polling frequency is set to ERROR_POLL and failover occurs.
RETRY_WARN [Default: 1 retry]	The number of warning polls before the interface is deemed dead and the IP addresses associated with it are removed. A value of zero will skip the WARN_POLL cycle.
ERROR_POLL [Default: 15 seconds]	The polling interval used when the interface is considered dead. failSAFE IP will monitor a dead interface at this frequency until it determines the interface has recovered, at which time the polling frequency is set back to the INFO_POLL period.

Detectable Failures

The failSAFE IP service periodically reads the network interface card's (NIC's) "*Bytes received*" counter to determine the health of an interface. This is the same counter that can be viewed using LANCP. For example, to view all interfaces' *"Bytes received"* counters:

```
$ pipe mcr lancp show device/count | search sys$pipe "Bytes received"/exact
```

failSAFE IP guards against any event that prevents the *"Bytes received"* counter from changing or any event that results in the deletion of an IP address, such as:

- Interface hardware failure
- Physical link disconnect
- Shuttng the interface down using TCP/IP management commands
- Shutting down TCP/IP Services

[6] A template file is provided when the service is configured via TCPIP$CONFIG. The template also describes the file syntax.

- Shutting down a node

Application

All environments that require high availability of IP addresses can benefit from failSAFE IP. There are two failure scenarios to consider:

1. If the IP address migrates to an interface on the same node, existing traffic-flow continues uninterrupted and both incoming and outgoing connections are maintained.
2. If the IP address migrates to an interface on another cluster member, existing connections are dropped. However remote clients will be able to establish new connections immediately. In this scenario, failSAFE IP addresses are better suited to UDP applications or those applications that permanently cache IP addresses. It is up to the network administrator whether to configure failSAFE IP addresses across interfaces within the same node or across cluster members.

failSAFE IP will always preferentially fail over addresses to interfaces on the same node before failing over across clustered nodes.

Management Utilities

For many situations, failSAFE IP requires no additional management beyond the initial configuration with TCPIP$CONFIG. This section describes new management commands that are used by the failSAFE IP service – or by system administrators who need to manually intervene with IP address assignment.

A failSAFE IP address may be configured using TCPIP$CONFIG, or manually using the TCPIP management commands. For instance to create an IP address of 10.10.10.1 on interface IE0 and a standby alias address on interface IE1 (pseudo-interface IEB0) the following commands may be used (the *ifconfig* command is shown for comparison):

```
$ TCPIP
TCPIP> SET INTERFACE IE0/HOST=10.10.10.1     ! ifconfig ie0 10.10.10.1
TCPIP> SET INTERFACE IEB0/HOST=10.10.10.1    ! ifconfig ie1 alias 10.10.10.1
```

To view the standby addresses, it is necessary to use the *ifconfig* command. For example:

```
$ ifconfig -a
IE0: flags=c43<UP,BROADCAST,RUNNING,MULTICAST,SIMPLEX>
   *inet 10.10.10.1 netmask ff000000 broadcast 10.255.255.255

IE1: flags=c03<UP,BROADCAST,MULTICAST,SIMPLEX>
     failSAFE IP Addresses:
        inet 10.10.10.1 netmask ff000000 broadcast 10.255.255.255 (on HUFFLE IE0)
```

Note that interface IE1 displays a failSAFE IP address, and that it is active on node HUFFLE, interface IE0.

Greater control of failSAFE IP addresses can be achieved with the *ifconfig* command. The *ifconfig* options that support failSAFE IP are described in Table 2.

Table 2 New ifconfig Options for failSAFE IP

Option	Description
[-]fail	Force an interface to fail by using the *fail* option or to recover by using the *–fail* option.
[-]home	Used when creating IP addresses. By default, all primary IP addresses are created with a home interface. To force an alias address to be created with a home interface, the *home* option must be used.
[-]fs	All IP addresses are created as failSAFE addresses by default, except for addresses assigned to the loopback interface *LO0,* for instance, the localhost address 127.0.0.1. To create an address that is not managed by failSAFE, use the *-fs* option.

Home Interfaces

failSAFE IP addresses may be created with a designated *home* interface. By default, all primary IP addresses are created with a *home* interface. The purpose of a home interface is to provide a preferential failover and recovery target in an effort to always migrate IP addresses to their home interface. This gives the network administrator greater control over how IP addresses are assigned to interfaces. The *ifconfig* management utility may be used to create and display addresses configured with home interfaces. For example to create three addresses:

```
$ ifconfig ie0 10.10.10.1            ! primary has home interface by default
$ ifconfig ie0 alias 10.10.10.2      ! alias does not
$ ifconfig ie0 home alias 10.10.10.3 ! create alias with home interface
```

Note that the *TCPIP SET INTERFACE* command may also be used to create primary and alias addresses. However, it does not support creation of the *home alias* address. For this, *ifconfig* must be used.

When addresses are displayed with the *ifconfig* utility, those addresses with a home interface are marked with an asterisk (*). For example, displaying the addresses created with the previous commands reveals:

```
$ ifconfig ie0
IE0: flags=c43<UP,BROADCAST,RUNNING,MULTICAST,SIMPLEX>
   *inet 10.10.10.1 netmask ff000000 broadcast 10.255.255.255
    inet 10.10.10.2 netmask ff000000 broadcast 10.255.255.255
   *inet 10.10.10.3 netmask ff000000 broadcast 10.255.255.255
```

The asterisk character indicates that the addresses 10.10.10.1 and 10.10.10.3 have a home interface of IE0. Note that *TCPIP SHOW INTERFACE* does not identify addresses with a home interface.

Creating IP addresses with home interfaces helps to maintain the spread of IP addresses across multiple interfaces. This is important for load-balancing and gaining higher aggregate throughput. In the event a home interface recovers after a failure, the addresses may return to their recovered home interface, thus maintaining the spread of addresses across the available interfaces.

Note that an address will *not* migrate toward a home interface if it will result in dropping TCP/IP connections.

Site-Specific Customization of failSAFE IP

A user-defined procedure may be invoked during selected transitions of the failSAFE IP service's finite state machine. Refer to Figure 1 for the state transitions that invoke the site-specific command procedure. These transitions describe one of three events:

1. When the interface first appears to have stopped responding. This is the first warning that a problem may exist, but no action to failover IP addresses is taken yet.
2. When an attempt to generate traffic on the interface fails. After the retry limit is reached, the interface is deemed dead, and IP addresses will be removed from the interface. Failover occurs.
3. When the interface recovers.

The procedure is called with two string parameters:

```
P1 = TCP/IP Interface Name (e.g. "IE0")
P2 = state ("INFO_STATE", "WARN_STATE", "ERROR_STATE")
```

The site-specific procedure may be defined by the logical name TCPIP$SYFAILSAFE; otherwise the default file is:

```
SYS$MANAGER:TCPIP$SYFAILSAFE.COM
```

Logical Names

The logical names in Table 3 may be used to customize the operating environment of failSAFE IP. These logical names must be defined in the LNM$SYSTEM_TABLE for them to take effect.

Table 3 failSAFE IP Logical Names

Logical Name	Description
TCPIP$FAILSAFE	Configuration file that is read by TCPIP$FAILSAFE during startup. If the logical is not defined then the default configuration file is: SYS$SYSDEVICE:[TCPIP$FSAFE]TCPIP$FAILSAFE.CONF This logical must be defined prior to starting the failSAFE IP service.
TCPIP$FAILSAFE_FAILED_<*ifname*>	This logical is used to simulate a failure for the named interface. The logical is translated each time failSAFE IP reads the LAN counters. The <*ifname*> can be determined using the TCPIP SHOW INTERFACE command.
TCPIP$SYFAILSAFE	The name of a site-specific command procedure, which is invoked when one of three conditions occurs: • Iinterface failure

	• Retry failure • Interface recovery If the logical is not defined, then the default procedure is: SYS$MANAGER:TCPIP$SYFAILSAFE.COM
TCPIP$FAILSAFE_LOG_LEVEL	Controls the volume of log messages sent to OPCOM and the log file. If the logical is undefined or has a value of zero, the default log level is assumed. Larger values are used for debugging. This logical name is translated each time failSAFE IP logs a message.
TCPIP$FSACP_LOG_LEVEL	Controls the volume of log messages sent to OPCOM by the ACP. This logical name should be used only when directed by customer support.

Static and Dynamic Routing

When an interface fails, failSAFE IP removes all addresses and static routes from the failed interface. The static routes are reestablished on every interface where the route's network is reachable. This may result in a static route being created on multiple interfaces and is most often observed with the default route.

Dynamic routing may need to be restarted to ensure the dynamic routing protocol remains current with changes in interface availability. If this is necessary, restart the routing process using the TCPIP$SYFAILSAFE procedure, as described in Site-Specific Customization of failSAFE IP. For example, for GATED:

```
$ TCPIP STOP ROUTING /GATED

$ TCPIP START ROUTING /GATED
```

For GATED users, the configuration supports the *scaninterval* option, which allows you to periodically scan the interfaces to detect any changes. Scanning can be forced by issuing the command:

```
$ TCPIP SET GATED/CHECK_INTERFACES
```

For more information on routing protocols refer to the appropriate section in the TCP/IP Services for OpenVMS *Management Guide*.

Best Practices

These best practices are a guide to assist the network administrator to quickly come up to speed with the various aspects of failSAFE IP by avoiding common pitfalls.

Validating failSAFE IP

Most contemporary networks are highly stable and rarely suffer from the problems that require failSAFE IP. Consequently, for the small number of occasions where failSAFE IP is required, it is critical that it has been previously validated in the environment where it is being deployed. Failure to do this may result in unexpected problems at the critical moment.

Since real failures are rare and sometimes difficult to simulate, the logical name TCPIP$FAILSAFE_FAILED_<*ifname*> has been provided. After configuring failSAFE IP addresses and starting the failSAFE IP service, the validation procedure is as follows:

1. *Establish connections and generate IP traffic*

   Using TELNET or FTP, create incoming and outgoing TCP connections to the multihomed host from inside and outside the subnet. Verify that these connections are established, using the following commands:

```
$ @sys$manager:tcpip$define_commands
$ ifconfig -a  ! Check the interface addresses
$ netstat -nr  ! Check the routing table
$ netstat -n   ! identify which interface(s) are being used
```

2. *Simulate a failure and observe*

   Simulate a failure and observe OPCOM and log file messages. The failure may be simulated with:

```
$ define/system tcpip$failsafe_failed_<ifname> 1  ! or disconnect the cable
```

   Wait long enough for failover to occur, which will be signaled by OPCOM messages. Now observe the effects of failover and verify TCP connections are still established and can transfer data. For example, TELNET sessions should respond to keyboard input.

```
$ ifconfig -a  ! Observe how the addresses have migrated
$ netstat -nr  ! Observe how the routing table has changed
```

3. *Recover and observe*
   Recover from the simulated failure and observe the OPCOM messages.

```
$ deassign/system tcpip$failsafe_failed_<ifname>  ! or reconnect the cable
$ ifconfig –a ! Observe how the addresses have migrated
$ netstat –nr ! Observe how the routing table has changed
```

Once again, ensure TCP connections are still established and can transfer data

Be aware that simulating a failure with the logical *TCPIP$FAILSAFE_FAILED_<ifname>* does not disrupt physical connections to the machine, and as such is not a true indicator of whether the services will survive a real failover situation. Consequently, this procedure should be repeated by physically removing a network cable from one or more of the interfaces. Since this may potentially be disruptive to network services, this operation should be scheduled into a maintenance period where a disruption may be tolerated.

Configuring failSAFE IP Service

The key concern for configuring the failSAFE IP service is the time it takes to detect a failure and for the standby IP address to become active. One goal of a failSAFE IP configuration is to avoid disrupting existing connections, so the failover time must be within the connection timeout.

The failover time is calculated as:

```
INFO_POLL + (WARN_POLL * RETRY)  <  failover time  <  (2 * INFO_POLL) + (WARN_POLL * RETRY)
```

Refer to Figure 1 for an explanation of the variables. The default values (INFO_POLL=3, WARN_POLL=2, RETRY=1) result in a failover interval range of between 5 and 8 seconds. Note that this does not take into account the system load.

The recovery time will be less than the ERROR_POLL period, which has a default of 30 seconds. See Configuring failSAFE IP for more information about the failSAFE IP configuration parameters.

Avoiding Phantom Failures

The health of a NIC is determined by monitoring the NIC's *"Bytes received"* counter. This provides a protocol-independent view of the NIC counters. However, in a quiet network, there may be insufficient traffic to keep the *"Bytes received"* counter changing within the *failover detection time*, thus causing a *phantom failure.* To counteract this, the failSAFE service attempts to generate MAC-layer broadcast messages, which are received on every interface on the LAN *except for the sending interface.*

Consequently, in a quiet network with just two interfaces being monitored by the failSAFE service, a single NIC failure may also result in a phantom failure of the other NIC, since the surviving NIC is not able to increase its own "*Bytes received*" counter.

You can reduce phantom failures in a quiet network by configuring the failSAFE IP service for at least three interfaces on the LAN. In the event that one interface fails, the surviving interfaces will continue to maintain each others "*Bytes received*" counter.

Creating IP Addresses with Home Interfaces

By default, the interface on which a primary IP address is created is its home interface, while an IP *alias* address is created without a home interface. To create an alias address with a home interface, use the *ifconfig* command, which should be added to the SYS$STARTUP:TCPIP$SYSTARTUP.COM procedure. For example to create an alias address of 10.10.10.3 on interface IE0 and designate IE0 as its home interface, the following command could be used:

```
$ ifconfig ie0 home alias 10.10.10.3/24
```

Private Addresses Should Not Have Clusterwide Standbys

For the purpose of this discussion, private addresses are those used for network administration and not published as well-known addresses for well-known services. A standby interface for a private address should be configured on the same node as the home interface. This avoids the situation where a node cannot assign any addresses to its interfaces if they have active connections on another node in the cluster. This is further illustrated in Example 2.

If it is desirable to associate the list of private addresses with a public DNS alias name, then it is recommended that the Load Broker be used to provide high availability of the DNS Alias. The Load Broker is described in DNS Alias with Load Broker and Metric Server.

Examples

Example 1 – Single node configured with two interfaces

Consider a node named *HUFFLE* with two interfaces *IE0* and *IE1*. Each interface is configured with a unique primary IP address and each interface is also configured as a standby for each other. The addresses are *10.10.10.1/24* and *10.10.10.2/24*. These addresses and the failSAFE IP standby aliases are easily created using the TCPIP$CONFIG *Core Environment* menu.

For the purpose of clarification, the commands that TCPIP$CONFIG uses are shown in the table below. The identical *ifconfig* commands are shown for comparison.

Configure IP addresses:

Action	TCP/IP Command	*ifconfig* command

Create Primary Addresses	`$ tcpip set interface ie0 – /host=10.10.10.1 - /net=255.255.255.0 - /broad=10.10.10.255`
	`$ tcpip set interface ie1 - /host=10.10.10.2 - /net=255.255.255.0 - /broad=10.10.10.255`
		`$ ifconfig ie0 10.10.10.1/24`
		`$ ifconfig ie1 10.10.10.2/24`

	`$ tcpip set interface iea0 – /host=10.10.10.1 - /net=255.255.255.0 - /broad=10.10.10.255`
	`$ tcpip set interface ieb0 - /host=10.10.10.2 - /net=255.255.255.0 - /broad=10.10.10.255`
		`$ ifconfig ie0 alias 10.10.10.2/24`
		`$ ifconfig ie1 alias 10.10.10.1/24`

At this point, the node will be configured as shown in Figure 2.

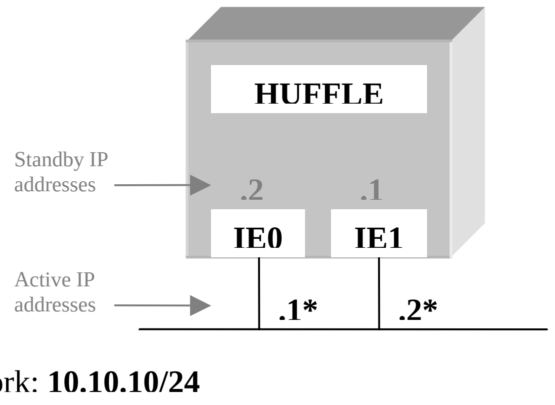


Figure 2 Simple failSAFE IP Configuration

Examining the configuration with *ifconfig* reveals how each interface is configured with an active primary address as well as a standby failSAFE IP address. The asterisk beside the address denotes that address's home interface. In the sample output below, note that the standby failSAFE IP addresses also describe where the IP address is active. For example, for interface IE0, the standby address 10.10.10.2 is active on node *HUFFLE*, interface *IE1.* The asterisk before the address indicates that the respective interface is its home interface. Home interfaces are described in more detail in Home Interfaces.

```
$ ifconfig ie0
IE0: flags=8000c43<UP,BROADCAST,RUNNING,MULTICAST,SIMPLEX>
     failSAFE IP Addresses:
        inet 10.10.10.2 netmask ffffff00 broadcast 10.10.10.255 (on HUFFLE IE1)
   *inet 10.10.10.1 netmask ffffff00 broadcast 10.10.10.255

$ ifconfig ie1
IE1: flags=c43<UP,BROADCAST,RUNNING,MULTICAST,SIMPLEX>
     failSAFE IP Addresses:
        inet 10.10.10.1 netmask ffffff00 broadcast 10.10.10.255 (on HUFFLE IE0)
   *inet 10.10.10.2 netmask ffffff00 broadcast 10.10.10.255
```

In the event of an interface failure (for example, IE0 fails), the failSAFE IP service marks the interface as failed, using the following command:

```
$ ifconfig ie0 fail
```

Figure 3 shows the state of the node after IE0 has failed. The *ifconfig* commands are also shown below. Note that interface IE1 is now configured with both addresses and the output from *ifconfig ie0* shows that the interface is in a failed state.

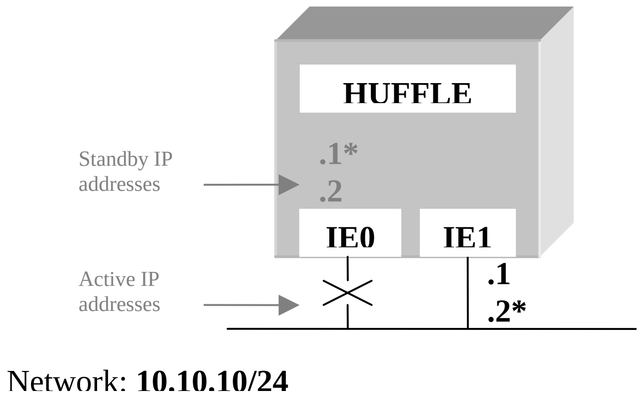


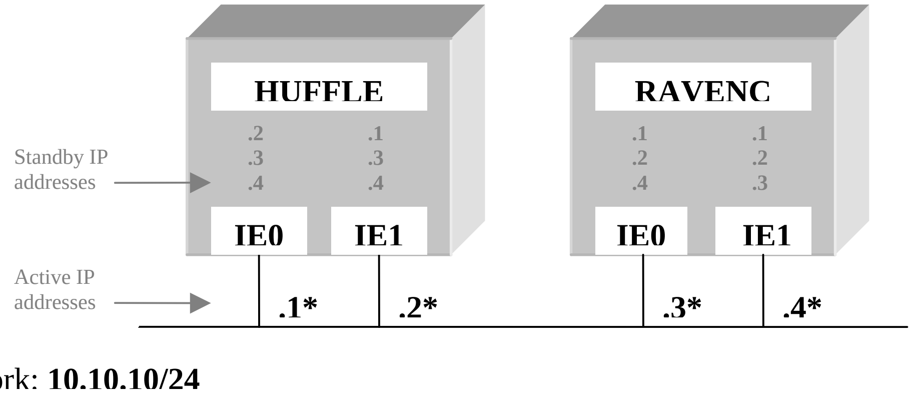
Figure 3 Interface IE0 has failed

```
$ ifconfig ie0
IE0: flags=8000c43<UP,BROADCAST,RUNNING,MULTICAST,SIMPLEX>
    *failSAFE IP - interface is in a failed state.
     failSAFE IP Addresses:
        inet 10.10.10.2 netmask ffffff00 broadcast 10.10.10.255 (on HUFFLE IE1)
       *inet 10.10.10.1 netmask ffffff00 broadcast 10.10.10.255 (on HUFFLE IE1)

$ ifconfig ie1
IE1: flags=c43<UP,BROADCAST,RUNNING,MULTICAST,SIMPLEX>
    inet 10.10.10.1 netmask ffffff00 broadcast 10.10.10.255
   *inet 10.10.10.2 netmask ffffff00 broadcast 10.10.10.255
```

Example 2 – Clustered Nodes configured with Two Interfaces

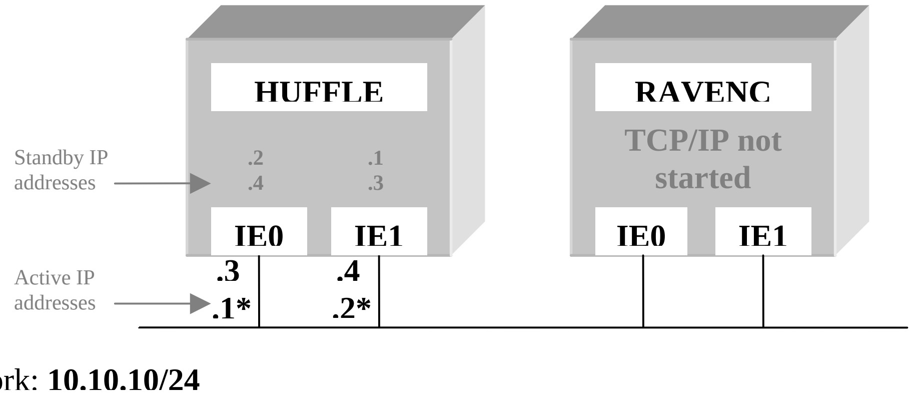
Extending the previous example to two similarly configured nodes in an OpenVMS cluster:



Executing an *ifconfig* command on node HUFFLE reveals information about all failSAFE IP addresses that are configured on HUFFLE. Note that IE0 is the home interface for address 10.10.10.1, as indicated by the asterisk in the diagram and the asterisk in the output below.

```
$ ifconfig ie0
IE0: flags=8000c43<UP,BROADCAST,RUNNING,MULTICAST,SIMPLEX>
     failSAFE IP Addresses:
        inet 10.10.10.2 netmask ffffff00 broadcast 10.10.10.255 (on HUFFLE IE1)
        inet 10.10.10.3 netmask ffffff00 broadcast 10.10.10.255 (on SLYTHE IE0)
        inet 10.10.10.4 netmask ffffff00 broadcast 10.10.10.255 (on SLYTHE IE1)
   *inet 10.10.10.1 netmask ffffff00 broadcast 10.10.10.255
```

Consider the situation where RAVENC is booted after HUFFLE starts TCP/IP Services. Before TCP/IP Services is started on RAVENC, all addresses are active on node HUFFLE. For instance, the figure above may become:



Note that this figure shows the *.3* and *.4* addresses being distributed among the interfaces on HUFFLE. In practice, this is indeterminate.

When TCP/IP Services is started on node RAVENC, it requests that its home addresses be returned. In this example, the *.3* and *.4* addresses have their home interface on RAVENC, so RAVENC requests that HUFFLE release the *.3* and *.4* addresses so that RAVENC can assign them.

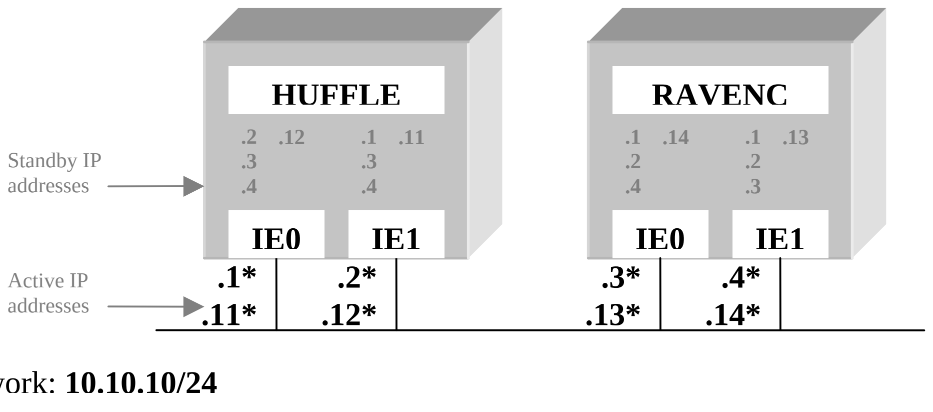
However, node HUFFLE will only release these addresses provided it does not have any outstanding connections to them. This situation could result in node RAVENC being started without any IP addresses being configured. To avoid this, only configure primary addresses with standby interfaces on the *same* node. See Example 3 for an alternate configuration that avoids this problem.

Example 3 – Preferred failSAFE IP Configuration – Putting it all together

Greater demands in availability of IP addresses require reconsideration of how addresses are assigned. One possibility is to create private maintenance IP addresses as primary addresses, and public IP address as aliases[7]. The disadvantage of this is that more IP addresses are required for the dedicated maintenance addresses. The advantage is that there is greater control and flexibility over address assignment. The concept of tying an address to a specific interface becomes less of a concern.

For instance, to build upon the previous example, private maintenance addresses could be assigned as primary addresses, and the public addresses would be configured as aliases, where each alias has at most one home interface. Consider the private primary addresses to be *.11, .12, .13,* and *.14*, and the public aliases to be *.1, .2, .3, and .4,* as shown below.

Note that the private primary addresses have standby addresses configured on the same node only. For example, on node HUFFLE, *.11* and *.12* are configured on both interfaces, but they are not configured on node RAVENC. The public alias addresses (*.1, .2, .3, and .4*) have addresses configured on each interface across both nodes. The asterisk beside these denotes the address's home interface. Thus, IE0 on node HUFFLE is the home interface for the alias 10.10.10.1.



[7] For the purpose of this example, consider the private maintenance addresses to be known only to the network administrator, whereas the public addresses are well-known and provide connectivity to well-known services.

To configure this, create the primary IP addresses using the TCPIP$CONFIG procedure, and create the alias addresses using *ifconfig*. For example, add the following lines to the TCPIP$SYSTARTUP.COM procedure:

On node HUFFLE:

```
$! Configure home aliases
$ ifconfig ie0 home alias 10.10.10.1/24
$ ifconfig ie1 home alias 10.10.10.2/24

$! Configure IE0 aliases (presumes .12 primary was created via TCPIP$CONFIG)
$ ifconfig ie0 alias aliaslist 10.10.10.2-4,12/24

                                                   created via TCPIP$CONFIG)
$ ifconfig ie1 alias aliaslist 10.10.10.1,3,4,11/24
```

On node RAVENC:

```
$! Configure home aliases
$ ifconfig ie0 home alias 10.10.10.3/24
$ ifconfig ie1 home alias 10.10.10.4/24

$! Configure IE0 aliases (presumes .14 primary was created via TCPIP$CONFIG)
$ ifconfig ie0 alias aliaslist 10.10.10.1,2,4,14/24

                                                    reated via TCPIP$CONFIG)
$ ifconfig ie1 alias aliaslist 10.10.10.1-3,13/24
```

IP Cluster Alias

The IP Cluster Alias provides a subset of the functionality provided by failSAFE IP (see failSAFE IP), and as such has been superseded by failSAFE IP. However, failSAFE IP is introduced in TCP/IP Services Version 5.4, whereas the IP Cluster Alias was introduced in UCX Version1.0. It is recommended that existing users of the IP Cluster Alias update their configuration to use failSAFE IP.

Introduction to IP Cluster Alias

In an OpenVMS cluster, there may be a single IP address designated to represent selected cluster members. This address is known as the IP Cluster Alias address. Each interface still has its own unique IP address while the IP Cluster Alias is an additional address that can be active on only one interface in the cluster at a time. The node holding the address is designated as the *Cluster Impersonator* and as such will field all connections to the Alias address. In the event of a failure, the IP Cluster Alias address will be reassigned to one of the remaining cluster members interfaces.

Note that the functionality provided by the IP Cluster Alias is a subset of that provided by failSAFE IP. IP Cluster Alias is supported for compatibility.

IP Cluster Alias Configuration Requirements

Configuring the IP Cluster Alias requires that the node be an active member of an OpenVMS cluster at the time TCP/IP Services is configured using the TCPIP$CONFIG procedure. The configuration procedure will detect this scenario and when configuring the interfaces, will ask if a cluster address should be assigned.

Only one interface in the cluster can hold the cluster alias address at any time. It is recommended that the cluster alias address be configured in the same subnet as the unique interface addresses. This ensures broadcast traffic to the subnet containing the interfaces will also appear on the IP Cluster Alias address.

Detectable Failures

The types of failures that are detected include:

- Shutting the interface down using TCP/IP management commands
- Shutting down TCP/IP Services
- Shutting down the node

Application

This form of failover provides high availability for incoming connections. In the event of a failure the IP Cluster Alias migrates across nodes. Existing TCP connections will abort and need to be reestablished. Connectionless protocols, such as UDP, will be unaffected.

There is no load-balancing across nodes with this mechanism. The cluster impersonator fields all incoming connections. Outgoing connections do not make use of the IP Cluster Alias. It is best suited to UDP applications like NFS, or for maintaining a high availability IP address where load-balancing of incoming connections is not a priority.

Management Utilities

To identify the node currently acting as the impersonator, enter the following command:

```
$ TCPIP SHOW INTERFACE/CLUSTER
```

The node acting as the impersonator will be labeled as "*Cluster Impersonator*". This needs to be performed on each node in the cluster. An easy way to do this is with the SYSMAN utility:

```
$ MCR SYSMAN
SYSMAN> SET ENVIRONMENT/CLUSTER
SYSMAN> DO PIPE TCPIP SHOW INTERFACE/CLUSTER | SEARCH SYS$PIPE IMPERSONATOR
```

Example

In Figure 3, the IP Cluster Alias address is designated as, 10.10.10.100. Node HUFFLE is currently the impersonator, and RAVENC is the standby node. In the event HUFFLE is shut down, then RAVENC will assume the IP Cluster Alias address and become known as the impersonator. New traffic to the IP Cluster Alias will go through RAVENC. Note that each node still has its own unique interface address, 10.10.10.1 and 10.10.10.3. The alias address 10.10.10.100 can be active on only one of the nodes at any time.

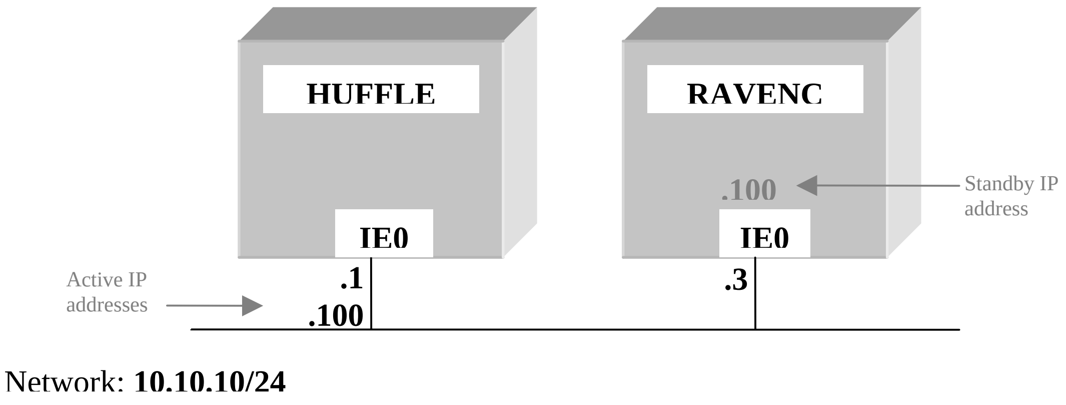


Figure 3 - IP Cluster Alias Configuration

Replacing the IP Cluster Alias with the failSAFE IP mechanism requires that the IP Cluster Alias first be deleted and the failSAFE IP address created. For instance, executing the following command on each node would delete the IP Cluster Alias:

```
$ tcpip set configuration interface ie0/nocluster  ! remove from config file
$ tcpip set interface ie0/nocluster                ! remove from active system
```

Now configure the 10.10.10.100 address on each node with a command similar to:

```
$ tcpip set configuration interface iea0 /host=10.10.10.100 /net=255.255.255.0
$ tcpip set interface iea0 /host=10.10.10.100 /net=255.255.255.0
```

DNS Alias with Load Broker and Metric Server

This solution provides high availability of the DNS alias by dynamically updating the alias name with the list of most-available IP addresses associated with that alias name. It requires a name server that supports dynamic updates, and the cooperation of DNS administrators to allow your Load Broker to dynamically update their databases[8].

Introduction to DNS Alias

The Load Broker polls nodes for their metric values and dynamically updates the DNS alias with the list of least loaded IP addresses. In this way, whenever a remote host requests a DNS name lookup, it will be presented with the list of IP addresses associated with the least loaded addresses. If a node does not respond with a metric value after 3 attempts, the Load Broker will remove that node's IP addresses from the DNS alias.

[8] Rather than convincing your central DNS administrators to allow you to dynamically update their DNS repository, it may be easier to have them delegate a sub-domain that you administer.

DNS Alias Configuration Requirements

The DNS Alias configuration requires a DNS server that supports dynamic updates and the cooperation of DNS administrators to allow your Load Broker to send dynamic updates to their database. The Load Broker is typically configured on a separate host and located in the network path used by the clients. The DNS Alias is comprised of a list of IP addresses. There is no requirement for these addresses to appear on OpenVMS clustered nodes.

The DNS Alias with the Load Broker and Metric Server are described in detail in the following subsections.

DNS Alias

The DNS Alias, by itself, does not provide high availability. This section provides an overview of the DNS alias and how it is updated by the Load Broker.

The Domain Name System provides a distributed repository for mapping between DNS alias names and IP addresses. In the repository, a single DNS Alias name may be associated with multiple IP addresses. Each time the DNS server is queried for an alias name, the list of IP addresses associated with that name is returned. That list is rotated in a round-robin fashion for each request, so that subsequent requests will return the list with a different IP address at the top of the list. Since applications typically choose the first IP address in the list, the round-robin feature provides load sharing, but not load balancing, across the list of IP addresses.

For example, consider the DNS Alias name *"hogwarts"* with four IP addresses associated with it. The IP addresses may all be associated with the same node, or the addresses may be spread across multiple nodes. The DNS entry in the forward lookup database may be:

```
hogwarts        IN      A       10.10.10.1
                IN      A       10.10.10.2
                IN      A       10.10.10.3
                IN      A       10.10.10.4
```

The first time a client queries the DNS server for the name *"hogwarts"*, the address list returned will be ordered as *(10.10.10.1, 10.10.10.2, 10.10.10.3, 10.10.10.4).* The client will use the first address in the list, and so connect to the 10.10.10.1 address. The next DNS query for "*hogwarts"* will result in the list being returned to the client as *(10.10.10.2, 10.10.10.3, 10.10.10.4, 10.10.10.1)*. This client will once again use the first address in the list and so connect to the 10.10.10.2 address. The pattern will continue for subsequent DNS requests. This round-robin effect can be observed with repeated queries using the *nslookup* utility. For example, notice the IP address list is rotated in the second query:

```
$ nslookup hogwarts
Server:  ns1.wizardry.edu
Address:  10.10.10.200

Name:    hogwarts.wizardry.edu

Addresses:  10.10.10.1, 10.10.10.2, 10.10.10.3, 10.10.10.4

$ nslookup hogwarts
Server:  ns1.wizardry.edu
Address:  10.10.10.200

Name:    hogwarts.wizardry.edu

Addresses:  10.10.10.2, 10.10.10.3, 10.10.10.4, 10.10.10.1
```

In the event of a failure where an IP address becomes unavailable, DNS will continue to dutifully answer queries and rotate the list of IP addresses. Each time the failed IP address is at the top of the list, a client application will not be able to connect, and as a result there will appear to be intermittent connection failures. To guard against this type of failure, the Load Broker and Metric Server may be used[9]. The Load Broker will remove any unresponsive IP addresses from the DNS repository and so provide high availability of the DNS alias name. The round-robin function will continue to share the load across each of the available IP addresses. The Load Broker may be further configured to maintain a maximum number of IP addresses in the DNS alias and it will update DNS with the IP addresses that return the more favorable metric.

The Load Broker and Metric server are discussed in the next section.

Load Broker and Metric Server

The Load Broker is configured to monitor selected IP addresses on hosts where the Metric Server is enabled. The Metric Server responds with a metric value, indicating the load of that machine. If there is no response from a Metric Server after 3 attempts, then the Load Broker will dynamically update the DNS repository excluding the unresponsive IP address. In addition, when a node becomes heavily loaded, it may be replaced in the list by a node with a more favorable metric.

Detectable Failures

The types of failures that are detected include:

- Shutting the interface down using TCP/IP management commands
- Shutting down TCP/IP Services
- Shutting down the node
- Path lost between Load Broker and Metric Server

Application

The Load Broker benefits incoming connections only. It has the additional benefit that the IP addresses with the more favorable metric will be associated with the DNS Alias. This configuration is suited to maintaining high availability and optimum performance for a well-known service that is

[9] failSAFE IP may also be used to provide high availability of IP addresses across clustered nodes. Use the Load Broker in situations where it is not desirable for an IP address to failover across clustered nodes. Load Broker does not require IP addresses to be within an OpenVMS cluster.

distributed across multiple nodes. There is no requirement for the nodes to be part of an OpenVMS Cluster. In order to be effective, the client application must retranslate the host name of the server following a failure. Applications that do not repeat the DNS query (such as many NFS clients) will never see the updated list of alias addresses.

Management Utilities

The metric value for any host on a LAN can be displayed with the *metricview* utility. For example:

```
$ @tcpip$define_commands
$ metricview
Host                                                    Rating
----                                                    ------
10.10.10.1      huffle-e0                                 136
10.10.10.3      ravenc-e0                                  51
```

The more favorable node is represented by a larger metric value.

Example

Placement of the Load Broker node is important when configuring the network. Since it can detect connectivity between the Load Broker and Metric Server, it is best placed in the same network path used by the clients that access the services. Similarly, the Load Broker should not be configured on a machine running the Metric Server, since it will always report full connectivity to that Metric Server, regardless of the state of the network paths to the clients. An example Load Broker configuration file for is shown below:

```
cluster "hogwarts.wizardy.edu"
{
        dns-ttl 45;
        dns-refresh 31;
        masters { 10.10.10.200; };
        polling-interval 10;
        max-members 3;
        members { 10.10.10.1; 10.10.10.2; 10.10.10.3; 10.10.10.4; };
        failover 10.10.0.150;
};
```

This configuration file indicates that four IP addresses participate in the load-balancing, but only three of these addresses, (*max-members*), will participate in the DNS alias, thus excluding the node with the least favored metric from the DNS alias. Note that Load Broker will only dynamically update the DNS alias if the alias must be modified with a different set of addresses. Load Broker does not compare the order of the DNS alias list with its current metric order because DNS will continue to adjust the order, providing load sharing amongst the DNS alias members.

The Load Broker will poll each Metric Server every 10 seconds (*polling-interval*). If a Metric Server does not respond after 3 polling intervals (30 seconds), then on the next *dns-refresh* timeout (31 seconds) the bad IP address will be excluded from the next dynamic update. The *dns-ttl* will force intermediate name-servers that cache the results of a DNS-query to time out this entry every 45 seconds. In this way, if a failure occurs, a client will take 45 seconds at most to retry a connection before DNS queries the primary server for the new DNS alias list.

This form of high availability is applicable to new incoming connections to a well-known service distributed amongst participating nodes. If a failure occurs during a connection, that connection will need to close and a new connection be established.

Summary

High availability of the network requires careful consideration of the network environment and understanding of the failures that must be protected against. As a result, one or more high availability solutions may be required. failSAFE IP provides high availability of IP addresses for both incoming and outgoing new connections as well as existing traffic flow. The DNS Alias with Load Broker and Metric Server provides high availability of a DNS Alias name and so benefits incoming connections only. The IP Cluster Alias has been superseded by failSAFE IP. LAN Failover provides high availability of a hardware MAC address and benefits all LAN protocols. LAN Failover is required for LAN protocols that do not provide a failover solution, such as LAT. Protocols such as DECnet-Plus, SCS, and IP implement a failover solution.

The various high availability solutions described in this paper require minimal configuration and management. However, since they are protecting vital parts of the network, any solution must be validated prior to being relied upon in a production environment.

For more information

For more information, contact HP OpenVMS products at http://www.hp.com/go/openvms.