



BIG-IP® Link Controller Reference Guide

version 4.3

MAN-0054-00

Product Version

This manual applies to version 4.3 of the BIG-IP® Link Controller.

Legal Notices

Copyright

Copyright 1997-2002, F5 Networks, Inc. All rights reserved.

F5 Networks, Inc. (F5) believes the information it furnishes to be accurate and reliable. However, F5 assumes no responsibility for the use of this information, nor any infringement of patents or other rights of third parties which may result from its use. No license is granted by implication or otherwise under any patent, copyright, or other intellectual property right of F5 except as specifically described herein. F5 reserves the right to change specifications at any time without notice.

Trademarks

F5, F5 Networks, BIG-IP, 3-DNS, SEE-IT, EDGE-FX, the F5 logo, and the GLOBAL-SITE logo are registered trademarks of F5 Networks, Inc. in the U.S. and certain other countries. FireGuard, iControl, GLOBAL-SITE, Internet Control Architecture, and IP Application Switch are trademarks of F5 Networks, Inc. In Japan, the F5 logo is trademark number 4386949, BIG-IP is trademark number 4435184, 3-DNS is trademark number 4435185, and SEE-IT is trademark number 4394516. All other product and company names are registered trademarks or trademarks of their respective holders.

Export Regulation Notice

This product may include cryptographic software. Under the Export Administration Act, the United States government may consider it a criminal offense to export this product from the United States.

Export Warning

This is a Class A product. In a domestic environment this product may cause radio interference in which case the user may be required to take adequate measures.

FCC Compliance

This equipment generates, uses, and may emit radio frequency energy. The equipment has been type tested and found to comply with the limits for a Class A digital device pursuant to Part 15 of FCC rules, which are designed to provide reasonable protection against such radio frequency interference.

Operation of this equipment in a residential area may cause interference, in which case the user at his own expense will be required to take whatever measures may be required to correct the interference.

Any modifications to this device, unless expressly approved by the manufacturer, can void the user's authority to operate this equipment under part 15 of the FCC rules.

Canadian Regulatory Compliance

This class A digital apparatus complies with Canadian I CES-003.

Standards Compliance

The product conforms to ANSI/UL Std 1950 and Certified to CAN/CSA Std. C22.2 No. 950.

Acknowledgments

This product includes software developed by the University of California, Berkeley and its contributors.

This product includes software developed by the Computer Systems Engineering Group at the Lawrence Berkeley Laboratory.

This product includes software developed by the NetBSD Foundation, Inc. and its contributors.

This product includes software developed by Christopher G. Demetriou for the NetBSD Project.

This product includes software developed by Adam Glass.

This product includes software developed by Christian E. Hopps.

This product includes software developed by Dean Huxley.

This product includes software developed by John Kohl.

This product includes software developed by Paul Kranenburg.

This product includes software developed by Terrence R. Lambert.

This product includes software developed by Philip A. Nelson.

This product includes software developed by Herb Peyerl.

This product includes software developed by Jochen Pohl for the NetBSD Project.

This product includes software developed by Chris Provenzano.

This product includes software developed by Theo de Raadt.

This product includes software developed by David Muir Sharnoff.

This product includes software developed by SigmaSoft, Th. Lockert.

This product includes software developed for the NetBSD Project by Jason R. Thorpe.

This product includes software developed by Jason R. Thorpe for And Communications,
<http://www.and.com>.

This product includes software developed for the NetBSD Project by Frank Van der Linden.

This product includes software developed for the NetBSD Project by John M. Vinopal.

This product includes software developed by Christos Zoulas.

This product includes software developed by Charles Hannum.

This product includes software developed by Charles Hannum, by the University of Vermont and State Agricultural College and Garrett A. Wollman, by William F. Jolitz, and by the University of California, Berkeley, Lawrence Berkeley Laboratory, and its contributors.

This product includes software developed by the University of Vermont and State Agricultural College and Garrett A. Wollman.

In the following statement, "This software" refers to the Mitsumi CD-ROM driver: This software was developed by Holger Veit and Brian Moore for use with "386BSD" and similar operating systems.

"Similar operating systems" includes mainly non-profit oriented systems for research and education, including but not restricted to "NetBSD," "FreeBSD," "Mach" (by CMU).

In the following statement, "This software" refers to the parallel port driver: This software is a component of "386BSD" developed by William F. Jolitz, TeleMuse.

This product includes software developed by the Apache Group for use in the Apache HTTP server project (<http://www.apache.org/>).

This product includes software developed by Darren Reed. (© 1993-1998 by Darren Reed).

This product includes software licensed from Richard H. Porter under the GNU Library General Public License (© 1998, Red Hat Software), www.gnu.org/copyleft/lgpl.html.

This product includes the standard version of Perl software licensed under the Perl Artistic License (© 1997, 1998 Tom Christiansen and Nathan Torkington). All rights reserved. You may find the most current standard version of Perl at <http://www.perl.com>.

This product includes software developed by Eric Young.

This product includes Malloc library software developed by Mark Moraes. (© 1988, 1989, 1993, University of Toronto).

This product includes open SSL software developed by Eric Young (ey@cryptsoft.com), (© 1995-1998).

This product includes open SSH software developed by Tatu Ylonen (ylo@cs.hut.fi), Espoo, Finland (© 1995).

This product includes open SSH software developed by Niels Provos (© 1999).

This product includes SSH software developed by Mindbright Technology AB, Stockholm, Sweden, www.mindbright.se, info@mindbright.se (© 1998-1999).

This product includes free SSL software developed by Object Oriented Concepts, Inc., St. John's, NF, Canada (© 2000).

This product includes software developed by Object Oriented Concepts, Inc., Billerica, MA, USA (© 2000).

This product includes RRDtool software developed by Tobi Oetiker (<http://www.rrdtool.com/index.html>) and licensed under the GNU General Public License.



Table of Contents

Introduction

Getting started	Intro-1
Choosing a configuration tool	Intro-1
Using the Administrator Kit	Intro-2
Stylistic conventions	Intro-2
Finding additional help and technical support resources	Intro-4
Learning more about the BIG-IP product family	Intro-4

I

Additional Base Network Configuration

Overview of the base network configuration	I-1
Interfaces	I-2
Interface naming conventions	I-2
Displaying status and settings for interfaces	I-2
Media type and duplex mode	I-3
VLANs	I-4
Default VLAN configuration	I-5
Creating, renaming, and deleting VLANs	I-6
Configuring packet access to VLANs	I-7
Managing the Layer 2 forwarding table	I-11
Creating VLAN groups	I-13
Setting up security for VLANs	I-15
Setting fail-safe timeouts for VLANs	I-15
Setting the MAC masquerade address	I-16
Self IP addresses	I-17
Enabling or disabling SNAT automap	I-18
Trunks	I-19
Spanning Tree Protocol (STP)	I-20
Creating and deleting STP domains	I-20
Setting time intervals for an STP domain	I-21
Adding or deleting interfaces in an STP domain	I-21
Disabling and re-enabling an STP domain	I-21
Disabling and re-enabling interfaces in an STP domain	I-21
Restarting stpd	I-22
Port mirroring	I-22
Setting up a port mirror	I-22
Deleting interfaces from a port mirror or deleting a port mirror	I-23

2

Configuring the High-Level Network

Overview of the high-level network	2-1
Pools	2-2
Working with pools	2-3
Pool Name	2-5
Member specification	2-5
Load balancing method	2-5
Persistence	2-21
HTTP redirection	2-28
HTTP header insertion	2-33
Quality of Service (QoS) level	2-34
Type of Service (ToS) level	2-35
Disabling SNAT and NAT connections	2-36
Forwarding pools	2-38
Virtual servers	2-40

Standard virtual servers	2-40
Wildcard virtual servers	2-41
Virtual server options	2-48
Using other Link Controller features with virtual servers	2-58
Nodes	2-59
Services	2-62
Address translation: SNATs, NATs, and IP forwarding	2-65
SNATs	2-65
NATs	2-74
IP forwarding	2-77
Health monitors	2-79
Selecting the monitor template	2-81
Configuring a monitor	2-89
Monitor attributes	2-90
Associating the monitor with a node or nodes	2-96

3

The big3d Agent

Working with the big3d agent	3-1
Collecting path data and server performance metrics	3-1
Understanding factories run by big3d agents	3-2
Tracking LDNS probe states with the big3d agent	3-2
Overview of communications protocols	3-3
Communications for remote administration of the Link Controller	3-3
Communications for resolving DNS requests	3-3
Communications for collecting path data	3-3

4

Internet Link Evaluator

Overview of the Internet Link Evaluator	4-1
Working with the Average Round Trip Time table	4-1
Working with the Average Completion Rate table	4-2
Working with the Average Router Hops table	4-2
Interpreting the Internet Link Evaluator data	4-3

5

Inbound Load Balancing

Working with load balancing modes for inbound traffic	5-1
Understanding inbound load balancing on the Link Controller	5-1
Using static load balancing modes	5-2
Using dynamic load balancing modes	5-4
Configuring inbound load balancing	5-11
Understanding wide IPs	5-11
Understanding wide IP pools	5-12
Defining a wide IP	5-12
Using wildcard characters in wide IP names	5-12
Using the LDNS round robin wide IP attribute	5-13
Setting up load balancing for services that require multiple ports	5-13

6

Monitoring and Administration

Monitoring and administration utilities	6-1
Using the bigpipe utility as a monitoring tool	6-1
Monitoring the Link Controller	6-2
Monitoring virtual servers, virtual addresses and services	6-12
Monitoring nodes and node addresses	6-14
Monitoring NATs	6-14
Monitoring SNATs	6-15
Viewing the status of the interface cards	6-15
Using the Configuration utility for administration and monitoring	6-15
Adding a user	6-15
Customizing the Configuration utility	6-16
Configuring SNMP	6-16
Working with the Syslog utility	6-16
Sample log messages	6-17
Powering down the Link Controller	6-17
Removing and returning items to service	6-17
Removing the Link Controller from service	6-18
Removing individual virtual servers, virtual addresses, and ports from service ...	6-19
Removing individual nodes and node addresses from service	6-20
Viewing the currently defined virtual servers and nodes	6-20
Viewing system statistics and log files	6-20
Viewing system statistics	6-20
Viewing log files	6-21
Printing the connection table	6-21
Changing passwords	6-21
Changing passwords and adding users for the Configuration utility	6-22
Working with the BIG/stat utility	6-23

7

Additional Setup Options

Overview of additional setup options	7-1
Defining additional host names	7-1
Using the MindTerm SSH Console	7-2
Downloading the SSH client to your administrative workstation	7-3
Downloading the SSH client from the web server	7-3
Setting up the SSH client on a Windows 95 or Windows NT workstation	7-3
Setting up the SSH client on a UNIX workstation	7-4
Addressing general networking issues	7-4
Addressing routing issues	7-5
Configuring DNS on the Link Controller	7-8
Configuring email	7-10
Using a serial terminal with the Link Controller	7-12
Configuring a serial terminal in addition to the console	7-13
Configuring a serial terminal as the console	7-13
Forcing a serial terminal to be the console	7-14
Configuring RADIUS or LDAP authentication	7-14
Configuring RADIUS login support	7-15
Configuring LDAP login support	7-16
Allowing multiple authentication styles	7-17
Requiring different authentication styles for different applications	7-18

8

Configuring a Redundant System

Overview of a redundant system	8-1
Synchronizing configurations between units	8-2
Configuring fail-safe settings	8-3
Arming or disarming fail-safe on a VLAN	8-3
Mirroring connection information	8-4
Commands for mirroring	8-5
Mirroring virtual server state	8-6
Mirroring SNAT connections	8-6
Using gateway fail-safe	8-7
Adding a gateway fail-safe check	8-7
Using network-based fail-over	8-9
Setting a specific Link Controller to be the preferred active unit	8-10

9

Configuring SNMP

Introducing SNMP on the Link Controller	9-1
Downloading the MIBs	9-1
Configuring SNMP using the Configuration utility	9-2
Setting up client access	9-3
Configuring system information	9-4
Configuring traps	9-4
SNMP configuration files	9-5
/etc/hosts.deny	9-6
/etc/hosts.allow	9-6
/etc/snmpd.conf	9-7
/etc/snmptrap.conf	9-8
Syslog	9-8
Configuring snmpd to send responses out of different ports or addresses	9-9

A

bigpipe Command Reference

bigpipe commands	A-1
-?	A-3
config	A-4
Options	A-4
Saving configuration files to an archive	A-4
Installing an archived configuration file	A-5
Synchronizing configuration files	A-5
conn	A-6
Displaying all current connections	A-6
Displaying connections for a specific virtual server	A-6
Displaying standby connections	A-6
default_gateway	A-7
Options	A-7
failover	A-8
Options	A-8
Changing failover state	A-8
Displaying failover status	A-8
Initializing failover state	A-8
global	A-9
-h and -help	A-16

interface	A-17
Options	A-17
Displaying interface information	A-18
Setting the media type	A-18
Setting the duplex mode	A-18
Resetting statistics	A-18
Enabling or disabling an interface	A-19
Changing driver name mapping	A-19
load	A-20
maint	A-21
merge	A-22
mirror	A-23
Options	A-23
Displaying port mirroring	A-23
Creating a port mirror	A-23
Deleting interfaces from a port mirror	A-24
Deleting a port mirror	A-24
monitor	A-25
Options	A-25
Creating a monitor	A-26
Modifying a monitor	A-26
Creating a monitor instance	A-26
Modifying a monitor instance	A-28
Deleting a monitor	A-28
Deleting a monitor instance	A-28
Displaying monitor templates	A-28
Displaying monitor instances	A-28
Monitor templates	A-29
-n	A-32
nat	A-33
Options	A-33
Defining a NAT	A-34
Deleting a NAT	A-34
Additional Restrictions	A-34
node	A-35
Options	A-35
Displaying nodes	A-35
Modifying nodes	A-36
pool	A-37
Options	A-37
Displaying a pool	A-38
Creating a pool	A-38
Modifying a pool	A-39
Deleting a pool	A-39
Specifying HTTP redirection	A-39
ratio	A-41
Options	A-41
Displaying ratio settings	A-41
Modifying ratio settings	A-41
reset	A-42
save	A-43
self	A-44
Options	A-44
Creating self IP addresses	A-44
service	A-46
Options	A-46

Table of Contents

---

snat	A-47
Options	A-48
Defining a SNAT	A-48
Deleting SNAT	A-48
stp	A-49
Options	A-49
summary	A-50
trunk	A-51
Options	A-51
unit	A-52
verbose	A-53
verify	A-54
version	A-55
virtual	A-56
Options	A-56
Defining a virtual server using pools and rules	A-57
Defining a virtual server with a wildcard port	A-57
Deleting a virtual server	A-58
vlan	A-59
Options	A-60
vlangroup	A-61
Options	A-62

Glossary

Index



Introduction

- Getting started
- Using the Administrator Kit
- Learning more about the BIG-IP product family

Getting started

Before you start installing the BIG-IP Link Controller, we recommend that you browse the ***BIG-IP Link Controller Solutions Guide*** to find the solution that most closely addresses your needs. Briefly review the basic configuration tasks and the few pieces of information, such as IP addresses and host names, that you should gather in preparation for completing the tasks.

Once you find your solution and gather the necessary network information, turn back to the ***Configuration Worksheet*** and ***Hardware Orientation*** poster for hardware installation instructions, and then return to the ***BIG-IP Link Controller Solutions Guide*** to follow the steps for setting up your chosen solution.

Choosing a configuration tool

The Link Controller offers both web-based and command line configuration tools, so that users can work in the environment with which they are most comfortable.

The Setup utility

The Setup utility guides you through the initial system setup. You can run the Setup utility from the command line, or from a web browser. The Setup utility prompts you to enter basic system information including a root password and the IP addresses that will be assigned to VLANs. For more information, see the ***BIG-IP Controller Solutions Guide***, Chapter 2, *Using the Setup Utility*.

The Configuration utility

The Configuration utility is a web-based application that you use to configure and monitor the setup on the Link Controller. Once you complete the installation instructions described in this guide, you can use the Configuration utility to perform the configuration steps necessary for your chosen load balancing solution. In the Configuration utility, you can also monitor current system performance, and download administrative tools such as the SNMP MIB or the SSH client. The Configuration utility is best viewed with the following browsers: Netscape Navigator® version 4.7, or Microsoft® Internet Explorer version 5.0 or later.

The bigpipe and bigtop command line utilities

The **bigpipe**™ utility is the command line counter-part to the Configuration utility. Using **bigpipe** commands, you can configure virtual servers, open ports to network traffic, and configure a wide variety of features. To monitor the Link Controller, you can use certain **bigpipe** commands, or you can use the **bigtop**™ utility, which provides real-time system monitoring. You can use the command line utilities directly on the Link Controller console, or you can run commands using a remote shell, such as the SSH client

(encrypted communications only), or a Telnet client (if you are restricted by cryptography export laws). For detailed information about the command line syntax, see Appendix A, *bigpipe Command Reference*.

Using the Administrator Kit

The Link Controller Administrator Kit provides all of the documentation you need to work with the BIG-IP Link Controller. The Link Controller includes the following printed documentation.

- ◆ **Hardware Orientation Poster**

This poster includes information about the Link Controller unit. It also contains important environmental warnings.

- ◆ **Configuration Worksheet**

This worksheet provides you with a place to plan the basic configuration for the Link Controller.

The Link Controller also includes a resource CD-ROM that provides the following guides, in PDF format. These guides are also available from the Welcome screen when you log in to the Configuration utility on the Link Controller.

- ◆ **BIG-IP Link Controller Solutions Guide**

This guide provides examples of common link load balancing solutions using the Link Controller. Before you begin installing the hardware, we recommend that you browse this guide to find the load balancing solution that works best for you.

- ◆ **BIG-IP Link Controller Reference Guide**

This guide provides detailed configuration information for the Link Controller. It also provides syntax information for **bigpipe** commands, other command line utilities, configuration files, system utilities, and monitoring and administration information.

Stylistic conventions

To help you easily identify and understand important information, this section describes the stylistic conventions used in our documentation.

Using the solution examples

All examples in this documentation use only non-routable IP addresses. When you set up the solutions we describe, you must use IP addresses suitable to your own network in place of our sample addresses.

Identifying new terms

To help you identify sections where a term is defined, we show the term itself in bold italic text. For example, a ***virtual server*** is a specific combination of a virtual address and virtual port, associated with a content site that is managed by a Link Controller or other type of host server.

Identifying references to objects, names, and commands

We apply bold text to a variety of items to help you easily pick them out of a block of text. These items include web addresses, IP addresses, utility names, and portions of commands, such as variables and keywords. For example, with the **bigpipe pool <pool_name> show** command, you can specify a specific pool to show by specifying a pool name for the **<pool_name>** variable.

Identifying references to other documents

We use italic text to denote a reference to another document. In references where we provide the name of a book as well as a specific chapter or section in the book, we show the book name in bold, italic text, and the chapter/section name in italic text to help quickly differentiate the two. For example, you can find information about **bigpipe** commands in Appendix A, *bigpipe Command Reference*.

Identifying command syntax

We show complete commands in bold Courier text. Note that we do not include the corresponding screen prompt, unless the command is shown in a figure that depicts an entire command line screen. For example, the following commands show the configuration of the specified pool name:

bigpipe pool <pool_name> show

or

b pool <pool_name> show

Table Intro.1 explains additional special conventions used in command line syntax.

Item in text	Description
\	Indicates that the command continues on the following line, and that users should type the entire command without typing a line break.
< >	Identifies a user-defined parameter. For example, if the command has <your name> , type in your name, but do not include the brackets.
	Separates parts of a command.
[ ]	Indicates that syntax inside the brackets is optional.
...	Indicates that you can type a series of items.

Table Intro.1 *Command line syntax conventions*

Finding additional help and technical support resources

You can find additional technical information about this product using the following resources:

- ◆ **Release notes**

Release notes for the current version of this product are available from the product web server home page, and are also available on the technical support site. The release notes contain the latest information for the current version, including a list of new features and enhancements, a list of fixes, and, in some cases, a list of known issues.

- ◆ **Online help**

You can find help online in three different locations:

- The web server on the product has PDF versions of the guides included in the Administrator Kit.
- The web-based Configuration utility has online help for each screen. Simply click the **Help** button.
- Individual **bigpipe** commands have online help, including command syntax and examples, in standard UNIX™ man page format. Simply type the command followed by the word **help**, and the Link Controller displays the syntax and usage associated with the command.

- ◆ **Third-party documentation for software add-ons**

The Welcome screen in the Configuration utility and the resource CD-ROM both contain online documentation for all third-party software, such as GateD.

- ◆ **Technical support through the World Wide Web**

The F5 Networks Technical Support web site, <http://tech.F5.com>, provides the latest technical notes, answers to frequently asked questions, updates for Administrator Kit guides (in PDF format), and the Ask F5 natural language question and answer engine. To access this site, you need to obtain a customer ID and a password from the F5 Help Desk.

Learning more about the BIG-IP product family

The BIG-IP platform offers many different software systems. The platform can run on a stand-alone system, or on a redundant system, with the exception of the BIG-IP e-Commerce Controller, which is only available as a stand-alone system. You can easily upgrade from any special-purpose BIG-IP to the BIG-IP HA software, which supports all BIG-IP features.

- ◆ **The BIG-IP**

The BIG-IP HA, and HA+ software provides the full suite of local area load balancing functionality. The BIG-IP unit also has an optional 3-DNS software module which supports wide-area load balancing.

- ◆ **The BIG-IP e-Commerce Controller**

The BIG-IP e-Commerce Controller uses SSL acceleration technology to increase the speed and reliability of the secure connections that drive e-commerce sites.

- ◆ **The BIG-IP Link Controller**

The BIG-IP Link Controller uses metrics and thresholds to manage inbound and outbound traffic through multiple gateways (routers) and Internet Service Providers (ISPs).

- ◆ **The BIG-IP special purpose products**

The special purpose BIG-IP provides the ability to choose from three different BIG-IP feature sets. When you run the Setup utility, you specify one of three types:

- **The BIG-IP Load Balancer**

The BIG-IP Load Balancer provides basic load balancing features.

- **The BIG-IP FireGuard**

The BIG-IP FireGuard provides load balancing features that maximize the efficiency and performance of a group of firewalls.

- **The BIG-IP Cache Controller**

The BIG-IP Cache Controller uses content-aware traffic direction to maximize the efficiency and performance of a group of cache servers.



I

Additional Base Network Configuration

- Overview of the base network configuration
- Interfaces
- VLANs
- Self IP addresses
- Trunks
- Spanning Tree Protocol (STP)
- Port mirroring

Overview of the base network configuration

Setting up the base network for the Link Controller means configuring elements such as the Link Controller host name, a default gateway pool, interface media settings, VLANs, and self IP addresses. You perform configuration tasks for the Link Controller base network using the Link Controller Setup utility. For information on using the Setup utility, see Chapter 2, *Using the Setup Utility*, in the **BIG-IP Link Controller Solutions Guide**.

Once you have configured the base network elements with the Setup utility, you might want to further enhance the configuration of these elements. This chapter provides the information you need to perform these additional configuration tasks. You can perform these tasks using either the Configuration utility or the **bigpipe** utility.

Elements you might want to configure further after running the Setup utility are:

- ◆ **Interfaces**

You can set the media type and the duplex mode for an interface, as well as display interface status.

- ◆ **VLANs**

VLAN options include tagging and assigning interfaces to VLANs. In addition, you can group separate VLANs together for the purpose of bridging packets between them.

- ◆ **Self IP addresses**

You can change self IP addresses or create any number of additional self IP addresses for a VLAN.

- ◆ **Trunks**

Trunks are aggregated links. In link aggregation, interfaces can be combined into a trunk to increase bandwidth, in an additive manner. The other benefit of link aggregation is link fail-over. If one link in a trunk goes down, traffic is simply redistributed over the remaining links.

- ◆ **Spanning Tree Protocol (STP)**

STP domains provide for loop resolution in configurations where one or more external switches is connected in parallel with a Link Controller.

- ◆ **Port mirroring**

This allows you to copy traffic from any interface or set of interfaces on a Link Controller to a single, separate interface. Typically you would install a sniffer device on the target port for debugging and/or monitoring.

- ◆ **Note**

---

*Once you have configured the base network, you configure the high-level network for load balancing. Examples of elements you configure as part of the high-level network are: Pools, rules, proxies, and network address translation (SNATs and NATs). For information on how to configure your high-level network, see Chapter 2, **Configuring the High-Level Network**.*

Interfaces

A Link Controller can have as few as two network interfaces and as many as twenty-nine. Before you perform configuration tasks such as displaying interface status and settings, setting the media type, and setting the duplex mode, it is helpful to understand interface naming conventions.

Interface naming conventions

By convention, the Ethernet interfaces on a Link Controller take the name **<s>.<p>** where **s** is the slot number of the NIC, and **p** is the port number on the NIC. For the Application Switch, slot numbering is left-to-right, and port numbering is top-to-bottom, as shown in Figure 1.1. Note that slot 2 is used for the gigabit ports, and slot 3 is a dedicated administrative port.

When a **bigpipe** command calls for a list of interfaces, the list may consist of one or more interfaces, with multiple interfaces separated by spaces. For example:

2.1 2.2 2.4 2.6

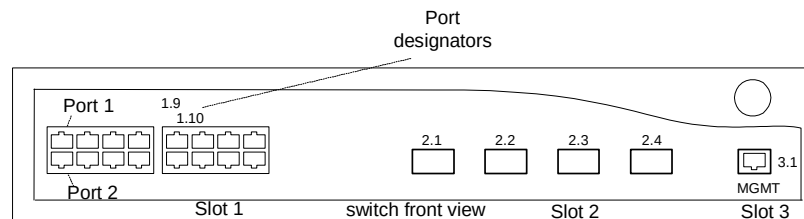


Figure 1.1 Application Switch slot and port numbering

Displaying status and settings for interfaces

Use the following command to display the current status and the settings for all installed interface cards:

bigpipe interface show

Figure 1.2 is an example of the output you see when you issue this command on an active/standby unit in active mode.

interface	speed	pkts	pkts	pkts	pkts	bits	bits	errors	trunk	STP
	Mb/s	in	out	drop	coll	in	out			
5.1 UP	100 HD	0	213	0	0	0	74.2K	0		
4.1 UP	100 HD	20	25	0	0	28.6K	33.9K	0		

Figure 1.2 The *bigpipe interface show* command output

Use the following command to display the current status and the setting for a specific interface.

```
bigpipe interface <if_name> show
```

Media type and duplex mode

Properties that are configurable on the interfaces include media type and duplex mode, as shown in Table 1.1.

Interface Properties	Description
media	You may specify a media type or use auto for automatic detection.
duplex	You may specify a full or half duplex mode, or use auto for automatic selection.

Table 1.1 The attributes you can configure for an interface

Setting the media type

You can set the media type to the specific media type for the interface card or to **auto** for auto detection. If the media type is set to **auto** and the card does not support auto detection, the default type for that interface is used, for example **100BaseTX**.

Use the following syntax to set the media type:

```
b interface <if_name> media <media_type> | auto
```

(The default media type is **auto**.)

◆ Note

*If the Link Controller is inter-operating with an external switch, the media setting should match that of the switch. For more information, see **Setting the interface media type**, in Chapter 2, **Using the Setup Utility**, in the **BIG-IP Link Controller Solutions Guide**.*

Setting the duplex mode

You can set the duplex mode to full duplex or half duplex. If the media type does not allow duplex mode to be set, an onscreen message indicates this. If the media type is set to **auto**, or if setting the duplex mode is not supported for the interface, the duplex setting is not saved to the **bigip_base.conf** file.

Use the following command to set the duplex mode:

```
b interface <if_name> duplex full | half | auto
```

(The default mode is **auto**.)

VLANs

A **VLAN** is a grouping of separate networks that allows those networks to behave as if they were a single local area network, whether or not there is a direct ethernet connection between them.

Link Controller offers several options that you can configure for a VLAN. Table 1.2 summarizes the VLAN configuration options.

Option	Description
Create a default VLAN configuration	Use the Setup utility to create a default VLAN configuration.
Create, rename, or delete VLANs	Create, rename, or delete a VLAN.
Configure packet access to VLANs	Using VLAN tagging, you can direct packets from multiple VLANs to a specific Link Controller interface, or direct traffic from a single VLAN to multiple interfaces.
Manage the L2 forwarding table	You can edit the L2 forwarding table to enter static MAC address assignments.
Create VLAN groups	Create a VLAN group to enable layer 2 packet forwarding between VLANs.
Set VLAN security	For each VLAN, you can restrict the ports using port lockdown.
Set fail-safe timeouts	You can set a failsafe timeout on a VLAN. You can use a failsafe timeout to trigger fail-over in a redundant system.
Set self IP addresses	You can set one or more self IP addresses for VLANs.
Set MAC masquerade	You can use the MAC masquerade to set up a media access control (MAC) address that is shared by a redundant system.

Table 1.2 Configuration options for VLANs

Default VLAN configuration

The Setup utility automatically configures each interface on the Link Controller as a member of a VLAN. The Link Controller identifies the fastest interfaces, makes the lowest-numbered interface in that group a member of the VLAN **external**, and makes all remaining interfaces members of the VLAN **internal**. This creates the default mapping shown in Figure 1.3.

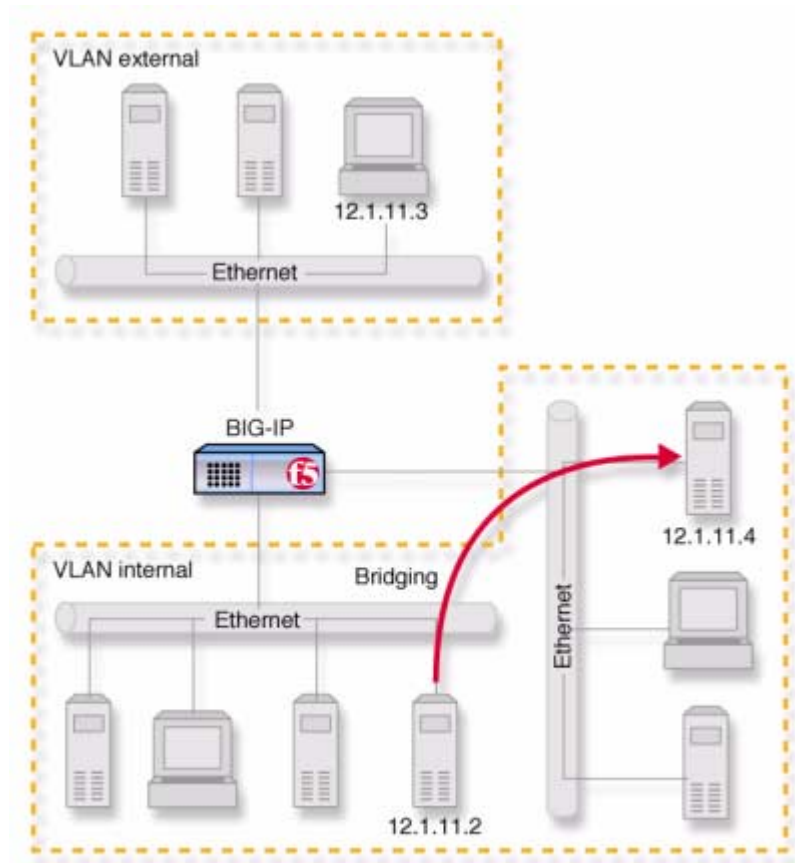


Figure 1.3 Default VLAN configuration

As Figure 1.3 shows, VLAN flexibility is such that separate IP networks can belong to a single VLAN, while a single IP network can be split among multiple VLANs. (The latter case enables you to install the Link Controller into an existing LAN without renaming the nodes.) The VLANs named **external** and **internal** are separate networks, and in the configuration shown they behave like separate networks. The networks belonging to the VLAN **internal** are also separate networks, but have been made to behave like a single network. This is accomplished using a feature called VLAN bridging.

Creating, renaming, and deleting VLANs

Typically, if you use the default configuration, the Setup utility automatically assigns one VLAN to each interface. However, if you need to change your network configuration, or if the default VLANs are not adequate for a network configuration, you can create new VLANs, rename existing VLANs, or delete a VLAN.

To create a VLAN using the Configuration utility

1. In the navigation pane, click **Network**.
The VLANs screen opens.
2. Click the **Add** button.
3. Type the attributes for the VLAN.
4. Click **Done**.

To rename or delete a VLAN using the Configuration utility

1. In the navigation pane, click **Network**.
The VLANs screen opens.
2. In the VLANs screen, use one of the following methods:
  - To rename a VLAN, click the VLAN name you want to change. The VLAN properties screen opens. Type the new name in the **VLAN name** box.
  - To delete a VLAN, click the **Delete** button for the VLAN you want to delete.
3. Click **Done**.

To create, rename, or delete a VLAN from the command line

To create a VLAN from the command line, use the following syntax:

```
b vlan <vlan name> interfaces add <if name> <if name>
```

For example, if you want to create a VLAN named **myvlan** that contains the interfaces **1.1** and **1.2**, type the following command:

```
b vlan myvlan interfaces add 1.1 1.2
```

To rename an existing VLAN, use the following syntax:

```
b vlan <vlan name> rename <new vlan name>
```

For example, if you want to rename the VLAN **myvlan** to **yourvlan**, type the following command:

```
b vlan myvlan rename yourvlan
```

To delete a VLAN, use the following syntax:

```
b vlan <vlan name> delete
```

For example, to delete the VLAN named **yourvlan**, type the following command:

```
b vlan yourvlan delete
```

Configuring packet access to VLANs

The Link Controller supports two methods for sending and receiving packets through an interface that is a member of one or more VLANs. These two methods are:

- ◆ **Port-based access to VLANs** - Packets are accepted for a VLAN because the packets have no tags in their headers and were received on an interface that is a member of a VLAN. With this method, an interface is configured as an *untagged* member of the VLAN. Packets sent out through untagged interfaces contain no tag in their header.
- ◆ **Tag-based access to VLANs** - Packets are accepted for a VLAN because the packets have tags in their headers and the tag matches the VLAN identification number for the VLAN. With this method, an interface is configured as a *tagged* member of the VLAN. Packets sent out through tagged interfaces contain a tag in their header.

The method used by a VLAN is determined by the way that you add a member interface to a VLAN. When creating a VLAN or modifying VLAN properties (using the Configuration utility or the **bigpipe** command), you can add an interface to that VLAN as either an untagged or a tagged interface.

The following two sections describe these two methods of providing packet access to a VLAN.

Port-based access to VLANs

Port-based access to VLANs occurs when an interface is added to a VLAN as an untagged interface. In this case, the interface can be added only to that VLAN and to no others. This limits the interface to accepting traffic only from that VLAN, instead of from multiple VLANs. To solve this problem, Link Controller allows you to configure a feature known as tagging, described in the following section.

Tag-based access to VLANs

Tag-based access to VLANs occurs when an interface is added to a VLAN as a tagged interface. A tagged interface can be added to multiple VLANs, thereby allowing the interface to accept traffic from each VLAN of which the interface is a member.

When you add an interface to a VLAN as a tagged interface, the Link Controller associates the interface with the VLAN identification number, or *tag*, which becomes embedded in the header of a packet.

◆ **Note**

Every VLAN has a VLAN identification number. This identification number is assigned to a VLAN either explicitly, when a user creates the VLAN, or automatically by the Link Controller, if the user does not supply one.

Each time you add an interface to a VLAN, either when creating a VLAN or modifying its properties, you can designate that interface as a tagged interface. A single interface can therefore have multiple tags associated with it.

The result is that whenever a packet comes into that interface, the interface reads the tag that is embedded in a header of the packet. If the tag in the packet matches any of the tags associated with the interface, the interface accepts the packet. If the tag in the packet does *not* match any of the tags associated with the interface, the interface rejects the packet.

Example

Figure 1.4, shows the difference between using three untagged interfaces (where each interface must belong to a separate VLAN) versus one tagged interface (which belongs to multiple VLANs).

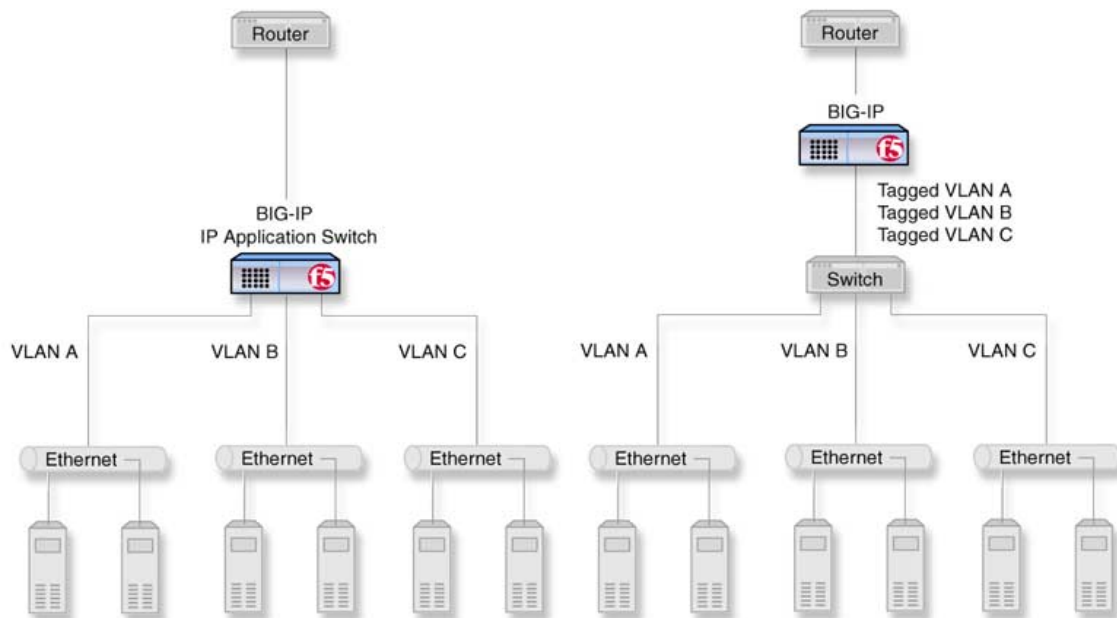


Figure 1.4 Equivalent solutions using untagged and tagged interfaces

The configuration on the left shows a Link Controller unit with three internal interfaces, each a separate, untagged interface. This is a typical solution for supporting three separate customer sites. In this scenario, each interface can only accept traffic from its own VLAN.

Conversely, the configuration on the right shows a Link Controller with one internal interface and an external switch. The switch places the internal interface on three separate VLANs. The interface is configured on each VLAN as a tagged interface. In this way, the single interface becomes a tagged member of all three VLANs, and accepts traffic from all three. The configuration on the right is the functional equivalent of the configuration on the left.

Not only can you add a single, tagged interface to multiple VLANs, as shown in the above example, you can also add multiple tagged interfaces to a single VLAN.

Configuration procedures

You configure tag-based access to VLANs using either the Configuration utility or the **bigpipe vlan** command. You can configure tag-based access either when you create a VLAN and add member interfaces to it, or by modifying the properties of an existing VLAN. In the latter case, you simply change the status of one or more member interfaces from untagged to tagged.

To create a VLAN that supports tag-based access using the Configuration utility

Creating a VLAN that supports tag-based access means creating the VLAN and then adding one or more tagged interfaces to it.

1. In the navigation pane, click **Network**.
The VLAN screen opens.
2. Click the **Add** button.
The Add VLAN screen opens.
3. In the **VLAN Name** box, type a name for the VLAN.
4. In the **Tag** box, you can specify an optional VLAN ID number. If you do not provide one, Link Controller assigns a default number.
5. In the **Resources** box, specify any tagged interfaces by selecting the appropriate interface numbers from the **Interface Number** list and clicking **tagged >>**.
6. Configure the other VLAN options.
7. Click **Done**.

To configure tag-based access on an existing VLAN using the Configuration utility

Configuring tag-based access on an existing VLAN means changing the existing status of one or more member interfaces from **untagged** to **tagged**.

1. In the navigation pane, click **Network**.
The VLAN screen opens.
2. Click the VLAN name in the list.
The properties screen for that VLAN opens.
3. In the **Resources** box, move any untagged interfaces from the **Current Interfaces** list to the **Interface Number** list using the Move button (<<).
4. Specify any tagged interfaces by selecting the appropriate interface numbers from the **Interface Number** list and clicking **tagged >>**.
5. Click **Done**.

To create a VLAN that supports tag-based access from the command line

1. Type the **bigpipe vlan** command, specifying a VLAN name, the **tag** keyword, and a VLAN ID number. The following example creates the VLAN **external** with a VLAN ID of **1209**.

```
b vlan external tag 1209
```

2. Add the interfaces to the VLAN **external** as tagged interfaces. This is done by specifying the VLAN name, the **tagged** keyword, and the interfaces to be tagged. For example:

```
b vlan external interfaces add tagged 4.1 5.1 5.2
```

The effect of this command is to associate a tag with interfaces **4.1** and **5.1**, so that packets with that tag can access the **external** VLAN.

The above procedure adds multiple tagged interfaces to a single VLAN. However, you can also add a single tagged interface to multiple VLANs (similar to the scenario presented in Figure 1.4, on page 1-8). This results in a single interface having more than one tag associated with it. For example, the following commands add the tagged interface 4.1 to the two VLANs **external** and **internal**:

```
b vlan external interfaces add tagged 4.1
```

```
b vlan internal interfaces add tagged 4.1
```

Managing the Layer 2 forwarding table

Layer 2 forwarding is the means by which packets are exchanged directly between nodes on separate VLANs that are members of the same VLAN group, as described in *Creating VLAN groups*, on page 1-13. This is accomplished using a simple forwarding table for each VLAN with **proxy forward** enabled. The forwarding table has an entry for each node in the VLAN and associates the MAC address of that node with the Link Controller interface using the following format:

<MAC address> -> <if>

For example:

00:a0:c9:9e:1e:2f -> 4.1

Viewing and editing the L2 forwarding table

You can view the L2 forwarding table, delete entries, and add static entries. The entries that appear in the table automatically are learned and periodically updated and are called **dynamic entries**. Entries that you add to the table manually are called **static entries**. Static entries are not automatically updated. Entering static entries is useful if you have network devices that do not advertise their MAC addresses.

You can view and edit the L2 forwarding table using the **bigpipe vlan <vlan_name> fdb** command. The **<vlan_name>** may be either a VLAN or a VLAN group.

To view the L2 forwarding table from the command line

Type the following command:

b vlan <vlan name> fdb show

For example:

b vlan internal fdb show

This produces a display such as the following:

Forwarding table --
00:40:05:30:cc:94 -> 5.1)

To view L2 forwarding table static entries from the command line

Type the following command:

b vlan <vlan name> fdb show static

For example:

b vlan internal fdb show static

To view L2 forwarding table dynamic entries from the command line

Type the following command:

```
b vlan <vlan name> fdb show dynamic
```

For example:

```
b vlan internal fdb show dynamic
```

To add an entry to the L2 forwarding table from the command line

Type the following command:

```
b vlan <vlan name> fdb add <MAC address> interface <ifname>
```

For example:

```
b vlan internal fdb add <MAC address> interface <ifname>
```

To delete an entry from the L2 forwarding table from the command line

Type the following command:

```
b vlan <vlan name> fdb delete <MAC address> interface <ifname>
```

For example:

```
b vlan <vlan name> fdb delete 00:a0:c9:9e:1e:2f interface 4.1
vlan <vlan name> fdb show static
vlan <vlan name> fdb show dynamic
vlan <vlan name> fdb show
```

Setting the L2 forwarding aging time

Entries in the L2 forwarding table have a specified life span, after which they are flushed out if the MAC address is no longer present on the network. This process is called the *L2 forward aging time* and you can set it using the global variable **L2 Aging Time**. The default value is 300 seconds.

To set the L2 forwarding aging time using the Configuration utility

1. In the navigation pane, click **System**.
The System Properties screen opens.
2. Click the **Advanced Properties** tab.
3. In **L2 Aging Time** box, enter the aging time in seconds.
4. Click **Done**.

To set the L2 forwarding aging time from the command line

Type the following command:

```
b global l2_aging_time <time_in_seconds>
```

For example:

```
b global l2_aging_time 200
```

Creating VLAN groups

A **VLAN group** is a grouping of two or more VLANs belonging to the same IP network for the purpose of enabling layer 2 packet forwarding, also known as L2 forwarding, between those VLANs. L2 forwarding is the equivalent of bridging where you want communication between VLANs. By creating a VLAN group, nodes on the separate VLANs can exchange packets directly.

In the example shown in Figure 1.3, on page 1-5, VLANs **external** and **internal** represent separate networks that were originally a single network. You can make them behave like a single network again, much like the networks contained in VLAN **internal**. You accomplish this by grouping them as shown in Figure 1.5, on page 1-14.

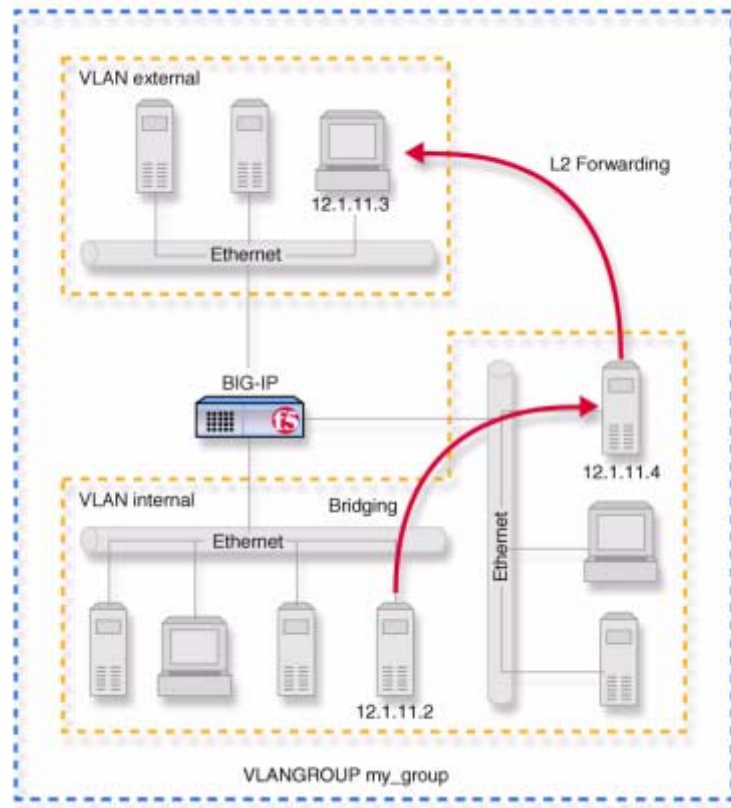


Figure 1.5 VLANs and a VLAN group

To configure a VLAN group to use layer 2 forwarding, you must:

- Create the VLAN group.
- Assign a self IP address to the VLAN group, for routing purposes.
- Verify that layer 2 forwarding (also known as *proxy forwarding*) is enabled.

The following sections describe these procedures.

To create a VLAN group

You can create a VLAN group from the command line using the **vlangroup** command. For example:

```
b vlangroup network11 vlans add internal external
```

To assign the self IP address to the VLAN group

You can assign a self IP address to the VLAN group using the following **bigpipe** command syntax:

```
b self <ip address> vlan <vlangroup name>
```

To verify that Layer 2 forwarding is enabled

Layer 2 forwarding is enabled for the VLAN group using the **vlan proxy_forward** attribute. This attribute is enabled by default when the VLAN group is enabled. To verify that proxy forwarding is enabled, type the following command:

```
b vlan show
```

Setting up security for VLANs

You can lock down a VLAN to prevent direct connection to the Link Controller through that VLAN. You can override this lockdown for specific services by enabling the corresponding global variable for that service. For example:

```
b global open_ssh_port enable
```

To enable or disable port lockdown using the Configuration utility

1. In the navigation pane, click **Network**.
The VLAN screen opens.
2. Click the VLAN name in the list.
The properties screen for that VLAN opens.
3. To enable port lockdown, check the **Port Lockdown** box.
To disable port lockdown, clear the **Port Lockdown** check box.
4. Click **Done**.

To enable or disable port lockdown from the command line

To enable port lockdown, type:

```
b vlan <vlan_name> port_lockdown enable
```

To disable port lockdown, type:

```
b vlan <vlan_name> port_lockdown disable
```

Setting fail-safe timeouts for VLANs

For redundant Link Controller pairs, you can enable a failsafe mechanism that will fail over when loss of traffic is detected on a VLAN, and traffic is not restored during the fail-over timeout period for that VLAN. You can enable a fail-safe mechanism to attempt to generate traffic when half the timeout has elapsed. If the attempt is successful, the fail-over is stopped.

To set the fail-over timeout and arm the fail-safe using the Configuration utility

1. In the navigation pane, click **Network**.
The VLAN screen opens.
2. Click the VLAN name in the list.
The properties screen for that VLAN opens.
3. Check the **Arm Failsafe** box and specify the timeout in seconds in the **Timeout** box.

To set the fail-over timeout and arm the fail-safe from the command line

Using the **vlan** command, you may set the timeout period and also arm or disarm the fail-safe.

To set the timeout, type:

```
b vlan <vlan_name> timeout <timeout_in_seconds>
```

To arm the fail-safe, type:

```
b vlan <vlan_name> failsafe arm
```

To disarm the fail-safe, type:

```
b vlan <vlan_name> failsafe disarm
```

Setting the MAC masquerade address

You can share the media access control (MAC) masquerade address between Link Controller units in a redundant pair. This has the following advantages:

- Increased reliability and failover speed, especially in lossy networks
- Interoperability with switches that are slow to respond to the network changes
- Interoperability with switches that are configured to ignore network changes

The MAC address for a VLAN is the MAC address of the first interface to be mapped to the VLAN, typically 4.1 for external and 5.1 for internal. You can view the interfaces mapped to a VLAN using the following command:

```
b vlan show
```

You can view the MAC addresses for the interfaces on the Link Controller using the following command:

```
b interface show verbose
```

Use the following syntax to set the MAC masquerade address that will be shared by both Link Controller units in the redundant system.

```
b vlan <vlan_name> mac_masq <MAC_addr>
```


Find the MAC address on both the active and standby units, and pick one that is similar but unique. A safe technique for selecting the shared MAC address follows.

Suppose you want to set up **mac_masq** on the external interfaces. Using the **b interface show** command on the active and standby units, you note that their MAC addresses are:

Active: 3.1 = 0:0:0:ac:4c:a2

Standby: 3.1 = 0:0:0:ad:4d:f3

In order to avoid packet collisions, you now must choose a unique MAC address. The safest way to do this is to select one of the addresses and logically **OR** the first byte with **0x40**. This makes the MAC address a locally administered MAC address.

In this example, either **40:0:0:ac:4c:a2** or **40:0:0:ad:4d:f3** would be a suitable shared MAC address to use on both Link Controller units in the redundant system.

The shared MAC address is used only when the Link Controller is in active mode. When the unit is in standby mode, the original MAC address of the network card is used.

If you do not configure **mac_masq** on startup, or when transitioning from standby mode to active mode, the Link Controller sends gratuitous ARP requests to notify the default router and other machines on the local Ethernet segment that its MAC address has changed. See RFC 826 for more details on ARP.

◆ **Note**

*The MAC masquerade information is stored in the **bigip_base.conf** file.*

Self IP addresses

A **self IP address** is an IP address mapping to one or more VLANs and their associated interfaces on a Link Controller. You assign a self IP address to each interface on the unit as part of Setup configuration, and you also assign a floating (shared) alias for units in a redundant pair. (A **floating self IP address** is the address to which the servers behind the Link Controller route traffic). You can create additional self IP addresses for health checking, gateway fail-safe, routing, or other purposes. You can create these additional self IP addresses using the **self** command.

To add a self IP address to a VLAN using the Configuration utility

1. In the navigation pane, click **Network**.
The VLANs screen opens.
2. Click the Self IP Addresses tab.
3. Click the **Add** button.

4. In the **IP Address** box, type the self IP address to be assigned.
5. In the **Netmask** box, type an optional netmask.
6. In the **Broadcast** box, type an optional broadcast address.
7. If you want to configure the self IP address as a floating address, check the **Floating** box.
8. If you want to enable the address for SNAT auto-mapping, check the **SNAT Automap** box.
9. In the **VLAN** box, type the name of the VLAN to which you want to assign the self IP address.
10. Click **Done**.

To add a self IP address to a VLAN from the command line

Use the following syntax:

```
b self <addr> vlan <vlan_name> [ netmask <ip_mask> ][ broadcast <broadcast_addr>] [unit <id>]
```

You can add any number of additional self IP addresses to a VLAN to create aliases. For example:

```
b self 11.11.11.4 vlan external
b self 11.11.11.5 vlan external
b self 11.11.11.6 vlan external
b self 11.11.11.7 vlan external
```

Also, any one self IP address may have **floating** enabled to create a floating alias that is shared by both units of a Link Controller redundant pair:

```
b self 11.11.11.8 floating enable
```

Assigning a self IP address to an interface automatically maps it to the VLAN of which it is a member. Assigning a self IP address to an interface not mapped to an untagged VLAN produces an error message.

Enabling or disabling SNAT automap

The self IP addresses you enable on the external VLAN determine the translation address for SNAT auto-mapping. For more information about SNAT auto-mapping, refer to *Configuring SNAT automapping*, on page 2-69.

Trunks

Link aggregation is the grouping of links (individual physical interfaces) to form a **trunk**. Link aggregation increases the bandwidth of the individual links in an additive manner. Thus, four fast Ethernet links, if aggregated, create a single 400 Mbps link. The other advantage of link aggregation is link fail-over. If one link in a trunk goes down, traffic is simply redistributed over the remaining links.

A trunk must have a controlling link, and acquires all the attributes of that controlling link from layer 2 and above. The trunk automatically acquires the VLAN membership of the controlling link but does not acquire its media type and speed. Outbound packets to the controlling link are load balanced across all of the known-good links in the trunk. Inbound packets from any link in the trunk are treated as if they came from the controlling link.

A maximum of eight links may be aggregated. For optimal performance, links should be aggregated in powers of two. Thus, you ideally will aggregate two, four, or eight links.

To configure a trunk using the Configuration utility

1. In the navigation pane, click **Network**.
The Network screen opens.
2. Click the **Trunks** tab.
The Trunks screen opens.
3. Click the **Add** button.
4. Select the link that is to be the controlling link from the Available Interfaces list, and click **controlling >>**.
The interface appears at the top of the Aggregated Interfaces list.
5. Select the remaining link(s) from the Available Interfaces list and click **aggregated >>**.
The interface(s) appears in the Aggregated Interfaces list below the controlling link.
6. Click **Done**.

To configure a trunk from the command line

Use the following syntax to configure a trunk from the command line:

```
b trunk <controlling_if> define <if_list>
```

Interfaces are specified using the **s.p** convention, where **s** is slot number and **p** is port number. An **<if_list>** is one or more such interfaces, with multiple interfaces separated by spaces.

For more information on interface naming, refer to *Interface naming conventions*, on page 1-2.

Spanning Tree Protocol (STP)

The Link Controller provides Spanning Tree Protocol (STP) implementation for loop resolution in configurations where one or more external switch is connected in parallel with the Link Controller. You can use this feature to configure two or more interfaces on the unit as an STP domain. For interfaces in the STP domain, the spanning tree algorithm identifies the most efficient path between the network segments, and establishes the switch associated with that path as the **root**. Links forming redundant paths are shut down, to be re-activated only if the root fails.

The STP domain should contain all ports that are connected in parallel to an external switch where there are nodes on the link capable of generating or receiving traffic. A second domain is called for if there is an additional switch or switches connected in parallel with additional Link Controller interfaces.

WARNING

Use of STP may slow performance significantly, particularly if more than one STP domain is created, and may have unforeseen effects on complex networks. It is important to test your STP configuration before placing it online.

Creating and deleting STP domains

You can create or delete STP domains using the Configuration utility or from the command line.

To create an STP domain using the Configuration utility

1. In the navigation pane, click **Network**.
The Network screen opens.
2. Click the **STP** tab.
The Trunks screen opens.
3. Click the **Add** button.
4. Configure the STP domain attributes.
5. Click **Done**.

To create or delete an STP domain from the command line

To create an STP domain from the command line, use the following syntax:

```
b stp <stp_name> interfaces add <if _list> | all
```

For example, if you want to create an STP domain named **mystp** that contains the interfaces 1.1 and 1.2, type the following command.

```
b stp mystp interfaces add 1.1 1.2
```

If you want to create an STP domain named **mystp** that contains all interfaces on the Link Controller, type:

```
b stp <stp_name> interfaces add all
```

To delete an STP domain, use the following syntax:

```
b stp <stp_name> delete
```

Setting time intervals for an STP domain

You can set the time intervals in seconds for **hello**, **max_age**, and **forward_delay** for the STP domain from the command line using the following syntax:

```
b stp <stp_name> hello <interval>
```

```
b stp <stp_name> max_age <interval>
```

```
b stp <stp_name> forward_delay <interval>
```

Adding or deleting interfaces in an STP domain

To add interfaces to an STP domain from the command line, use the following syntax:

```
b stp <stp_name> interfaces add <if _list>
```

To delete interfaces from an STP domain, use the following syntax.

```
b stp <stp_name> interfaces delete <if _list>
```

Disabling and re-enabling an STP domain

To disable an STP domain from the command line, use the following syntax:

```
b stp <stp_name> disable
```

To re-enable interfaces on an STP domain, use the following syntax:

```
b stp <stp_name> enable
```

◆ Note

Disabling or deleting all interfaces on an STP domain disables the domain. You cannot re-enable the domain without adding interfaces.

Disabling and re-enabling interfaces in an STP domain

To disable specific interfaces in the STP domain from the command line, use the following syntax:

```
b stp <stp_name> interfaces disable <if _list>
```

To re-enable interfaces in an STP domain, use the following syntax:

```
b stp <stp_name> interfaces enable <if _list>
```

Restarting stpd

The **stpd** does not automatically restart when you synchronize configurations between units in a Link Controller redundant pair. In order to restart the **stpd**, type the following command:

```
bigstart restart stpd
```

Port mirroring

For the IP Application Switch, you can copy traffic from any port or set of ports to a single, separate port. This is called **port mirroring**. You should attach a sniffer device to the target port (called the **mirror-to** port) for debugging and/or monitoring.

Setting up a port mirror

Port mirroring consists of specifying a mirror-to port and adding to it one or more ports (that is, a port list) to be mirrored. You can set up port mirroring using the Configuration utility or from the command line.

To set up port mirroring using the Configuration utility

1. In the navigation pane, click **Network**.
The Network screen opens.
2. Click the **Interfaces** tab.
3. Click the **Port Mirroring** subtab.
4. In the Port Mirroring screen, configure the port mirror attributes.
5. Click **Done**.

To set up port mirroring from the command line

Use this **bigpipe** syntax for setting up port mirroring:

```
b mirror <mirror_to_if> interfaces add <if_list>
```

Example:

```
b mirror 3.24 interfaces add 3.1 3.3 3.10
```

Deleting interfaces from a port mirror or deleting a port mirror

You can delete individual interfaces from a port mirror, or you can completely delete a port mirror.

To delete interfaces from the port mirror using the command line

Use this **bigpipe** syntax to delete interfaces from the port mirror:

```
b mirror <mirror_to_if> interfaces delete <if_list>
```

For example:

```
b mirror 3.24 interfaces delete 3.10
```

To delete the port mirror from the command line

Use this **bigpipe** syntax to delete the port mirror:

```
b mirror <mirror_to_if> delete
```

For example:

```
b mirror 3.24 delete
```




2

Configuring the High-Level Network

- Overview of the high-level network
- Pools
- Virtual servers
- Nodes
- Services
- Address translation: SNATs, NATs, and IP forwarding
- Health monitors

Overview of the high-level network

This chapter describes the elements that make up the *high-level network* of Link Controller. The high-level network is distinct from the *base network*, which is configured with the Setup utility.

Just as the base network is built on the Link Controller interfaces, the high-level network is built on the load balancing pool. The high-level network includes all of the properties associated with pools, as well as virtual servers, and nodes. It can also include pool-selection rules, as well as services, proxies, SNATs, NATs, and health monitor associations for nodes.

- **Pools** represent groups of nodes that can receive traffic from Link Controller according to a specified load balancing method.
- **Virtual servers** can be one of four types: standard, wildcard, network, or forwarding.
- **Services** correspond to the ports (for example, port **80** and port **443**) specified for nodes as they are defined in load balancing pools. Service options include enabling/disabling of service, connection limits, and timeouts for UDP and TCP.
- **SNATs** and **NATs** are secure network address translations and network address translations, respectively, and are used primarily to allow servers to establish outgoing connections as clients.
- **Health monitors** are status checking devices that may be configured by the user, and are associated with nodes for ongoing monitoring.

The remaining sections of this chapter describe each of these elements and the procedures for configuring them for Link Controller.

Pools

A load balancing pool is the primary object in the high-level network. A **pool** is a set of devices grouped together to receive traffic according to a load balancing method. When you create a pool, the members of the pool become visible nodes on the high-level network and can acquire the various properties that attach to nodes. Pools can be accessed through a virtual server, either directly or through a rule, which chooses among two or more pools. The Rules section of this chapter describes several ways to select pools using rules.

You can use the Configuration utility or the **bigpipe pool** command to create, delete, modify, or display the pool definitions on the Link Controller.

When creating a pool, you can configure various pool attributes. Table 2.1 lists the attributes you can configure for a pool.

Pool Attributes	Description	Required or Optional?
Pool name	You can define the name of the pool.	Required
Member specification	You can define each network device, or node, that is a member of the pool.	Required for non-forwarding pools
Load balancing method	You can define a specific load balancing mode for a pool, and you can configure priority-based member activation. Various pools can be configured with different load balancing modes.	Required (Default=Round Robin)
Persistence method	You can define a specific persistence method for a pool. You can have various pools configured with different persistence methods.	Optional
HTTP redirection	You can redirect HTTP requests to a fallback host, protocol, port, or URI path.	Optional
HTTP header insertion	You can configure a pool to insert a header into an HTTP request. For example, the header could include an original client IP address, to preserve the address during a SNAT connection.	Optional
Quality of Service (QoS) level	You can configure a pool to set a specific QoS level within a packet, based on the targeted pool.	Optional
Type of Service (ToS) level	You can configure a pool to set a specific ToS level within a packet, based on the targeted pool.	Optional
Disabling of SNAT and NAT connections	You can configure a pool so that SNATs and NATs are automatically disabled for any connections using that pool.	Optional
Forwarding	You can configure a forwarding pool, which causes a connection to be forwarded, using IP routing, instead of load balanced. Creating a forwarding pool allows you to use pool-based features for traffic that should be forwarded.	Optional

Table 2.1 The attributes of a pool

Working with pools

You can manage pools using either the web-based Configuration utility or the command-line interface. This section describes how to create, delete, modify, or display a pool, using each of these configuration methods.

To create a pool using the Configuration utility

1. In the navigation pane, click **Pools**.
The Pools screen opens.
2. Click the **Add** button.
The Add Pool screen opens.
3. In the Add Pool screen, fill in the fields to create the new pool and configure its attributes.
4. Click **Done**.

To create a pool from the command line

To define a pool and configure its attributes from the command line, use the following syntax:

```
b pool <pool_name> { member <member_definition> ... member <member_definition> }
```

For example, if you want to create the pool **my_pool** with two members, you would type the following command:

```
b pool my_pool { member 11.12.1.101:80 member 11.12.1.100:80 }
```

Use the elements shown in Table 2.2 to construct pools from the command line. These elements correspond to the pool attributes listed in Table 2.1.

Pool Element	Syntax
Pool name	A string from 1 to 31 characters, for example: new_pool
Member definition	member <ip_address>:<service> [ratio <value>] [priority <value>]
lb_method_specification	lb_method [rr ratio fastest least_conn predictive observed ratio_member fastest_member least_conn_member observed_member predictive_member dynamic_ratio]
persist_mode_specification	persist_mode [simple cookie ssl sip sticky msrdp]
fallback_host_specification	fallback <fallback_host>
fallback_protocol_specification	fallback <fallback_protocol>
fallback_port_specification	fallback <fallback_port>
fallback_path_specification	fallback <fallback_path>
header insert	header_insert <quoted string>

Table 2.2 Elements for pool construction

Pool Element	Syntax
link_qos to client level	link_qos to client <level>
link_qos to server level	link_qos to server <level>
ip_tos to client level	ip_tos to client <level>
ip_tos to server level	ip_tos to server <level>
snat disable	snat <ip address> disable
nat disable	nat <ip address> disable
forward	forward

Table 2.2 Elements for pool construction

To delete a pool from the command line

To delete a pool, use the following command:

```
b pool <pool_name> delete
```

You must remove all references to a pool before you can delete a pool.

To modify a pool from the command line

In addition to adding nodes to a pool or deleting nodes from a pool, you can also modify pool attributes. You can add a new member to a pool, change the load-balancing mode, or delete a member from a pool.

For example, to change the default load-balancing mode from Round Robin to Predictive and add two new members to the pool, use a command such as the following:

```
b pool <pool_name> { \  
lb_method predictive \  
member 11.12.1.101:80 \  
member 11.12.1.100:80 }
```

To display one or more pools from the command line

Use the following syntax to display all pools:

```
b pool show
```

Use the following syntax to display a specific pool:

```
b pool <pool_name> show
```

The following sections describe the various pool attributes that you can configure for a pool.

Pool Name

The most basic attribute you can configure for a pool is the pool name. Pool names are case-sensitive, and may contain letters, numbers, and underscores (_) only. Reserved keywords are not allowed.

Each pool that you define must have a unique name.

Member specification

For each pool that you create, you must specify the nodes that are to be members of that pool. Nodes must be specified by their IP addresses.

Load balancing method

Load balancing is an integral part of the Link Controller. Configuring load balancing on the Link Controller means determining your load balancing scenario, that is, which node should receive a connection hosted by a particular virtual server. Once you have decided on a load balancing scenario, you can specify the appropriate load balancing method for that scenario.

The load balancing method is a pool attribute and consists of two properties: the load-balancing mode and priority-based activation.

Load balancing modes

A **load balancing mode** is an algorithm or formula that the Link Controller uses to determine the node to which traffic will be sent. Individual load balancing modes take into account one or more dynamic factors, such as current connection count. Because each application of the Link Controller is unique, and node performance depends on a number of different factors, we recommend that you experiment with different load balancing modes, and select the one that offers the best performance in your particular environment.

The default load balancing mode on the Link Controller is **Round Robin**, which simply passes each new connection request to the next server in line. All other load balancing modes take server capacity and/or status into consideration.

If the equipment that you are load balancing is roughly equal in processing speed and memory, Round Robin mode works well in most configurations. If you want to use the Round Robin mode, you can skip the remainder of this section, and begin configuring other pool attributes that you want to add to the basic pool configuration.

If you are working with servers that differ significantly in processing speed and memory, you may want to switch to Ratio mode or to one of the dynamic modes.

The individual load balancing modes are as follows.

Round Robin

This is the default load balancing mode. Round Robin mode passes each new connection request to the next server in line, eventually distributing connections evenly across the array of machines being load balanced. Round Robin mode works well in most configurations, especially if the equipment that you are load balancing is roughly equal in processing speed and memory.

Ratio

Link Controller distributes connections among machines according to ratio weights that you define, where the number of connections that each machine receives over time is proportionate to a ratio weight you define for each machine. This is a static load balancing mode, basing distribution on static user-assigned ratio weights that are proportional to the capacity of the servers.

Load balancing calculations may be localized to each pool (member-based calculation) or they may apply to all pools of which a server is a member (node-based calculation). Member-based calculation is specified by the extension ***ratio_member***. This distinction is especially important; in Ratio Member mode, the actual ratio weight is a member attribute in the pool definition, whereas in Ratio mode, the ratio weight is an attribute of the node.

Dynamic ratio

Dynamic Ratio mode is like Ratio mode, except that ratio weights are based on continuous monitoring of the servers and are therefore continually changing.

This is a dynamic load balancing mode, distributing connections based on various aspects of real-time server performance analysis, such as the current number of connections per node or the fastest node response time.

Dynamic Ratio mode is used specifically for configuring RealNetworks® RealSystem® Server platforms, Windows® platforms equipped with Windows Management Instrumentation (WMI), or any server equipped with an SNMP agent such as the UC Davis SNMP agent or the Windows 2000 Server SNMP agent. To install and configure the necessary server software for these systems, refer to *Configuring servers and the Link Controller for Dynamic Ratio load balancing*, on page 2-10.

Fastest

Fastest mode passes a new connection based on the fastest response of all currently active nodes. Fastest mode may be particularly useful in environments where nodes are distributed across different logical networks.

Load balancing calculations may be localized to each pool (member-based calculation) or they may apply to all pools of which a server is a member (node-based calculation). The variant of the mode using member-based calculation is distinguished by the extension ***fastest_member***.

Least Connections

Least Connections mode is relatively simple in that the Link Controller passes a new connection to the node that has the least number of current connections. Least Connections mode works best in environments where the servers or other equipment you are load balancing have similar capabilities.

This is a dynamic load balancing mode, distributing connections based on various aspects of real-time server performance analysis, such as the current number of connections per node or the fastest node response time.

Load balancing calculations may be localized to each pool (member-based calculation) or they may apply to all pools of which a server is a member (node-based calculation). The variant of the mode using member-based calculation is distinguished by the extension **least_conn_member**.

Observed

Observed mode uses a combination of the logic used in the Least Connection and Fastest modes. In Observed mode, nodes are ranked based on a combination of the number of current connections and the response time. Nodes that have a better balance of fewest connections and fastest response time receive a greater proportion of the connections. Observed mode also works well in any environment, but may be particularly useful in environments where node performance varies significantly.

This is a dynamic load balancing mode, distributing connections based on various aspects of real-time server performance analysis, such as the current number of connections per node or the fastest node response time.

Load balancing calculations may be localized to each pool (member-based calculation) or they may apply to all pools of which a server is a member (node-based calculation). The variant of the mode using member-based calculation is distinguished by the extension **observed_member**.

Predictive

Predictive mode also uses the ranking methods used by Observed mode, where nodes are rated according to a combination of the number of current connections and the response time. However, in Predictive mode, the Link Controller analyzes the trend of the ranking over time, determining whether a node's performance is currently improving or declining. The nodes with better performance rankings that are currently improving, rather than declining, receive a higher proportion of the connections. Predictive mode works well in any environment.

This is a dynamic load balancing mode, distributing connections based on various aspects of real-time server performance analysis, such as the current number of connections per node or the fastest node response time.

Load balancing calculations may be localized to each pool (member-based calculation) or they may apply to all pools of which a server is a member (node-based calculation). The variant of the mode using member-based calculation is distinguished by the extension **predictive_member**.

Setting the load balancing mode for a pool

You can specify a load balancing mode as a pool attribute when a pool is defined. For information about configuring a pool, see *Working with pools*, on page 2-3. The following example describes how to configure a pool to use Ratio Member load balancing. Note that for Ratio Member mode, in addition to changing the load balancing attribute, you must assign a ratio weight to each member node.

◆ Tip

The default ratio weight for a node is 1. If you keep the default ratio weight for each node in a virtual server mapping, the nodes receive an equal proportion of connections as though you were using Round Robin load balancing.

To configure the pool and load balancing mode using the Configuration utility

1. In the navigation pane, click **Pools**.
The Pools screen opens.
  - If you are adding a new pool, click the **Add** button.
The Add Pool screen opens.
  - If you are changing an existing pool, click the pool name in the **Pools** list.
The Pool Properties screen opens.
2. In the Add Pool screen or Pool Properties screen, configure the pool attributes. For additional information about defining a pool, click the **Help** button.

◆ Note

Round Robin is the default load balancing mode and never needs to be set unless you are returning to it from a non-default mode.

To switch a pool to Ratio Member mode using the Configuration utility

1. In the **Current Members** list, click the member you want to edit.
2. Click the Back button (<<) to pull the member into the resources section.
3. Change or add the ratio value for the member.
4. Click the Add button (>>) to add the member back to the **Current Members** list.
5. Click **Done**.

To switch a pool to Ratio Member mode from the command line

To switch a pool to Ratio Member load balancing, use the **modify** keyword with the **bigpipe pool** command. For example, if you want to change the pool **my_pool** to use the **ratio_member** load balancing mode, and to assign each member its ratio weight, you type the following command:

```
b pool my_pool modify { lb_method ratio_member member 11.12.1.101:80 ratio 1 member
11.12.1.100:80 ratio 3}
```

Setting ratio weights for node addresses

The default ratio setting for any node address is **1**. If you use the Ratio (as opposed to Ratio Member) load balancing mode, you must set a ratio other than **1** for at least one node address in the configuration. If you do not change at least one ratio setting, the load balancing mode has the same effect as the Round Robin load balancing mode.

To set ratio weights using the Configuration utility

1. In the navigation pane, click **Nodes**.
2. In the Nodes list, click the Node Addresses tab.
The Node Addresses screen opens.
3. In the Node Addresses screen, click the **Address** of the node.
The Global Node Address screen opens.
4. In the **Ratio** box, type the ratio weight of your choice.
5. Click the **Apply** button to save your changes.

To set ratio weights from the command line

The **bigpipe ratio** command sets the ratio weight for one or more node addresses:

```
b ratio <node_ip> [<node_ip>...] <ratio weight>
```

For example, the following command sets the ratio weight to **3** for a specific node address:

```
b ratio 192.168.103.20 3
```

◆ Note

The <weight> parameter must be a whole number, and must be equal to or greater than 1.

Displaying ratio weights for node addresses

To display the ratio weights for all node addresses

The following command displays the current ratio weight settings for all node addresses.

b ratio show

The command displays the output shown in Figure 2.1.

192.168.200.51	ratio = 3
192.168.200.52	ratio = 1

Figure 2.1 Ratio weights for node addresses

To display ratio weight for specific node addresses

Use the following syntax to display the ratio setting for one or more node addresses:

b ratio <node_ip> [...<node_ip>] show

Configuring servers and the Link Controller for Dynamic Ratio load balancing

You can configure Dynamic Ratio load balancing on RealNetworks RealSystem Server platforms, Windows platforms equipped with Windows Management Instrumentation (WMI), or any server equipped with an SNMP agent such as the UC Davis SNMP agent or the Windows 2000 Server SNMP agent.

Configuring RealNetwork RealSystem Server platforms

For the RealSystem Server platform, we provide a monitor plugin for the server that gathers the necessary metrics. Configuring a RealSystem Server for Dynamic Ratio load balancing consists of four tasks:

- Installing the monitor plugin on the RealSystem Server
- Configuring a **real_server** health check monitor on the Link Controller
- Associating the health check monitor with the server to gather the metrics
- Creating or modifying the server pool to use Dynamic Ratio load balancing

To install the monitor plugin on the RealSystem Server (Windows version)

Use the following procedure to install the monitor plugin on a Windows RealSystem Server.

1. Download the monitor plugin **F5RealMon.dll** from the Link Controller. The plugin is located in **/usr/contrib/f5/isapi**. (The URL is **https://<bigip_address>/doc/rsplugin/f5realmon.dll**.)
2. Copy **f5realmon.dll** to the RealServer **Plugins** directory. (For example, **C:\Program Files\RealServer\Plugins**.)
3. If the RealSystem Server process is running, restart it.

To install the monitor plugin on the RealSystem Server (UNIX version)

Use the following procedure to install the monitor plugin on a UNIX RealSystem Server.

1. Download the monitor plugin **F5RealMon.so** from the Link Controller. The plugin is located in **/usr/contrib/f5/isapi**. (The URL is **https://<bigip_address>/doc/rsplugin/f5realmon.so**.)
2. Copy **f5realmon.so** to the RealServer **Plugins** directory. (For example, **/home/RealServer/Plugins**)
3. If the RealSystem Server process is running, restart it.

To configure a real_server monitor for the server node

Using the Configuration utility or the **bigpipe** command, create a health-check monitor using the **real_server** monitor template. The **real_server** monitor template is shown in the Figure 2.2.

```
monitor type real_server {
    interval 5
    timeout 16
    dest *.12345
    method "GET"
    cmd "GetServerStats"
    metrics "ServerBandwidth:1.5,CPUPercentUsage,MemoryUsage,
    TotalClientCount"
    agent "Mozilla/4.0 (compatible: MSIE 5.0; Windows NT)"
}
```

Figure 2.2 *real_server monitor template*

The **real_server** monitor template can be used as is, without modifying any of the attributes. Alternatively, you can add metrics and modify metric attribute values. To do this, you need to create a custom monitor. For example:

```
b monitor my_real_server '{ use real_server metrics "ServerBandwidth:2.0" }'
```

The complete set of metrics and metric attribute default values is shown in Table 2.3.

Metric	Default Coefficient	Default Threshold
ServerBandwidth (Kbps)	1.0	10,000
CPUPercentUsage	1.0	80
MemoryUsage (Kb)	1.0	100,000
TotalClientCount	1.0	1,000

Table 2.3 *real_server monitor metrics*

Metric	Default Coefficient	Default Threshold
RTSPClientCount	1.0	500
HTTPClientCount	1.0	500
PNAClientCount	1.0	500
UDPTTransportCount	1.0	500
TCPTransportCount	1.0	500
MulticastTransportCount	1.0	500

Table 2.3 real_server monitor metrics

The metric coefficient is a factor determining how heavily the metric's value counts in the overall ratio weight calculation. The metric threshold is the highest value allowed for the metric if the metric is to have any weight at all. To understand how to use these values, it is necessary to understand how the overall ratio weight is calculated. The overall ratio weight is the sum of relative weights calculated for each metric. The relative weights, in turn, are based on three factors:

- the value for the metric returned by the monitor
- the coefficient value
- the threshold value

Given these values, the relative weight is calculated as follows:

$$w = ((\text{threshold} - \text{value}) / \text{threshold}) * \text{coefficient}$$

You can see that the higher the coefficient, the greater the relative weight calculated for the metric. Similarly, the higher the threshold, the greater the relative weight calculated for any metric value that is less than the threshold. (When the value reaches the threshold, the weight goes to zero.)

Note that the default coefficient and default threshold values shown in Table 2.3 are *metric* defaults, not *template* defaults. The template defaults take precedence over the metric defaults, just as user-specified values in the custom **real_server** monitor take precedence over the template defaults. For example, in Figure 2.2, the template specifies a coefficient value of **1.5** for **ServerBandwidth** and no value for the other metrics. This means that the template will use the template default of **1.5** for the **ServerBandwidth** coefficient and the metric default of **1** for the coefficients of all other metrics. However, if a custom monitor **my_real_server** were configured specifying **2.0** as the **ServerBandwidth** coefficient, this user-specified value would override the template default.

The syntax for specifying non-default coefficient or threshold values is:

<metric>:<coefficient | <*>:<threshold>

The following examples show how to specify a coefficient value only, a threshold value only, and a coefficient and a threshold value, respectively.

```
b monitor my_real_server '{ use real_server metrics CPUPercentUsage:1.5 }'
b monitor my_real_server '{ use real_server metrics CPUPercentUsage:*:70 }'
b monitor my_real_server '{ use real_server metrics CPUPercentUsage:1.5:70 }'
```

Metric coefficient and threshold are the only non-template defaults. If a metric not in the template is to be added to the custom monitor, it must be added to the **metric** list:

```
b monitor my_real_server '{ use real_server metrics "HTTPClientCount" }'
```

To associate the monitor with the member node

Associate the custom health check monitor with the server node, creating an instance of the monitor for that node:

```
b node <node_addr> monitor use my_real_server
```

To set the load balancing method to Dynamic Ratio

Create or modify the load balancing pool to which the server belongs to use Dynamic Ratio load balancing:

```
b pool <pool_name> { lb_method dynamic_ratio <member definition>... }
```

Configuring Windows servers with WMI

For Windows, Link Controller provides a Data Gathering Agent **F5Isapi.dll** for the server. Configuring a Windows platform for Dynamic Ratio load balancing consists of four tasks:

- Installing the Data Gathering Agent **F5Isapi.dll** on the server
- Configuring a **wmi** health check monitor on the Link Controller
- Associating the health check monitor with the server to gather the metrics
- Creating or modifying the server pool to use Dynamic Ratio load balancing

To install the Data Gathering Agent (F5Isapi) on the server

1. Download the **Data Gathering Agent (F5Isapi.dll)** from the Link Controller. The plugin is located in **/usr/contrib/f5/isapi**. (The URL is **https://<bigip_address>/doc/isapi/f5isapi.dll**.)
2. Copy **f5isapi.dll** to the directory **C:\inetpub\scripts**.
3. Open the Internet Services Manager.
4. In the left pane of the Internet Services Manager, open the folder **<machine_name>\Default Web Site\Script**, where **<machine_name>** is the name of the server you are configuring. The contents of **Scripts** folder opens in the right pane.
5. In the right pane, right click **F5Isapi.dll** and select **Properties**. The Properties dialog box for **F5Isapi.dll** opens.

6. Deselect **Logvisits**. (Logging of each visit to the agent quickly fills up the log files.)
7. Click the File Security tab.
The File Security options appears.
8. In the **Anonymous access and authentication control group** box, click **Edit**.
The Authentication Methods dialog box opens.
9. In the **Authentication methods** dialog box, clear all check boxes, then select **Basic Authentication**.
10. In the **Authentication methods** dialog box, click **OK** to accept the changes.
11. In the **Properties** dialog box, click **Apply**.
12. The WMI Data Gathering Agent is now ready to be used.

To configure a wmi monitor for the server node

Using the Configuration utility or the **bigpipe** command, create a health check monitor using the **wmi** monitor template. The **wmi** monitor template is shown in Figure 2.3.

```
monitor type wmi {  
  interval 5  
  timeout 16  
  dest *:12346  
  username ""  
  password ""  
  method "POST"  
  urlpath "/scripts/F5Isapi.dll"  
  cmd "GetCPUInfo, GetDiskInfo, GetOSInfo"  
  metrics "LoadPercentage, DiskUsage, PhysicalMemoryUsage:1.5,  
  VirtualMemoryUsage:2.0"  
  post "<input type='hidden' name='RespFormat' value='HTML'>"  
  agent "Mozilla/4.0 (compatible: MSIE 5.0; Windows NT)"  
}
```

Figure 2.3 *wmi monitor template*

The monitor template contains default values for all the attributes. These are template defaults. In creating a custom monitor from the template, the only default values you are required to change are the null values for user name and password. For example:

```
b monitor my_wmi '{ use wmi username "dave" password "$getm" }'
```


You may also add commands and metrics and modify metric attribute values. The complete set of commands, associated metrics, and metric attribute default values is shown in Table 2.4.

Command	Metric	Default Coefficient	Default Threshold
GetCPUInfo	LoadPercentage (%)	1.0	80
GetOSInfo	PhysicalMemoryUsage (%)	1.0	80
	VirtualMemoryUsage (%)	1.0	80
	NumberRunningProcesses	1.0	100
GetDiskInfo	DiskUsage (%)	1.0	90
GetPerfCounters	TotalKBytesPerSec	1.0	10,000
	ConnectionAttemptsPerSec	1.0	500
	CurrentConnections	1.0	500
	GETRequestsPerSec	1.0	500
	PUTRequestsPerSec	1.0	500
	POSTRequestsPerSec	1.0	500
	AnonymousUsersPerSec	1.0	500
	CurrentAnonymousUsers	1.0	500
	NonAnonymousUsersPerSec	1.0	500
	CurrentNonAnonymousUser	1.0	500
	CGIRequestsPerSec	1.0	500
	CurrentCGIRequests	1.0	500
	ISAPIRequestsPerSec	1.0	500
	CurrentISAPIRequests	1.0	500
GetWinMediaInfo	AggregateReadRate	1.0	10,000 Kbps
	AggregateSendRate	1.0	10,000 Kbps
	ActiveLiveUnicastStreams	1.0	1000
	ActiveStreams	1.0	1000

Table 2.4 *wmi* monitor commands and metrics

Command	Metric	Default Coefficient	Default Threshold
	ActiveTCPStreams	1.0	1000
	ActiveUDPStreams	1.0	1000
	AllocatedBandwidth	1.0	10,000 Kbps
	AuthenticationRequests	1.0	1000
	AuthenticationsDenied	1.0	100
	AuthorizationRequests	1.0	1000
	AuthorizationsRefused	1.0	100
	ConnectedClients	1.0	500
	ConnectionRate	1.0	500
	HTTPStreams	1.0	1000
	HTTPStreamsReadingHeader	1.0	500
	HTTPStreamsStreamingBody	1.0	500
	LateReads	1.0	100
	PendingConnections	1.0	100
	PluginErrors	1.0	100
	PluginEvents	1.0	100
	SchedulingRate	1.0	100
	StreamErrors	1.0	100
	StreamTerminations	1.0	100
	UDPResendRequests	1.0	100
	UDPResendsSent	1.0	100

Table 2.4 *wmi* monitor commands and metrics

For more information about the metric coefficients and thresholds, refer to the description accompanying Table 2.3, *real_server* monitor metrics, on page 2-11. Note that for a **wmi** monitor, you can add commands. To do this, simply add them to the **cmd** list.

To associate the monitor with the member node

Associate the custom health check monitor with the server node, creating an instance of the monitor for that node:

```
b node <node_addr> monitor use my_wmi
```

To set the load balancing mode to Dynamic Ratio

Use the following syntax to create or modify the load balancing pool to which the server belongs to use Dynamic Ratio load balancing:

```
b pool <pool_name> { lb_method dynamic_ratio <member definition>...}
```

Configuring SNMP servers

The Link Controller includes an SNMP data collecting agent that can query remote SNMP agents of various types, including the UC Davis agent and the Windows 2000 Server agent. Configuring a server to use its SNMP agent for Dynamic Ratio load balancing consists of three tasks:

- Configuring a health check monitor, using either the Configuration utility or the **bigpipe** command
- Associating the health check monitor with the server to gather the metrics
- Creating or modifying the server pool to use Dynamic Ratio load balancing

Link Controller provides two templates that you can use to create a health monitor for a server that uses an SNMP agent. These two monitor templates are:

- **snmp_dca**
Use this template when you want to use default values or specify new values for CPU, memory, and disk metrics. When using this template, you can also specify values for other types of metrics that you wish to gather.
- **snmp_dca_base**
Use this template when you want to use default values or specify values for metrics other than CPU, memory, and disk usage. When using this template, values for CPU, memory, and disk metrics are omitted.

◆ Note

*For a description of these templates and the default values for each metric, see **Working with templates for EAV monitors**, on page 2-84.*

Figure 2.4 shows a monitor based on the **snmp_dca** monitor template. This monitor uses the default metric values. A user can optionally specify variables for user-defined metrics.

```
monitor my_snmp_dca
  '{ use snmp_dca
    interval 10
    timeout 30
    dest *:161
    agent_type "UCD"
    cpu_coefficient "1.5"
    cpu_threshold "80"
    mem_coefficient "1.0"
    mem_threshold "70"
    disk_coefficient "2.0"
    disk_threshold "90"
    USEROID ""
    USEROID_COEFFICIENT "1.0"
    USEROID_THRESHOLD "90"
  }'
```

Figure 2.4 A monitor based on the **snmp_dca** template

Figure 2.5 shows a monitor based on the **snmp_dca_base** monitor template. This monitor uses the default metric values.

```
monitor my_snmp_dca_base
  '{ use snmp_dca_base
    interval 10
    timeout 30
    dest *:161

    USEROID ""
    USEROID_COEFFICIENT "1.0"
    USEROID_THRESHOLD "90"
  }'
```

Figure 2.5 A monitor based on the **snmp_dca_base** template

◆ Note

*Note that in the above examples, the user-defined variables are specified as **USEROID**, **USEROID_COEFFICIENT**, and **USEROID_THRESHOLD**. You can create any variable names you want. Although the values shown in the above examples are entered in uppercase, uppercase is not required.*

To configure a monitor based on either the **snmp_dca** or **snmp_dca_base** template, you can use either the Configuration utility or the **bigpipe** command.

◆ Note

*The default agent type specified in the **snmp_dca** template is UC Davis. When configuring a monitor for a Windows 2000 server, you must change the agent type to Windows 2000.*

To configure an SNMP monitor using the Configuration utility

1. In the Navigation pane, click **Monitors**.
2. Click the **Add** button.
This displays the Configure Monitor Name and Parent screen.
3. Enter a unique name for the monitor in the **Name** box and select a template from the **Inherits from** box. If you want the monitor to include CPU, memory, disk, and user metrics, select the **snmp_dca** template. If you want the monitor to include user metrics only, select the **snmp_dca_base** template.
4. Click **Next**. This displays the Configure Basic Properties screen.
5. Retain or change the values in the **Interval** and **Timeout** boxes.
6. Click **Next**. This displays the Configure EAV SNMP DCA Monitor screen.
7. Retain or change the values for CPU, memory, and disk use. Also note that in the **snmp_dca** template, the default value for the **Agent Type** property is **UCD**. To configure a monitor for a Windows 2000 agent, change this value to **WIN2000**.
8. Click **Next**.
This displays the Configure EAV Variables screen.
9. If you are specifying user-defined metrics, configure the EAV variables by specifying a unique name and a value for each Name/Value pair.
The three variables (that is, Name/Value pairs) correspond to OID, coefficient, and threshold. Note that if the value of the OID variable is an absolute value, verify that the user-defined threshold value is also an absolute value. If the threshold value is not absolute, Link Controller might not factor the value into the load calculation. The default user-defined threshold value is **90**.
10. Click **Next**.
This displays the Configure Destination Address and Service (Alias) screen. We recommend that you use the default values shown here.
11. Click **Done**.

To configure an SNMP monitor using the **bigpipe** command

When configuring an SNMP monitor using the **bigpipe** command, you can use the default CPU, memory, and disk coefficient and threshold values specified in the templates, or you can change the default values. Optionally, you can specify coefficient and threshold values for gathering other types of data. Note that if the monitor you are configuring is for a type of SNMP agent other than UC Davis, you must specify the agent type as an argument to the **bigpipe** command.

The following command-line examples show various ways to configure an SNMP monitor. Note that although arguments for user-defined metrics are shown in uppercase, uppercase is not required.

To configure a monitor for a UC Davis SNMP agent, using default CPU, memory, and disk use values, use the **bigpipe monitor** command, as in the following example.

```
b monitor my_snmp_dca '{ use snmp_dca }'
```

To configure a monitor for a UC Davis SNMP agent, using all default CPU, memory threshold, and disk use values and specifying a non-default memory coefficient value, use the **bigpipe monitor** command, as in the following example.

```
b monitor my_snmp_dca '{ use snmp_dca mem_coefficient "1.5" }'
```

To configure a monitor for a UC Davis SNMP agent, using default CPU, memory threshold, and disk use values and specifying non-default memory coefficient and user values, use the **bigpipe monitor** command, as in the following example.

```
b monitor my_snmp_dca '{ use snmp_dca mem_coefficient\
"1.5"/USEROID ".1.3.6.1.4" USEROID_COEFFICIENT "1.5"\
USEROID_THRESHOLD/"80" }'
```

To configure a monitor for a UC Davis SNMP agent, omitting CPU, memory, and disk use values and using default user coefficient and user threshold values (1.0 and 90 respectively), use the **bigpipe monitor** command, as in the following example.

```
b monitor my_snmp_dca '{ use snmp_dca_base USEROID\
".1.3.6.1.4"}'
```

To configure a monitor for a UC Davis SNMP agent, omitting CPU, memory, and disk use values and specifying non-default user values, use the **bigpipe monitor** command, as in the following example.

```
b monitor my_snmp_dca_base '{ use snmp_dca_base\
USEROID/"1.3.6.1.4" USEROID_COEFFICIENT/ "1.5"\
USEROID_THRESHOLD "80" )'
```

To configure a monitor for a Windows 2000 SNMP agent, using default CPU, memory, and disk use values, use the **bigpipe monitor** command, as in the following example.

```
b monitor my_win2000_snmp_dca '{use snmp_dca agent_type\
"WIN2000"}'
```

To associate the health check monitor with the member node

Use the following syntax to associate the custom health check monitor with the server node and create an instance of the monitor for that node:

```
b node <node_addr> monitor use my_snmp_dca
```

To set the load balancing method to Dynamic Ratio

Use the following syntax to create or modify the load balancing pool to which the server belongs to use Dynamic Ratio load balancing:

```
b pool <pool_name> { lb_method dynamic_ratio\
<member definition>... }
```

Priority-based member activation

You can load balance traffic across all members of a pool or across only members that are currently activated according to their priority number. In priority-based member activation, each member in a pool is assigned a priority number that places it in a priority group designated by that number. With all nodes available (meaning they are enabled, marked **up**, and have not exceeded their connection limit), the Link Controller distributes connections to all nodes in the highest priority group only, that is, the group designated by the highest priority number. The **min_active_members** value determines the minimum number of members that must remain available for traffic to be confined to that group. If the number of available nodes in the highest priority group goes below the minimum number, the Link Controller also distributes traffic to the next higher priority group, and so on.

```
pool my_pool {
    lb_mode fastest
    min_active_members 2
    member 10.12.10.1:80 priority 3
    member 10.12.10.2:80 priority 3
    member 10.12.10.3:80 priority 3
    member 10.12.10.4:80 priority 2
    member 10.12.10.5:80 priority 2
    member 10.12.10.6:80 priority 2
    member 10.12.10.7:80 priority 1
    member 10.12.10.8:80 priority 1
    member 10.12.10.9:80 priority 1
}
```

Figure 2.6 Sample pool configuration for priority based member activation

The configuration shown in Figure 2.6 has three priority groups, **3**, **2**, and **1**. Connections are first distributed to all nodes with priority **3**. If fewer than two priority **3** nodes are available, traffic is directed to the priority **2** nodes as well. If both the priority **3** group and the priority **2** group have fewer than two nodes available, traffic is directed to the priority **1** group as well. The Link Controller continuously monitors the higher priority groups, and each time a higher priority group once again has the minimum number of available nodes, the Link Controller again limits traffic to that group.

◆ WARNING

If you set the load balancing mode to Ratio (as opposed to Ratio Member), you must define the ratio settings for each node address.

Persistence

If you are setting up an e-commerce or other type of transaction-oriented site, you may need to configure persistence on the Link Controller. Persistence is one of the pool attributes listed in Table 2.1, on page 2-2.

Whether you need to configure persistence or not simply depends on how you store client-specific information, such as items in a shopping cart, or airline ticket reservations. For example, you may store the airline ticket

reservation information in a back-end database that all nodes can access, or on the specific node to which the client originally connected, or in a cookie on the client's machine.

If you store client-specific information on specific nodes, you need to configure persistence. When you turn on persistence, returning clients can bypass load balancing and instead can go to the node where they last connected in order to get to their saved information.

The Link Controller tracks information about individual persistent connections, and keeps the information only for a given period of time. The way in which persistent connections are identified depends on the type of persistence.

Types of persistence

The types of persistence are:

- ◆ **Simple persistence**
Simple persistence supports TCP and UDP protocols, and tracks connections based only on the client IP address.
- ◆ **Destination address affinity (sticky persistence)**
Destination address affinity directs requests for a certain destination to the same proxy server, regardless of which client the request comes from.

◆ **Note**

All persistence methods are properties of pools.

Persistence options

When setting up persistence, you can enable either of the following two options:

- ◆ **Persistence across virtual servers with the same address**
Persistence across virtual servers with the same address causes the Link Controller to maintain persistence only when the virtual server hosting the connection has the same virtual address as the virtual server hosting the initial persistent connection.
- ◆ **Persistence across all virtual servers**
Persistence across all virtual servers causes the Link Controller to maintain persistence for all connections requested by the same client, regardless of which virtual server hosts each individual connection initiated by the client.

Simple persistence

Simple persistence tracks connections based only on the client IP address. When a client requests a connection to a virtual server that supports simple persistence, the Link Controller checks to see if that client previously connected, and if so, returns the client to the same node.

You may want to use SSL persistence and simple persistence together. In situations where an SSL session ID times out, or where a returning client does not provide a session ID, you may want the Link Controller to direct the client to the original node based on the client's IP address. As long as the client's simple persistence record has not timed out, the Link Controller can successfully return the client to the appropriate node.

Persistence settings for pools apply to all protocols. When the persistence timer is set to a value greater than 0, persistence is **on**. When the persistence timer is set to 0, persistence is **off**.

To configure simple persistence for pools using the Configuration utility

1. In the navigation pane, click **Pools**.
The Pools screen opens.
2. Select the pool for which you want to configure simple persistence.
The Pool Properties screen opens.
3. Click the Persistence tab.
The Persistence Properties screen opens.
4. In the Persistence Type section, click the **Simple** button.
Type the following information:
  - **Timeout (seconds)**
Set the number of seconds for persistence on the pool. (This option is not available if you are using rules.)
  - **Mask**
Set the persistence mask for the pool. The persistence mask determines persistence based on the portion of the client's IP address that is specified in the mask.
5. Click the **Apply** button.

To configure simple persistence for pools from the command line

You can use the **bigpipe pool** command with the **modify** keyword to set simple persistence for a pool. Note that a timeout greater than 0 turns persistence **on**, and a timeout of 0 turns persistence **off**.

```
b pool <pool_name> modify { \  
persist_mode simple \  
simple_timeout <timeout> \  
simple_mask <ip_mask> }
```

For example, to set simple persistence on the pool **my_pool**, type this command:

```
b pool my_pool modify { persist_mode simple simple_timeout 3600\  
simple_mask 255.255.255.0 }
```

Using a simple timeout and a persist mask on a pool

The persist mask feature works only on pools that implement simple persistence. By adding a persist mask, you identify a range of client IP addresses to manage together as a single simple persistent connection when connecting to the pool.

To apply a simple timeout and persist mask using the Configuration utility

1. In the navigation pane, click **Pools**.
The Pools screen opens.
2. In the Pools list, click the pool for which you want to set up simple persistence.
The properties screen opens.
3. Click the Persistence tab.
The Persistence screen opens.
4. Select **Simple** mode.
5. In the **Timeout** box, type the timeout in seconds.
6. In the **Mask** box, type the persist mask you want to apply.
7. Click the **Apply** button.

To apply a simple timeout and persist mask from the command line

The complete syntax for the command is:

```
b pool <pool_name> modify { \  
[<lb_mode_specification>] \  
persist_mode simple \  
simple_timeout <timeout> \  
simple_mask <dot_notation_longword> }
```

For example, the following command would keep persistence information together for all clients within a C class network that connect to the pool **my_pool**:

```
b pool my_pool modify { \  
persist_mode simple \  
simple_timeout 1200 \  
simple_mask 255.255.255.0 }
```

You can turn off a persist mask for a pool by using the **none** option in place of the **simple_mask** mask. To turn off the persist mask that you set in the preceding example, use the following command:

```
b pool my_pool modify { simple_mask none }
```

To display all persistence information for the pool named **my_pool**, use the **show** option:

```
b pool my_pool persist show
```

Destination address affinity (sticky persistence)

You can optimize your proxy server array with destination address affinity (also called sticky persistence). **Destination address affinity** directs requests for a certain destination IP address to the same proxy server, regardless of which client the request comes from.

This enhancement provides the most benefits when load balancing caching proxy servers. A caching proxy server intercepts web requests and returns a cached web page if it is available. In order to improve the efficiency of the cache on these proxies, it is necessary to send similar requests to the same proxy server repeatedly. Destination address affinity can be used to cache a given web page on one proxy server instead of on every proxy server in an array. This saves the other proxies from having to duplicate the web page in their cache, wasting memory.

To activate destination address affinity using the Configuration utility

You can activate destination address affinity only on pools that are directly or indirectly referenced by wildcard virtual servers. For information on setting up a wildcard virtual server, see the *Wildcard virtual servers*, on page 2-41.

1. In the navigation pane, click **Pools**.
The Pools screen opens.
2. In the Pools list, click the pool for which you want to set up destination address affinity.
The properties screen for the pool you clicked opens.
3. Click the Persistence tab.
The Persistence screen opens.
4. Click the **Destination Address Affinity** button to enable destination address affinity.
5. In the **Mask** box, type in the mask you want to apply to sticky persistence entries.

To activate destination address affinity from the command line

Use the following command to activate sticky persistence for a pool:

```
b pool <pool_name> modify { persist_mode sticky sticky_mask <ip address> }
```

Use the following command to delete sticky entries for the specified pool:

```
b pool <pool_name> sticky clear
```

To show the persistence configuration for the pool:

```
b pool <pool_name> persist show
```

Maintaining persistence across virtual servers that use the same virtual addresses

When this mode is turned on, the Link Controller attempts to send all persistent connection requests received from the same client, within the persistence time limit, to the same node only when the virtual server hosting the connection has the same virtual address as the virtual server hosting the initial persistent connection. Connection requests from the client that go to other virtual servers with different virtual addresses, or those connection requests that do not use persistence, are load balanced according to the load balancing mode defined for the pool.

If a Link Controller configuration includes the following virtual server mappings, where the virtual server **v1:http** references the **http_pool** (contains the nodes **n1:http** and **n2:http**) and the virtual server **v1:ssl** references the pool **ssl_pool** (contains the nodes **n1:ssl** and **n2:ssl**). Each virtual server uses persistence:

```
b virtual v1:http use pool http_pool
b virtual v1:ssl use pool ssl_pool
b virtual v2:ssl use pool ssl_pool
```

However, if the client subsequently connects to **v1:ssl**, the Link Controller uses the persistence session established with the first connection to determine the node that should receive the connection request, rather than the load balancing mode. The Link Controller should send the third connection request to **n1:ssl**, which uses the same node address as the **n1:http** node that currently hosts the client's first connection with which it shares a persistent session.

For example, a client makes an initial connection to **v1:http** and the load balancing mechanism assigned to the pool **http_pool** chooses **n1:http** as the node. If the same client then connects to **v2:ssl**, the Link Controller starts tracking a new persistence session, and it uses the load balancing mode to determine which node should receive the connection request because the requested virtual server uses a different virtual address (**v2**) than the virtual server hosting the first persistent connection request (**v1**). In order for this mode to be effective, virtual servers that use the same virtual address, as well as those that use TCP or SSL persistence, should include the same node addresses in the virtual server mappings.

To activate persistence for virtual servers that use the same address using the Configuration utility

1. In the navigation pane, click **System**.
The Network Map screen opens.
2. Click the Advanced Properties tab.
The Link Controller System Control Variables screen opens.
3. Click the **Allow Persistence Across All Ports for Each Virtual Address** check box. (To disable this persistence mode, clear the check box).
4. Click the **Apply** button.

To activate persistence for virtual servers that use the same address from the command line

The global variable **persist_across_services** turns this mode on and off. To activate the persistence mode, type:

```
b global persist_across_services enable
```

To deactivate the persistence mode, type:

```
b global persist_across_services disable
```

Maintaining persistence across all virtual servers

You can set the Link Controller to maintain persistence for all connections requested by the same client, regardless of which virtual server hosts each individual connection initiated by the client. When this mode is turned on, the Link Controller attempts to send all persistent connection requests received from the same client, within the persistence time limit, to the same node. Connection requests from the client that do not use persistence are load balanced according to the currently selected load balancing mode.

The following examples show virtual server mappings, where the virtual servers **v1:http** and **v2:http** reference the **http1_pool** and **http2_pool** (both pools contain the nodes **n1:http** and **n2:http**) and the virtual servers **v1:ssl** and **v2:ssl** reference the pools **ssl1_pool** and **ssl2_pool** (both pools contain the nodes **n1:ssl** and **n2:ssl**). Each virtual server uses persistence:

```
b virtual v1:http use pool http1_pool
```

```
b virtual v1:ssl use pool ssl1_pool
```

```
b virtual v2:http use pool http2_pool
```

```
b virtual v2:ssl use pool ssl2_pool
```

Say that a client makes an initial connection to **v1:http** and the Link Controller load balancing mechanism chooses **n1:http** as the node. If the same client subsequently connects to **v1:ssl**, the Link Controller would send the client's request to **n1:ssl**, which uses the same node address as the **n1:http** node that currently hosts the client's initial connection. What makes this mode different from maintaining persistence across virtual servers that use the same virtual address is that if the same client subsequently connects to **v2:ssl**, the Link Controller would send the client's request to **n1:ssl**, which uses the same node address as the **n1:http** node that currently hosts the client's initial connection.

WARNING

In order for this mode to be effective, virtual servers that use pools with TCP or SSL persistence should include the same member addresses in the virtual server mappings.

To activate persistence across all virtual servers using the Configuration utility

1. In the navigation pane, click the **System** icon.
The Network Map screen opens.

2. Click the Advanced Properties tab.
The Link Controller System Control Variables screen opens.
3. Click the **Allow Persistence Across All Virtual Servers** check box to activate this persistence mode.
4. Click the **Apply** button.

To activate persistence across all virtual servers from the command line

The global variable **persist_across_virtuals** turns this mode on and off. To activate the persistence mode, type:

```
b global persist_across_virtuals enable
```

To deactivate the persistence mode, type:

```
b global persist_across_virtuals disable
```

HTTP redirection

Another attribute of a pool is HTTP redirection. ***HTTP redirection*** allows you to configure a pool so that HTTP traffic is redirected to another protocol identifier, host name, port number, or URI path. For example, if all members of a pool are unavailable (that is, the members are disabled, marked **down**, and have exceeded their connection limit), the HTTP request can be redirected to the fallback host, with the HTTP reply Status Code **302 Found**.

When configuring a pool to redirect HTTP traffic to a fallback host, you can use an IP address or a fully-qualified domain name (FQDN), or you can use a special format string included in the Link Controller. These format strings can also be used for specifying protocol identifiers, ports, and URIs.

The following two sections describe these two ways of redirecting HTTP requests. Following these two sections is a description of a related feature, which allows you to configure a server to rewrite the specified HTTP redirection.

Using IP addresses and fully qualified domain names

When redirecting traffic to a fallback host, you can specify the fallback host as an IP address or as a fully qualified domain name (FQDN). In either case, it may include a port number. The example in Figure 2.7 redirects the request to **http://redirector.sam.com**.

```
pool my_pool {
    member 10.12.10.1:80
    member 10.12.10.2:80
    member 10.12.10.3:80
    fallback redirector.sam.com
}
```

Figure 2.7 Fallback host in a pool

◆ Note

The HTTP redirect mechanism is not a load balancing method. The redirect URL may be a virtual server pointing to the requested HTTP content, but this is not implicit in its use.

Table 2.5 shows how different fallback host specifications are resolved.

Requested URL	Fallback Host Specification	Redirect URL
http://www.sam.com/	fallback.sam.com	http://fallback.sam.com/
http://www.sam.com/	fallback.sam.com:8002	http://fallback.sam.com:8002/
http://www.sam.com:8001	fallback.sam.com	http://fallback.sam.com/
http://www.sam.com:8001/	fallback.sam.com:8002	http://fallback.sam.com:8002/
http://www.sam.com/sales	fallback.sam.com	http://fallback.sam.com/sales
http://192.168.101.3/	fallback.sam.com	http://fallback.sam.com/
http://192.168.101.3/sales	fallback.sam.com	http://fallback.sam.com/sales
http://www.sam.com/sale	192.168.101.5	http://192.168.101.5/sales
http://192.168.101.3/sales/default.asp?q=6	fallback.sam.com	http://fallback.sam.com/sales/default.asp?q=6

Table 2.5 How the fallback host specifications are resolved

Using format strings (expansion characters)

To allow HTTP redirection to be fully configurable with respect to target URI, the following format strings are available. These strings can be used within pools.

Table 2.6 lists and defines the format strings that you can use to specify HTTP redirection.

Format String	Description
%h	host name, as obtained from the Host: header of the client
%p	Port, from the virtual server listening port
%u	URI path, as obtained from a GET/POST request

Table 2.6 *Format strings for HTTP redirection*

An example of a fallback host string is **https://%h/sample.html**. In this string, specifying **https** as the protocol identifier causes the traffic to be redirected to that protocol instead of the standard **http** protocol. Also, the string **sample.html** causes the traffic to be redirected to that URI instead of to the standard URI specified in the HTTP header, which would normally be represented in the fallback string as **%u**.

Table 2.7 shows some sample redirection specifications, their explanations, and their resulting redirection.

Redirection string	Explanation	Resulting redirection
%h:%p/%u	No redirection (preserve host name, port, and path)	http://www.example.com:8080/sample
%h/unavailable	change path, remove port	http://www.example.com/unavailable
https://%h/unavailable	Specify https as protocol, remove port, change path	https://www.example.com/unavailable
www.sample.com:8080/%u	Change host name and port, preserve path	http://www.sample.com:8080/sample
https://1.2.3.4:443/%u/unavailable.html	Specify https as protocol, change host name, port, and path	https://1.2.3.4:443/sample/unavailable.html
ftp://1.2.3.4:%p/unavailable/%u	Specify ftp as protocol, change host name and path	ftp://1.2.3.4:8080/unavailable/sample
rtsp://%h:554/streamingmedia/%u	Specify rtsp as protocol, change port and path	rtsp://www.example.com:554/streamingmedia/sample

Table 2.7 Sample HTTP redirections using format strings

The example in Figure 2.8 shows a pool configured to redirect an HTTP request to a different protocol (**https**) host name (**1.2.3.4**), port number (**443**), and path (**unavailable.html**).

```
pool my_pool {
member 10.12.10.1:80
member 10.12.10.2:80
member 10.12.10.3:80

https://1.2.3.4:443/%u/unavailable.html
}
```

Figure 2.8 HTTP redirection specified in a pool

Rewriting HTTP redirection

Sometimes, a client request is redirected from the HTTPS protocol to the HTTP protocol, which is a non-secure channel. If you want to ensure that the request remains on a secure channel, you can cause that redirection to be rewritten so that it is redirected back to the HTTPS protocol. Also, through the rewriting of redirections, you can rewrite a port number or a URI path.

You can rewrite HTTP redirections for Microsoft Internet Information Server (IIS) servers. If your web server is an IIS server, you can configure that server to rewrite any HTTP redirections. Part of this IIS server configuration includes the installation of a special Link Controller filter, **redirectfilter.dll**, on the IIS server. The following section provides this IIS configuration procedure.

To install the filter for rewriting HTTP redirection

To install the ISAPI filter (**redirectfilter.dll**) for use with IIS, version 4.0 or 5.0, follow these steps:

1. Copy the filter DLL to an appropriate folder, such as the **SCRIPTS** or **CGI-BIN** subdirectory.
2. Open the Internet Service Manager (MMC).
3. Select the appropriate level for the ISAPI filter:
  - If you intend to use the ISAPI filter with all Web sites, select the **ServerName** icon.
  - If you intend to use the ISAPI filter with a specific Web site, select the icon for that Web site (for example, the default Web site).

4. Right-click the level (icon) that you selected.
5. Click the **ISAPI Filters** tab.

***Note:** To configure an ISAPI filter for all Web sites, first click the **Edit** button that is next to the Master Properties of the WWW Service.*

6. Click **Add**.
7. Type a name for the ISAPI filter.
8. Click **Browse** and select the ISAPI filter that you copied in step 1.
9. Click **OK**.
10. Stop the IISADMIN service. To do this, either type **net stop iisadmin /y** at a command prompt, or use the Services applet that is located in Control Panel (in Windows NT® 4.0) or Administrative Tools (in Windows 2000).
11. Start the World Wide Web Publishing Service by typing **net start w3svc** at a command prompt, or by using the Services applet that is located in Control Panel (in Windows NT 4.0) or Administrative Tools (in Windows 2000).

12. Repeat the previous step for any other services that were stopped in step 11.
13. Browse back to the **ISAPI Filters** tab (by following steps 1-5) and verify that the filter is loaded properly. You should see a green arrow that is pointing up under the Status column.

◆ Note

*The **ISAPI Filters** tab specifies a load order, with the filter at the top of the list loading first. Normally **Sspifilt.dll**, the ISAPI filter for SSL, is at the top of the list to allow any other filters to access data before IIS encrypts and transmits or decrypts and receives TTPS traffic.*

HTTP header insertion

An optional attribute of a pool is HTTP header insertion. Using this attribute, you can configure a pool to insert a header into an HTTP request. The HTTP header being inserted can include a client IP address. Including a client IP address in an HTTP header is useful when a connection goes through a secure network address translation (SNAT) and you need to preserve the original client IP address.

The header insertion must be specified in the pool definition as a quoted string. Figure 2.9 shows the required syntax.

```
pool <pool_name> {
  header insert <quoted string>
}
```

Figure 2.9 Syntax of a header insertion string within a pool

Optionally, you can include rule variables in the quoted string. For example, the pool definition shown in 2.10 uses the rule variable **client_addr** to represent the original client IP address of an HTTP request.

```
pool my_pool {
  header insert "OrigClientAddr:${client_addr}"
  member 10.0.0.1:80
  member 10.0.0.2:80
  member 10.0.0.3:80
}
```

Figure 2.10 Example of a rule variable within a pool for header insertion

The rule variables that can be used for header insertion are:

- **client_addr**
- **client_port**

- **server_addr**
- **server_port**
- **link_qos**
- **ip_qos**

Figure 2.11 shows a pool that inserts a header, using all of the above rule variables.

```
pool my_pool {  
    header insert "ClientSide:${client_addr}:${client_port} ->  
    ${server_addr}:${server_port} tos=${ip_tos} qos=${link_qos}"  
    member 10.0.0.1:80  
    member 10.0.0.2:80  
    member 10.0.0.3:80  
}
```

Figure 2.11 Pool with header insertion string using multiple rule variables

The above header insertion string inserts a header such as that shown in Figure 2.12 into an HTTP request:

```
GET /index.html HTTP/1.0  
ClientSide: 10.0.0.1:3340 -> 10.0.0.101:80 tos=16 qos=0  
Host: www.yahoo.com  
Connection: Keep-Alive
```

Figure 2.12 Header resulting from a header insertion string within a pool

---

◆ **Note**

If the rule variable specified is not a valid variable, the invalid variable name is inserted directly into the HTTP request, with no substitution.

In addition to inserting a client IP address into an HTTP request, you can configure an SSL Accelerator proxy to insert other types of headers into HTTP requests. Examples of headers that an SSL proxy can insert are: information on client certificates, cipher specifications, and client session IDs.

Quality of Service (QoS) level

Another attribute of a pool is the Quality of Service (QoS) level. The **QoS** level is a means by which network equipment can identify and treat traffic differently based on an identifier. As traffic enters the site, the Link Controller can set the QoS level on a packet, based on the QoS level defined in the pool to which the packet is sent. The Link Controller can also apply a rule that sends the traffic to different pools of servers based on the Quality of Service level.

The Link Controller can tag outbound traffic (the return packets based on an **HTTP GET**) based on the QoS value set in the pool. That value is then inspected by upstream devices and given appropriate priority. Based on a rule, the Link Controller can examine incoming traffic to see if it has a particular QoS or ToS tag in the header. The Link Controller can then make a rule-based load balancing decision based on that tag.

Figure 2.13 shows how to configure a pool so that a QoS level is set for a packet sent to that pool. In this example, the QoS tag, represented by the *link_qos* variable, is set to **3** when sending packets to the client, and set to **4** when packets are sent to the server.

```
pool http_pool {  
    link_qos to client 3  
    link_qos to server 4  
}
```

Figure 2.13 Example of a pool that sets the QoS level on a packet

In addition to configuring a pool to set the QoS level on a packet, you can configure a rule that selects a pool based on the existing QoS value within the packet. For more information, see *Quality of Service (QoS) level*, on page 2-34.

Type of Service (ToS) level

Another attribute of a pool is the Type of Service (ToS) level. The *ToS* level, also known as the *DiffServ* value, is a means by which network equipment can identify and treat traffic differently based on an identifier. As traffic enters the site, the Link Controller can set the ToS level on a packet, based on the ToS level defined in the pool to which the packet is sent. The Link Controller can also apply a rule and send the traffic to different pools of servers based on the ToS level.

The Link Controller can tag outbound traffic (the return packets based on an **HTTP GET**) based on the ToS value set in the pool. That value is then inspected by upstream devices and given appropriate priority. Based on a rule, the Link Controller can examine incoming traffic to see if it has a particular ToS tag in the header. The Link Controller can then make a rule-based load balancing decision based on that tag.

Figure 2.14 shows how to configure a pool so that a ToS level is set for a packet sent to that pool. In this example, the ToS tag, represented by the *ip_tos* variable, is set to **16** when sending packets to the client, and set to **16** when packets are sent to the server.

```
pool http_pool {  
    ip_tos to client 16  
    ip_tos to server 16  
}
```

Figure 2.14 Example of a pool that sets the ToS level on a packet

In addition to configuring a pool that sets the ToS level on a packet, you can configure a rule that selects a pool based on the existing ToS value within the packet. For more information, see *Type of Service (ToS) level*, on page 2-35.

Disabling SNAT and NAT connections

When configuring a pool, you can specifically disable any secure network address translations (SNATs) or network address translations (NATs) for any connections that use that pool. By default, this setting is enabled.

For general information on SNATs and NATs, see *Address translation: SNATs, NATs, and IP forwarding*, on page 2-65.

The example in Figure 2.15 shows the syntax for disabling SNAT and NAT translation for any connections that use the pool **my_pool**.

```
pool my_pool {  
    snat disable  
    nat disable  
    member 10.1.1.1:80  
    member 10.1.1.2:80  
}
```

Figure 2.15 Disabling SNAT and NAT translations

To disable a SNAT or NAT connection for a pool using the Configuration utility

1. In the navigation pane, click **Pools**.
2. Click the **Add** button.
3. Clear the **Enable SNATs** check box. (By default, this box is checked.)
4. Click **Done**.

To disable a SNAT or NAT connection for a pool from the command line

Use the following command to disable a SNAT or NAT connection:

```
b pool <pool_name> modify { snat disable }
```

One case in which you might want to configure a pool to disable SNAT or NAT connections is when you want the pool to disable SNAT or NAT connections for a specific service. In this case, you could create a separate pool to handle all connections for that service, and then configure the **snat disable** or **nat disable** attribute on that pool. The following section describes this procedure.

To disable SNAT connections that use a specific service

The following procedure creates an automapped SNAT for a VLAN, creates a pool that disables SNAT or NAT connections, and then directs a wildcard virtual server using port **162** to send connections to the newly-defined pool.

1. Enable SNAT automapping on the self IP address for VLAN **my_vlan**. For example:

```
b self 192.168.33.14 vlan my_vlan snat automap enable
```
2. Create an automapped SNAT for the VLAN **my_vlan**. For example:

```
b snat map my_vlan to auto
```
3. Create a forwarding pool with the **snat disable** attribute defined. For example:

```
b pool snat_disable_pool { snat disable forward }
```

Note: For information on forwarding pools, see **Forwarding pools**, on page 2-38.
4. Create a wildcard virtual server for the VLAN **my_vlan**, specifying port **162** and the pool **snat_disable_pool**. For example:

```
b virtual my_vlan :162 use pool snat_disable_pool
```

Figure 2.16 shows the resulting entries in the **/config/bigip.conf** file.

```
# self IP addresses
self 192.168.33.14 {
  vlan my_vlan
  netmask 255.255.255.0
  broadcast 192.168.33.255
  snat automap enable
}

# server pools
pool snat_disable_pool {
  snat disable
  forward
}

# virtual servers
virtual servers:162 unit 1 {
  use pool snat_disable_pool
  translate addr disable
}
```

Figure 2.16 Sample entries in the `/config/bigip.conf` file

Figure 2.17 shows an example of a rule that sends SNAT connections to a pool that disables SNAT connections on a range of ports, defined in the class **IP_Port_Range**.

```
# The snat_disable pool disables all SNAT connections.
if (client_port == one of IP_Port_Range {
  use ( snat_disable)
}
else {
  use ( other_pool)
}

# The IP_Port_Range class contains a list of two
ports/services.

class IP_Port_Range {
  161
  162
}
```

Figure 2.17 A rule that disables SNAT connections for a range of ports

Forwarding pools

A *forwarding pool* is a pool that specifies that a connection should be forwarded, using IP routing, instead of load balanced. In many cases, this eliminates the need to create a forwarding virtual server.

Forwarding pools are typically used with wildcard virtual servers or network virtual servers only. When you enable forwarding on a pool, you can apply any feature that can be configured on a pool to a forwarding connection.

A pool configured for forwarding has no members. Also, this type of pool cannot be the default gateway pool.

Figure 2.18 shows an example of a pool configured for forwarding.

```
pool my_pool {  
    link_qos to client 5  
    link_qos to server 5  
    forward  
}
```

Figure 2.18 Example of a pool configured for forwarding

To configure a pool for forwarding using the Configuration utility

1. In the navigation pane, click **Pools**.
2. Click the **Add** button.
3. Click the **Forwarding** button. If you enable forwarding, you cannot enter a list of pool members.
4. Click **Done**.

To configure a pool for forwarding from the command line

Use the following command to configure a forwarding pool:

```
b pool <pool_name> { forward }
```

◆ Note

*If you want to enable IP forwarding for a virtual server or globally for the Link Controller, see **Forwarding virtual servers**, on page 2-47 and **IP forwarding**, on page 2-77, respectively.*

Virtual servers

A **virtual server** with its virtual address is the visible, routable entity through which nodes in a load balancing pool are made available to a client, either directly or indirectly through a rule. (The exception is the forwarding virtual server, which simply forwards traffic and has no associated pools.)

You must configure a pool of servers before you can create a virtual server that references the pool. Before you configure virtual servers, you need to know:

- Which virtual server type meets your needs
- Whether you need to activate optional virtual server properties

Once you know which virtual server options are useful in your network, you can define any one of the four types of virtual servers.

You can configure various types of virtual servers, depending on your needs. Table 2.8 shows the types of virtual servers that you can create.

Virtual Server Type	Description
Standard virtual server	A standard virtual server is a virtual server with a full IP address. For example: 192.168.200.30:http
Wildcard virtual server	There are two types of wildcard servers: A port-specific wildcard virtual server is a virtual server with a port specified. A port-specific wildcard virtual server is used to accept all traffic for a specific service. A default wildcard virtual server is a wildcard virtual server with the service 0 , * , or any . A default wildcard server acts like a default router, accepting all traffic that does not match a standard, network, or port-specific wildcard server.
Network virtual server	A network virtual server is a virtual server with a network IP address, allowing it to handle a whole range of addresses in a network. For example: 192.168.200.0:http
Forwarding virtual server	A forwarding virtual server is a virtual server without a pool that simply forwards traffic to the destination node.

Table 2.8 Virtual server types

Standard virtual servers

A **standard virtual server** represents a specific site, such as an Internet web site or an FTP site, and it load balances content servers that are members of a pool. The IP address that you use for a standard virtual server should match the IP address that DNS associates with the site's domain name.

To define a standard virtual server using the Configuration utility

1. In the navigation pane, click **Virtual Servers**.

2. Click the **Add** button.
The Add Virtual Server screen opens.
3. In the **Address** box, type the virtual server's IP address or host name.
4. In the **Port** box, either type a port number or select a service name from the list.
5. In the Select Physical Resources screen, click the **Pool** button.
If you want to assign a load balancing rule to the virtual server, click **Rule** and select a rule you have configured.
6. In the **Pool** list, select the pool you want to apply to the virtual server.
7. Click the **Apply** button.

To define a standard virtual server from the command line

Type the **bigpipe virtual** command as shown below. Also, remember that you can use host names in place of IP addresses, and that you can use standard service names in place of port numbers.

```
b virtual <virt_ip>:<service> use pool <pool_name>
```

For example, the following command defines a virtual server that maps to the pool **my_pool**:

```
b virtual 192.200.100.25:80 use pool my_pool
```

◆ Note

*If a virtual server is to have the same IP address as a node in an associated VLAN, you must perform some additional configuration tasks. These tasks consist of: creating a VLAN group that includes the VLAN in which the node resides, assigning self-IP addresses to the VLAN group, and disabling the virtual server on the relevant VLAN. For information on creating VLAN groups and assigning self IP addresses to them, see **Creating VLAN groups**, on page 1-13. For information on disabling a virtual server for a specific VLAN, see **Enabling or disabling a virtual server**, on page 2-49.*

Wildcard virtual servers

Wildcard virtual servers are a special type of virtual server designed to manage network traffic for transparent network devices, such as transparent firewalls, routers, proxy servers, or cache servers. A wildcard virtual server manages network traffic that has a destination IP address unknown to the Link Controller. A standard virtual server typically represents a specific site, such as an Internet web site, and its IP address matches the IP address that DNS associates with the site's domain name. When the Link Controller receives a connection request for that site, the Link Controller recognizes that the client's destination IP address matches the IP address of the virtual server, and it subsequently forwards the client to one of the content servers that the virtual server load balances.

However, when you are load balancing transparent nodes, a client's destination IP address is going to seem random. The client is connecting to an IP address on the other side of the firewall, router, or proxy server. In this situation, the Link Controller cannot match the client's destination IP address to a virtual server IP address. Wildcard virtual servers resolve this problem by not translating the incoming IP address at the virtual server level on the Link Controller. For example, when the Link Controller does not find a specific virtual server match for a client's destination IP address, it matches the client's destination IP address to a wildcard virtual server. The Link Controller then forwards the client's packet to one of the firewalls or routers that the wildcard virtual server load balances, which in turn forwards the client's packet to the actual destination IP address.

Default vs. port-specific wildcard servers

When you configure wildcard virtual servers and the nodes that they load balance, you can use a wildcard port (port **0**) in place of a real port number or service name. A wildcard port handles any and all types of network services.

A wildcard virtual server that uses port **0** is referred to as a **default wildcard virtual server**, and it handles traffic for all services. A **port-specific wildcard virtual server** handles traffic only for a particular service, and you define it using a service name or a port number. If you use both a default wildcard virtual server and port-specific wildcard virtual servers, any traffic that does not match either a standard virtual server or one of the port-specific wildcard virtual servers is handled by the default wildcard virtual server.

By default, a default wildcard virtual server is enabled for all VLANs. However, you can specifically disable any VLANs that you do not want the default wildcard virtual server to support. Disabling VLANs for the default wildcard virtual server is done by creating a VLAN disabled list. Note that a VLAN disabled list applies to default wildcard virtual servers only. You cannot create a VLAN disabled list for a wildcard virtual server that is associated with one VLAN only.

You can use port-specific wildcard virtual servers for tracking statistics for a particular type of network traffic, or for routing outgoing traffic, such as HTTP traffic, directly to a cache server rather than a firewall or router.

We recommend that when you define transparent nodes that need to handle more than one type of service, such as a firewall or a router, you specify an actual port for the node and turn off port translation for the virtual server.

For the procedure to create a default wildcard server, see *Creating a default wildcard virtual server*, on page 2-44.

Configuring wildcard virtual servers

Creating a wildcard virtual server involves three tasks. First, you must create a pool that contains the addresses of the transparent devices. Next, you must create the wildcard virtual server. Then you must turn port translation off for the virtual server. The following sections describe these tasks, followed by the procedure for creating a default wildcard server.

Creating a pool of transparent devices

Before you create the wildcard virtual server, you create the pool of transparent devices that are the resources for the wildcard virtual server.

To create a pool of transparent devices using the Configuration utility

To create a pool of transparent devices using the Configuration utility, use the Add Pool wizard, available from the Pools screen. For more information, see *To create a pool using the Configuration utility*, on page 2-3.

To create a pool of transparent devices from the command line

Create the pool of transparent devices, using the **bigpipe pool** command. For example, you can create a pool of transparent devices called **transparent_pool** that uses the Round Robin load balancing mode:

```
b pool transparent_pool { member 10.10.10.101:80 member \
10.10.10.102:80 member 10.10.10.103:80 }
```

Creating a wildcard virtual server

After you create the pool of transparent devices, you create the wildcard virtual server.

To create a wildcard virtual server using the Configuration utility

1. In the navigation pane, click **Virtual Servers**.
2. Click the **Add** button.
The Add Virtual Server screen opens.
3. In the **Address** box, type the wildcard IP address **0.0.0.0**.
4. In the **Port** box, type a port number, or select a service name from the list. Note that port **0** defines a wildcard virtual server that handles all types of services. If you specify a port number, you create a port-specific wildcard virtual server. The wildcard virtual server handles traffic only for the port specified. For more information, see *Default vs. port-specific wildcard servers*, on page 2-42.
5. In Resources, click the **Pool** button.
6. In the Pool list, select the pool you want to apply to the virtual server.
7. Click the **Apply** button.

To create a wildcard virtual server from the command line

To create a wildcard virtual server that maps to the pool **transparent_pool**, use the following command:

```
b virtual 0.0.0.0:0 use pool transparent_pool
```

Because the pool members are firewalls and need to handle a variety of services, the virtual server is defined using port **0** (or ***** or **any**). You can specify any valid non-zero port for the node port and then turn off port translation for that port. In this example, service checks ping port **80**.

Disabling port translation for a wildcard virtual server

To turn off port translation for a wildcard virtual server using the Configuration utility

After you define the wildcard virtual server with a wildcard port, you must disable port translation for the virtual server.

1. In the navigation pane, click **Virtual Servers**.
The Virtual Servers screen opens.
2. In the virtual server list, click the virtual server for which you want to turn off port translation.
The Virtual Server Properties screen opens.
3. In the Enable Translation section, clear the **Port** box.
4. Click the **Apply** button.

To turn off port translation for a wildcard virtual server from the command line

To turn off port translation for the port in the virtual server definition, use the following command:

```
b virtual 0.0.0.0:0 translate port disable
```

Creating a default wildcard virtual server

You can create a default wildcard virtual server to handle any traffic that does not match either a standard virtual server or one of the port-specific wildcard virtual servers.

To create a default wildcard virtual server using the Configuration utility

1. In the Navigation pane, select **Virtual Servers**.
The Virtual Servers screen displays.
2. Click the **Add** button.
3. In the **Address** field, type the IP address **0.0.0.0**.
4. Click **Next**.
5. From the **VLAN** box, select **all**.
6. Click **Done**.

To create a default wildcard virtual server from the command line

To create a default wildcard virtual server from the command line, use the **bigpipe virtual** command with the following syntax:

```
b virtual *:* use pool <pool_name>
```

Creating multiple wildcard servers

Link Controller supports multiple wildcard virtual servers, all running simultaneously. Each wildcard virtual server must be assigned to an individual VLAN, and therefore handles packets for that VLAN only.

To create multiple wildcard virtual servers, you can use either the Configuration utility or the **bigpipe virtual** command.

To create multiple wildcard virtual servers using the Configuration utility

To create multiple wildcard virtual servers using the Configuration utility, use the following procedure:

1. In the Navigation pane, select **Virtual Servers**.
The Virtual Servers screen displays.
2. Click the **Add** button.
3. In the **Address** field, type the IP address **0.0.0.0**.
4. In the **Service** field, type the name of a service or select a service from the list box.
5. Click **Next**.
6. From the VLAN box, choose a VLAN name. Selecting **all** creates a default wildcard virtual server.
7. Click **Next**.
8. Continue configuring all properties for the wildcard virtual server.
Note that on the Configure Basic Properties screen, if you are creating a default wildcard virtual server, you can disable any VLANs associated with that wildcard virtual server.
9. Click **Done**.

Repeat for each wildcard virtual server that you want to create.

To create multiple wildcard virtual servers from the command line

To create a separate wildcard virtual server per VLAN from the command line, use the following command-line syntax:

```
b virtual <vlan_name> use pool <pool_name>
```

For example, the following commands define two wildcard virtual servers, the first for VLAN internal, and the second for VLAN external:

```
b virtual internal use pool my_pool
```

```
b virtual external use pool my_pool
```

Network virtual servers

You can configure a *network virtual server* to handle a whole network range, instead of just one IP address, or all IP addresses (a wildcard virtual server). For example, the virtual server in Figure 2.19 handles all traffic addresses in the 192.168.1.0 network.

```
bigpipe virtual 192.168.1.0:0 {  
    netmask 255.255.255.0 use pool ingress_firewalls  
}
```

Figure 2.19 A sample network virtual server

A network virtual server is a virtual server that has no bits set in the host portion of the IP address. The example above directs all traffic destined to the subnet **192.168.1.0/24** through the Link Controller to the **ingress_firewalls** pool.

The netmask of a network virtual server establishes which portion of the address is actually the network of a network virtual server. By default, this is the netmask of the self IP address. In the example, the network mask of **255.255.255.0** states that the network portion of the address is **192.168.1**, which in this case is obvious because only the last octet has a zero value.

A less obvious case would be the network virtual server **10.0.0.0:0**. Here, the zero in the second octet is ambiguous: it could be a wildcard or it could be a literal zero. If it is a wildcard, this would be established by a netmask of **255.0.0.0**. If it is a literal zero, this would be established by a netmask of **255.255.0.0**.

Another way you can use this feature is to create a catch-all web server for an entire subnet. For example, you could create the following network virtual server, shown in Figure 2.20.

```
bigpipe virtual 192.168.1.0:http {  
    netmask 255.255.255.0 broadcast 192.168.1.255  
    use pool default_webservers  
}
```

Figure 2.20 A catch-all web server configuration.

This configuration directs a web connection destined to any address within the subnet **192.168.1.0/24** to the **default_webservers** pool.

Forwarding virtual servers

A *forwarding virtual server* is just like other virtual servers, except that the virtual server has no nodes to load balance. It simply forwards the packet directly to the node. Connections are added, tracked, and reaped just as with other virtual servers. You can also view statistics for forwarding virtual servers.

To configure forwarding virtual servers using the Configuration utility

1. In the navigation pane, click **Virtual Servers**.
The Virtual Servers screen opens.
2. Click the **Add** button.
The Add Virtual Server screen opens.
3. Type the virtual server attributes, including the address and port number.
4. Under **Configure Basic Properties**, clear the check from **Enable Arp**.
5. On the Select Physical Resources screen, click the **Forwarding** button.
6. Click the **Apply** button.

To configure a forwarding virtual server from the command line

Use the following syntax to configure forwarding virtual servers:

```
b virtual <virt_ip>:<service> forward  
b virtual <virt_ip>:<service> arp disable
```

For example, to allow only one service in:

```
b virtual 206.32.11.6:80 forward  
b virtual <virt_ip>:<service> arp disable
```

Use the following command to allow only one server in:

```
b virtual 206.32.11.5:0 forward  
b virtual <virt_ip>:<service> arp disable
```

To forward all traffic, use the following command:

```
b virtual 0.0.0.0:0 forward
```

In some of the configurations described here, you need to set up a wildcard virtual server on one side of the Link Controller to load balance connections across transparent devices. You can create another wildcard virtual server on

the other side of the Link Controller to forward packets to virtual servers receiving connections from the transparent devices and forwarding them to their destination.

◆ Tip

*If you do not want Link Controller to load balance your traffic but do want to take advantage of certain pool attributes, you can instead use a feature called a **forwarding pool**. For more information on forwarding pools, see **Forwarding pools**, on page 2-38.*

◆ Note

*If a forwarding virtual server is to have the same IP address as a node in an associated VLAN, you must perform some additional configuration tasks. These tasks consist of: creating a VLAN group that includes the VLAN in which the node resides, assigning self-IP addresses to the VLAN group, and disabling the virtual server on the relevant VLAN. For information on creating VLAN groups and assigning self IP addresses to them, see **Creating VLAN groups**, on page 1-13. For information on disabling a virtual server for a specific VLAN, see **Enabling or disabling a virtual server**, on page 2-49.*

Virtual server options

For each type of virtual server, you can configure several options. These options are shown in 2.9.

Option	Description
Display virtual server information	Using the bigpipe virtual show command, you can display information about one or more virtual servers.
Enable/disable a virtual server	By default, a virtual server is enabled. If you want to disable a virtual server, you can do so.
Enable/disable a virtual address	By default, a virtual address is enabled. If you want to disable a virtual address, you can do so.
*Override netmask and broadcast	*You can override the default netmask and broadcast for a network virtual address.
Enable/disable address and port translation	You can turn port translation off for a virtual server if you want to use the virtual server to load balance connections to any service.
Reset connections when service is down	You can configure the Link Controller to automatically reset connections to the virtual server when the service is down.
Enable/disable dynamic connection rebinding	You can cause any connections that were made to a node address or service to be redirected to another node, if the original node transitions to a down state.

Table 2.9 Virtual server configuration options

Option	Description
Disable ARP requests	You can control gratuitous ARPs and ARP requests.
Enable/disable TCP Resets on timeout	When you enable this option, the virtual server sends a TCP Reset when a TCP connection is timed out. The default setting for this option is Enabled .
Set FastFlow packet acceleration	You can speed up packet flow for TCP connections when the packets are not fragmented.
Set a connection limit	You can set a concurrent connection limit on one or more virtual servers.
Mirror state information	You can use mirroring to maintain the same state information in the standby unit that is in the active unit, allowing transactions such as FTP file transfers to continue as though uninterrupted.
Define a last hop pool	You can direct reply traffic to the last hop router using a last hop pool. This overrides the auto_lasthop setting.
Define resources	You can configure a virtual server to use a pool or reference a rule. You can also specify that the virtual server is to function as a forwarding virtual server.
Load balance traffic for any IP protocol	You can configure a virtual server to load balance traffic that uses IP protocols other than TCP and UDP. For example, you can load balance IPSEC traffic from virtual private networks (VPNs).
Delete a virtual server	You can delete an existing virtual server.
Reset virtual server statistics	You can reset virtual server statistics.

Table 2.9 Virtual server configuration options

Displaying information about virtual addresses

You can display information about the virtual addresses that host individual virtual servers. Use the following syntax to display information about one or more virtual addresses included in the configuration:

```
b virtual <virt_ip> [... <virt_ip> ] show
```

The command displays information such as the virtual servers associated with each virtual address, the status, and the current, total, and maximum number of connections managed by the virtual address since the Link Controller was last rebooted, or since the Link Controller became the active unit (redundant configurations only).

Enabling or disabling a virtual server

You can remove an existing virtual server from network service, or return the virtual server to service, using the **disable** and **enable** keywords. As an option, you can enable or disable a virtual server for a specific VLAN only.

When you disable a virtual server, the virtual server no longer accepts new connection requests, but it allows current connections to finish processing before the virtual server goes to a **down** state.

To disable or enable a virtual server using the Configuration utility

1. In the navigation pane, click **Virtual Servers**.
The list of virtual servers displays.
2. Click on the virtual server that you want to disable or enable.
This displays the properties for that virtual server.
3. If you want to disable the virtual server for all VLANs, clear the **Enabled** check box. If you want to disable the virtual server for a specific VLAN, locate the **VLANs Disabled** box and move the relevant VLAN name from the **Existing** list to the **Disabled** list, using the Move button (>>).
4. If you want to enable the virtual server for all VLANs, click the **Enabled** check box (if not already checked). If you want to enable the virtual server for a specific VLAN, locate the **VLANs Disabled** box and move the relevant VLAN name from the **Disabled** list to the **Existing** list, using the Move button (>>).
5. Click **Done**.

◆ Note

*If the **Enabled** check box is checked and no VLANs are listed in the **Disabled** list of the **VLANs Disabled** box, the virtual server is enabled for all VLANs. If the **Enabled** check box is **not** checked, the virtual server is disabled for all VLANs.*

To disable or enable a virtual server from the command line

Use the following syntax to disable a virtual server from network service:

```
b virtual <virt_ip>:<service> [...<virt_ip>:<service>] disable
```

If you want to disable or enable a virtual server for one or more specific VLANs only, use the following syntax:

```
b virtual <virt_ip>:<service> vlans <vlan_list> disable | enable
```

Use the following syntax to return a virtual server to network service:

```
b virtual <virt_ip>:<service> enable
```

◆ Note

*If you do not specify a VLAN name with the **b virtual** command, the virtual server is enabled or disabled on all VLANs.*

Enabling or disabling a virtual address

You can remove an existing virtual address from network service, or return the virtual address to service, using the **disable** and **enable** keywords. Note that when you enable or disable a virtual address, you inherently enable or disable all of the virtual servers that use the virtual address.

```
b virtual <virt_ip> disable
```

Use the following syntax to return a virtual address to network service:

```
b virtual <virt_ip> enable
```

Setting a user-defined netmask and broadcast

The default netmask for a virtual address, and for each virtual server hosted by that virtual address, is determined by the network class of the IP address entered for the virtual server. The default broadcast is automatically determined by the Link Controller, and it is based on the virtual address and the current netmask. You can override the default netmask and broadcast for a network virtual address only.

All virtual servers hosted by the virtual address use the netmask and broadcast of the virtual address, whether they are default values or they are user-defined values.

To set a custom netmask and broadcast

If you want to use a custom netmask and broadcast, you define both when you define the network virtual server:

```
b virtual <virt_ip>[:<service>] [vlan <vlan_name> disable | enable] [netmask <ip>]
[broadcast <ip>] use pool <pool_name>
```

◆ Note

The Link Controller calculates the broadcast based on the IP address and the netmask. In most cases, a user-defined broadcast address is not necessary.

Again, even when you define a custom netmask and broadcast in a specific network virtual server definition, the settings apply to all virtual servers that use the same virtual address. The following sample command shows a user-defined netmask and broadcast:

```
b virtual www.SiteOne.com:http netmask 255.255.0.0\
broadcast 10.0.140.255 use pool my_pool
```

The **/bitmask** option shown in the following example applies network and broadcast address masks. In this example, a 24-bit bitmask sets the network mask and broadcast address for the virtual server:

```
b virtual 206.168.225.0:80/24 use pool my_pool
```

You can generate the same broadcast address by applying the **255.255.255.0** netmask. The effect of the bitmask is the same as applying the **255.255.255.0** netmask. The broadcast address is derived as **206.168.225.255** from the network mask for this virtual server.

Setting translation properties for virtual addresses and ports

Turning off port translation for a virtual server is useful if you want to use the virtual server to load balance connections to any service.

You can also configure the translation properties for a virtual server address. This option is useful when the Link Controller is load balancing devices that have the same IP address. This is typical with the nPath routing configuration where duplicate IP addresses are configured on the loopback device of several servers.

To enable or disable port translation

Use the following syntax to enable or disable port translation for a virtual server:

```
b virtual <virt_ip>:<service> translate port enable | disable | show
```

To enable or disable address translation

Use the following syntax to enable or disable address translation for a virtual server:

```
b virtual <virt_ip>:<service> translate addr enable | disable | show
```

Resetting connections when a service is down

Using either the Configuration utility or the **bigpipe virtual** command, you can configure the Link Controller to automatically reset connections to the virtual server when the service becomes unavailable.

To reset connections using the Configuration utility

1. In the navigation pane, click **Virtual Servers**.
2. In the list of virtual servers, click a virtual server.
3. Click the Virtual Server Properties tab.
4. Check the **Enable Reset on Service Down** box.
5. Press the **Apply** button.

To reset connections from the command line

Use the following command-line syntax:

```
b virtual <ip address>:<service> svc_down reset enable
```

Setting dynamic connection rebinding

Dynamic connection rebinding is a feature for those virtual servers that are load balancing transparent devices such as firewalls or routers. Dynamic connection rebinding causes any connections that were made to a node address or service to be redirected to another node, if the original node transitions to a **down** state. In this case, all connections to the failed node that were made through the virtual server are moved to a newly-selected

node from the virtual server's pool. The new node is selected using the pool's load-balancing algorithm. By default, dynamic connection rebinding is disabled.

◆ **Note**

This feature does not apply to virtual servers for non-transparent devices because they usually involve application state between the client and server node. This state cannot be recreated with a newly-selected node.

To enable, disable, or show the status of dynamic connection rebinding, you can use either the Configuration utility or the **bigpipe virtual** command.

To set dynamic connection rebinding using the Configuration utility

To set dynamic connection rebinding using the Configuration utility, use the following procedure.

1. In the Navigation pane, click **Virtual Servers**.
The Virtual Servers screen opens.
2. Click the IP address for the virtual server that you want to modify.
The Properties page for the virtual server opens.
3. Check the **Enable Connection Rebind** box.
4. Click the **Apply** button.

To set dynamic connection rebinding from the command line

To manage dynamic connection rebinding using the **bigpipe virtual** command, type one of the following commands.

```
b virtual <ip>:<service> conn rebind enable
b virtual <ip>:<service> conn rebind disable
b virtual <ip>:<service> conn rebind show
```

Disabling ARP requests

By default, the Link Controller responds to ARP requests for the virtual server address and sends a gratuitous ARP request for router table updates.

To disable ARP requests using the Configuration utility

1. In the navigation pane, click **Virtual Servers**.
The list of virtual servers displays.
2. Click the virtual server that you want to disable or enable.
This displays the properties for that virtual server.
3. Clear the **Enable ARP** check box.
4. Click the **Apply** button.

To disable ARP requests from the command line

To disable ARP requests from the command line, use the following command line syntax:

```
b virtual <ip address>:<service> arp disable
```

Turning software acceleration off for virtual servers using IPFW rate filters

The software acceleration feature speeds packet flow for TCP connections when the packets are not fragmented. For configurations with no IPFW rate filter present, software acceleration is turned on by default by giving the global variable **fastflow_active** a default value of **auto**. The variable **fastflow_active auto** enables acceleration globally if IPFW filters are not present, and disables it globally if IPFW filters are present. (This is because, with acceleration on, IPFW only examines the first SYN packet in any given connection, rather than filtering all packets.) If you want to turn on acceleration globally but turn it off for the specific virtual servers that use IPFW rate filters, you must change the **fastflow_active** setting from **auto** to **on**, then disable the virtual servers individually using the **bigpipe virtual <ip>:<service> accelerate disable** command.

To set software acceleration controls from the command line

To enable software acceleration globally in a way that can be overridden for individual virtual servers, set the **bigpipe global** variable **fastflow_active** to **on** with the following command:

```
b global fastflow_active on
```

Then, to disable software acceleration for individual virtual servers that use IPFW rate filtering, use the following **bigpipe** command:

```
b virtual <ip>:<service> accelerate disable
```

For example, if you want to turn acceleration off for the virtual server **10.10.10.50:80**, type the following command:

```
b virtual 10.10.10.50:80 accelerate disable
```

You can define a virtual server with acceleration disabled using the following syntax:

```
b virtual <ip>:<service> use pool the_pool accelerate disable
```

For example, if you want to define the virtual server **10.10.10.50:80** with the pool **IPFW_pool** and acceleration turned off, type the following command:

```
b virtual 10.10.10.50:80 use pool IPFW_pool accelerate disable
```

Setting a connection limit

The default setting is to have no limit to the number of concurrent connections allowed on a virtual server.

To set a concurrent connection limit

You can set a concurrent connection limit on one or more virtual servers using the following command:

```
b virtual <virt_ip> [:<service>] [...<virt_ip>[:<service>]]\
limit <max_conn>
```

The following example shows two virtual servers set to have a concurrent connection limit of 5000 each:

```
b virtual www.SiteOne.com:http www.SiteTwo.com:ssl limit 5000
```

To turn off the connection limit

To turn off the connection limit, set the **<max conn>** variable to zero:

```
b virtual <virt_ip>[:<service>] [...<virt_ip>[:<service>]] \
limit 0
```

Mirroring virtual server state

Mirroring provides seamless recovery for current connections, persistence information, SSL persistence, or sticky persistence when a Link Controller fails. When you use the mirroring feature, the standby unit maintains the same state information as the active unit. Transactions such as FTP file transfers continue as though uninterrupted.

◆ Note

Mirroring slows Link Controller performance and is primarily useful for long-lived services like FTP and Telnet. Mirroring is not useful for short-lived connections like HTTP.

Since mirroring is not intended to be used for all connections and persistence, it must be specifically enabled for each virtual server.

To control mirroring for a virtual server

To control mirroring for a virtual server, use the **bigpipe virtual mirror** command to enable or disable mirroring of persistence information, or connections, or both. The syntax of the command is:

```
b virtual <virt addr>:<service> mirror [conn] enable|disable
```

To mirror connection information for the virtual server

Use the **conn** argument to mirror connection information for the virtual server. To display the current mirroring setting for a virtual server, use the following syntax:

```
b virtual <virt addr>:<service> mirror [conn] show
```

If you do not specify **conn** for connection information, the Link Controller assumes that you want to display that information.

◆ Note

If you set up mirroring on a virtual server that supports FTP connections, you need to mirror the control port virtual server, and the data port virtual server.

The following example shows the two commands used to enable mirroring for virtual server **v1** on the FTP control and data ports:

```
b virtual v1:21 mirror conn enable
b virtual v1:20 mirror conn enable
```

Setting up last hop pools for virtual servers

In cases where you have more than one router sending connections to a Link Controller, connections are automatically sent back through the same router from which they were received when the **auto_lasthop** global variable is enabled, as it is by default. If you want to exclude one or more routers from **auto-lasthop**, or if the global **auto_lasthop** is disabled for any reason (for example, you may not want it for an SSL gateway), you can use a last hop pool instead. (If **auto_lasthop** is enabled, the last hop pool takes precedence over it.)

To configure a last hop pool, you must first create a pool containing the router inside addresses. After you create the pool, use the following syntax to configure a last hop pool for a virtual server:

```
b virtual <virt_ip>:<service> lasthop pool <pool_name> | none | \
show
```

Referencing Link Controller resources

Once you have created a pool or a rule, you must configure the virtual server to reference the pool or rule. You can configure a virtual server to reference a pool or rule by using either the Configuration utility or the **bigpipe** command.

To configure a virtual server that references a pool or rule using the Configuration utility

1. In the navigation pane, click **Virtual Servers**.
The Virtual Servers screen opens.
2. Add the attributes you want for the virtual server such as **address**, **port**, **unit ID**, and **interface**.
3. In the Resources section, click **Pool** or **Rule**.
4. In the list of pools or rules, select the pool or rule you want to apply to the virtual server.
5. Click the **Apply** button.

To configure a virtual server that references a rule from the command line

There are several elements required for defining a virtual server that references a rule from the command line:

```
b virtual <virt_serv_key> { <virt_options> <rule_name_reference> }
```

Each of these elements is described in Table 2.10.

Rule element	Description
<virt_serv_key>	A virtual server key definition: <virtual_address>:<virt_port> [unit <ID>]
<virt_options>	Virtual server options. For more information, see <i>Virtual server options</i> , on page 2-48.
<rule_name_reference>	A rule name reference. Rule names are strings of 1 to 31 characters. use rule <rule_name>

Table 2.10 The command line rule elements

Load balancing traffic for any IP protocol

Link Controller allows you to configure a virtual server to load balance IP traffic other than TCP and UDP traffic. This option can be configured on both translating and non-translating virtual servers.

One benefit of this feature is that you can load balance virtual private network (VPN) client connections across several VPNs, eliminating the possibility of a single point of failure. A typical use of this feature would be for load balancing multiple VPN gateways in an IPSEC VPN sandwich, using non-translating virtual servers.

An important point to note, however, is that although address translation of such protocols can be optionally activated, some protocols, such as IPSEC in AH mode, rely on the IP headers remaining unchanged. In such cases, you should use non-translating network virtual servers.

By default, this feature is disabled on a virtual server.

To enable load balancing for any IP protocol using the Configuration utility

1. In the navigation pane, click **Virtual Servers**.
2. In the **Virtual Server** list, click the virtual server for which you want to enable any IP load balancing.
3. Click the Virtual Address Properties tab.
4. In the **Any IP Traffic** section of the screen, check the **Enable** box.
5. Click the **Apply** button.

To enable load balancing for any IP protocol from the command line

Use the **any_ip** option with the **bigpipe virtual** command, as shown in the following syntax:

```
b virtual <ip address:service> any_ip enable
```

Deleting a virtual server

Use the following syntax to permanently delete one or more virtual servers from the Link Controller configuration:

```
b virtual <virt_ip>:<service> [... <virt_ip>:<service>] delete
```

Resetting statistics for a virtual server

Use the following command to reset the statistics for an individual virtual server:

```
b virtual [<virt_ip:port>] stats reset
```

Using other Link Controller features with virtual servers

After you create a pool and define a virtual server that references the pool, you can set up additional features, such as network address translation (NATs) or extended content verification (ECV). For details on network address translations, see *NATs*, on page 2-74. For details on persistence for connections that should return to the node to which they last connected, see *Persistence*, on page 2-21.

Nodes

Nodes are the network devices to which the Link Controller passes traffic. A network device becomes a node when it is added as a member to a load balancing pool. You can display information about nodes and set properties for nodes.

The attributes you can configure for a node are listed in Table 2.11.

Node Attributes	Description
Enable/Disable nodes	You can enable or disable nodes independent of a load balancing pool.
Set node up/down	You can set a node to up or down .
Connection limit	You can place a connection limit on a node.
Associate a node with a monitor	You can associate a health monitor with a node, creating an instance of that monitor.
Add a node as a member of a pool	You can add a node to a pool as a member. This allows you to use the load balancing and persistence methods defined in the pool to control connections handled by the node.

Table 2.11 The attributes you can configure for a node.

To enable and disable nodes and node addresses

A node must be enabled in order to accept traffic. When a node is disabled, it allows existing connections to time out or end normally and accept new connections only if they belong to an existing persistence session. (In this way a disabled node differs from a node that is set **down**. The **down** node allows existing connections to time out, but accepts no new connections.)

To enable a node or node address, use the **node** command with the **enable** option:

```
b node 192.168.21.1 enable
```

To disable a node or node address, use the **node** command with the **disable** option:

```
b node 192.168.21.1 disable
```

To mark nodes and node ports up or down

A node must be marked **up** in order to accept traffic. When a node is marked **down** it allows existing connections to time out but accepts no new connections.

To mark a node **down**, specify the **node** command with a node address and the **down** option. (Note that marking a node **down** prevents the node from accepting new connections. Existing connections are allowed to complete.)

```
b node 192.168.21.1 down
```

To mark a node up, use the **node** command with the **up** option:

```
b node 192.168.21.1 up
```

To mark a particular service **down**, specify the **node** command with a node address and port, and the **down** option. (Note that marking a port **down** prevents the port from accepting new connections. Existing connections are allowed to complete.)

```
b node 192.168.21.1:80 down
```

To mark a particular port **up**, use the **node** command with **up** option:

```
b node 192.168.21.1:80 up
```

To set connection limits for nodes

Use the following command to set the maximum number of concurrent connections allowed on a node:

```
b node <node_ip>[:<service>][...<node_ip>[:<service>]] limit <max conn>
```

To remove a connection limit, set the **<max conn>** variable to **0** (zero). For example:

```
b node 192.168.21.1:80 limit 0
```

The following example shows how to set the maximum number of concurrent connections to **100** for a list of nodes:

```
b node 192.168.21.1 192.168.21.1 192.168.21.1 limit 100
```

To remove a connection limit, set the **<max conn>** variable to **0** (zero).

To associate a health monitor with a node

Use the following command to associate a health monitor with a node:

```
node <node> monitor use <monitor>
```

A monitor can be placed on multiple nodes and a node can have multiple monitors placed on it. To place a monitor on multiple nodes:

```
node <node_list> monitor use <monitor>
```

To place multiple monitors on a node:

```
node <node> monitor use <monitor1> and <monitor2>...
```

For more information on using the **node** command with health monitors, refer to *Health monitors*, on page 2-79.

To display status of all nodes

When you issue the **bigpipe node show** command, the Link Controller displays the node status (**up** or **down**, or **unchecked**), and a node summary of connection statistics, which is further broken down to show statistics by port. To display the status of a node from the command line, type the following command:

```
b node show
```

The report shows the following information:

- Current number of connections
- Total number of connections made to the node since last boot
- Maximum number of concurrent connections since the last boot
- Concurrent connection limit on the node
- The total number of connections made to the node since last boot
- Total number of inbound and outbound packets and bits

Figure 2.21 shows the output of this command.

```
bigpipe node 192.168.200.50:20
NODE 192.168.200.50      UP
|   (cur, max, limit, tot) = (0, 0, 0, 0)
|   (pkts,bits) in = (0, 0), out = (0, 0)
+-  PORT 20              UP
    (cur, max, limit, tot) = (0, 0, 0, 0)
    (pkts,bits) in = (0, 0), out = (0, 0)
```

Figure 2.21 Node status and statistics

To display the status of individual nodes and node addresses

Use the following command to display status and statistical information for one or more node addresses:

```
b node 192.168.21.1 show
```

The command reads the status of each node address, the number of current connections, total connections, and connections allowed, and the number of cumulative packets and bits sent and received.

Use the following command to display status and statistical information for one or more specific nodes:

```
b node 192.168.21.1:80 show
```

To reset statistics for a node

Use the following command to reset the statistics for an individual node address:

```
b node [<node_ip>:<service>] stats reset
```

To add a node as a member of a pool

You can add a node as a member of a load balancing pool. For detailed information about how to do this, see *Member specification*, on page 2-5.

Services

Services are the standard Internet applications supported by Link Controller, such as **HTTP**, **HTTPs**, **FTP**, and **POP3**. Each service is known by its name and also by its well-known or reserved port number such as **80** or **443**. (Specifically, a service is any valid service name in the `/etc/services` file or any valid port number between **0** and **65535**.) The **bigpipe service** command allows you to enable and disable network traffic on services, and also set connection limits and timeouts. You can use the service name or the port number for the `<service>` parameter. Note that the settings you define with this command control the service for all virtual servers that use it. By default, access to all services is disabled.

Table 2.12 lists the attributes you can configure for a service.

Attributes	Description
Allow access to services	As a security measure, all services are locked down on the Link Controller. In order for the Link Controller to load balance traffic, you must enable access to the service on which the Link Controller will receive traffic.
Connection limits	You can define a connection limit for a service so that a flood of connections does not overload the Link Controller.
Set idle connection timeouts	You can set the idle connection timeout to close idle connections.

Table 2.12 *The attributes you can configure for a service.*

◆ Tip

Virtual servers using the same service actually share a port on the Link Controller. This command is global, you only need to open access to a port once; you do not need to open access to a port for each instance of a virtual server that uses it.

To allow access to services using the Configuration utility

Any time you create a virtual server and define a port or service with the Configuration utility, the port or service is automatically enabled.

To allow access to services from the command line

Using the **bigpipe service** command, you can allow access to one or more services at a time.

```
b service <service>...<service> <protocol> [tcp|udp] enable
```

For example, in order to enable HTTP (service 80) and Telnet (service 23) services, you can type the following **bigpipe service** command:

```
b service 80 23 443 tcp enable
```


To set connection limits on services

Use the following syntax to set the maximum number of concurrent connections allowed on a service. Note that you can configure this setting for one or more services.

```
b service <service> [...<service>] limit <max conn>
```

To turn off a connection limit for one or more services, use the same command, setting the **<max conn>** parameter to **0** (zero) like this:

```
b service <service> [...<service>] limit 0
```

To enable or disable TCP for services

You can enable or disable TCP for specific services. The default setting for all services is enabled. Use the following syntax to disable TCP for one or more services:

```
b service <service> [...<service>] tcp disable
```

To re-enable TCP, use this syntax:

```
b service <service> [...<service>] tcp enable
```

To enable or disable UDP for services

You can enable or disable UDP for specific services. The default setting for all services is disabled. Use the following syntax to enable UDP for one or more services:

```
b service <service> [...<service>] udp enable
```

To disable UDP, use this syntax:

```
b service <service> [...<service>] udp disable
```

To set the idle connection timeout for TCP traffic

To set the TCP timeout on one or more services, where the **<seconds>** parameter is the number of seconds before an idle connection is dropped, use the following syntax:

```
b service <service> [<service>...] timeout tcp <seconds>
```

For example, the following command sets the TCP timeout to **300** seconds for port 53:

```
b service 53 timeout tcp 300
```

To turn off TCP timeout for a service, use the above command, setting the **<seconds>** parameter to zero:

```
b service 53 timeout tcp 0
```

To set the idle connection timeout for UDP traffic

To set the UDP timeout on one or more services, where the **<seconds>** parameter is the number of seconds before an idle connection is dropped, use the following syntax:

```
b service <service> [<service>...] timeout udp <seconds>
```

For example, the following command sets the UDP timeout to **300** seconds for port 53:

```
b service 53 timeout udp 300
```

To turn off UDP timeout for a service, use the above command, setting the **<seconds>** parameter to zero:

```
b service 53 timeout udp 0
```

To display service settings

Use the following command to display the settings for all services:

```
b service show
```

Use the following syntax to display the settings for a specific service of services:

```
b service <service> [...<service>] show
```

For example, the command **b service http show** displays the output shown in Figure 2.22.

```
SERVICE 80 http tcp enabled timeout 1005 udp disabled timeout 60
      (cur, max, limit, tot, reaped) = (0, 0, 0, 0, 0)
      (pkts,bits) in = (0, 0), out = (0, 0)
```

*Figure 2.22 Sample output of the **bigpipe service show** command*

Address translation: SNATs, NATs, and IP forwarding

The Link Controller uses address translation and forwarding in various ways to make nodes accessible that would otherwise be hidden on its internal VLAN.

- ◆ A virtual server translates the destination address of an inbound packet from its own address (the virtual server's) to the address of the node to which it load balances the packet. It then translates the origin address of the reply back to its own address so the originating host will not try to address the member node directly. This translation is basic to the way the virtual server works in most configurations and it is enabled by default.
- ◆ You can configure a SNAT (Secure Network Address Translation) or NAT (Network Address Translation) to give a node that is a member of a load balancing pool a routable address as an origin address for purposes of generating its own outbound traffic. A SNAT can be configured manually, or automatically using the SNAT auto-map feature.
- ◆ You can configure a forwarding virtual server to expose selected nodes to the external network.
- ◆ You can configure IP forwarding globally to expose all internal nodes to the external network

For more information on enabling address translation for virtual servers, refer to *Virtual servers*, on page 2-40. The following sections describe how to configure SNATs, NATs, and IP forwarding.

SNATs

A **secure network address translation** (SNAT) provides a routable alias IP address that a node can use as its source IP address when making connections to clients on the external network. Unlike a **network translation address** (NAT), a SNAT does not accept inbound traffic, and this is where its security lies. When you define a SNAT, you can use it in any of the following ways:

- Assign a single SNAT address to a single node
- Assign a single SNAT address to multiple nodes
- Enable a SNAT for a VLAN

Note that a SNAT address does not necessarily have to be unique; for example, it can match the IP address of a virtual server.

The attributes you can configure for a SNAT are shown in Table 2.13.

Attributes	Description
Global SNAT properties	Before you configure a SNAT, you can configure global properties for all SNATs on the Link Controller. Configuring global properties for a SNAT is optional.
Manual SNAT mapping	You can define a specific translation address to be mapped to an individual host.
SNAT automapping	You can configure Link Controller to automatically map a translation address.

Table 2.13 *The attributes you can configure for a SNAT*

Setting SNAT global properties

The SNAT feature supports three global properties that apply to all SNAT addresses:

- ◆ **Connection limits**
The connection limit applies to each node that uses a SNAT.
- ◆ **TCP idle connection timeout**
This timer defines the number of seconds that TCP connections initiated using a SNAT address are allowed to remain idle before being automatically disconnected.
- ◆ **UDP idle connection timeout**
This timer defines the number of seconds that UDP connections initiated using a SNAT address are allowed to remain idle before being automatically disconnected. This value should not be set to **0**.

To configure SNAT global properties using the Configuration utility

1. In the navigation pane, click **SNATs**.
The SNATs screen opens.
2. In the **Connection Limit** box, type the maximum number of connections you want to allow for each node using a SNAT.
3. To turn connection limits off, set the limit to **0**.
4. In the **TCP Idle Timeout** box, type the number of seconds that TCP connections initiated by a node using a SNAT are allowed to remain idle.
5. In the **UDP Idle Timeout** box, type the number of seconds that UDP connections initiated by a node using a SNAT are allowed to remain idle. This value should not be set to **0**.
6. Click the **Apply** button.

To configure SNAT global properties from the command line

Configuring global properties for a SNAT requires that you enter three **bigpipe** commands. The following command sets the maximum number of connections you want to allow for each node using a SNAT.

```
b snat limit <value>
```

The following commands set the TCP and UDP idle connection timeouts:

```
b snat timeout tcp <seconds>
```

```
b snat timeout udp <seconds>
```

Configuring a SNAT manually

Once you have configured the SNAT global properties, you can manually configure SNAT address mappings. When you map a SNAT manually, you specify a particular translation IP address that you want the Link Controller to assign from any of the following:

- One or more specified node addresses
- One or more VLANs
- A combination of specified node addresses and VLANs
- All node addresses (known as a default SNAT)

Note that a SNAT address does not necessarily have to be unique; for example, it can match the IP address of a virtual server. A SNAT address cannot match an address already in use by a NAT or another SNAT address.

The following sections describe how to add a default SNAT and how to add a SNAT manually for individual node addresses, VLANs, or a combination of both.

Adding a default SNAT manually

If you do not want to configure a SNAT for each individual node, you can manually create a default SNAT. When you add a default SNAT, you are directing the Link Controller to map every node on the internal network to a default translation address.

To add a default SNAT manually using the Configuration utility

1. In the navigation pane, click **NATs**.
The NATs screen displays.
2. Click the SNATs tab.
3. Click the **Add Default** button.
The Add Default SNAT screen opens.
4. In the **Translation Address** section, click **IP**, and type the IP address that you want Link Controller to assign as a translation address.
5. Click **Done**.

To add a default SNAT manually from the command line

Use the following syntax to manually define the default SNAT. If you use the netmask parameter and it is different from the external interface default netmask, the command sets the netmask and derives the broadcast address.

```
b snat map default to <snat_ip> [vlan <vlan_name>\
disable|enable] [netmask <ip>]
```

Adding a SNAT for individual node addresses and VLANs

If you do not want to add a default SNAT, you can add a SNAT for any individual node address or VLAN. The following procedures describe how to manually add a SNAT.

To manually add a SNAT using the Configuration utility

The Configuration utility allows you to define one SNAT for one or more original IP addresses, where the original IP address can be either a specific node address or a VLAN name.

1. In the navigation pane, click **NATs**.
The NATs screen displays.
2. Click the SNATs tab.
3. Click the **Add** button.
The Add SNAT screen opens.
4. In the **Translation Address** section, click **IP**, and type the IP address that you want Link Controller to assign as a translation address.
5. Type each node's IP address into the **Original Address:** box and move the address to the **Current List:** box, using the Move button (>>). Also, verify that the option **choose** appears in the **VLAN** box.
6. If you want to map the translation address from a VLAN, select the VLAN name from the **VLAN** box and move the selection to the **Current List:** box, using the Move button (>>).
7. Click **Done**.

To add a manual SNAT from the command line

The **bigpipe snat** command defines one SNAT for one or more original IP addresses, where the original IP address can be either a specific node address or a VLAN name. To manually add a SNAT using the **bigpipe snat** command, use the following syntax.

```
b snat map <orig_ip>... to <snat_ip>
```

For example, to define a SNAT for two specific nodes:

```
b snat map 192.168.75.50 192.168.75.51 to 192.168.100.10
```

To define a SNAT for two internal VLANs:

```
b snat map internal1 internal2 to 192.168.102.11
```

To define a SNAT for both a node address and a VLAN:

```
b snat map 192.168.75.50 internal2 to 192.168.100.12
```

To create individual SNAT addresses

Use the following command-line syntax to create a SNAT mapping:

```
b snat map <orig_ip> [...<orig_ip>] to <snat_ip>\
[vlan <vlan_name> disable | enable] [unit <unit ID>]\
[netmask <ip>]
```

If the netmask is different from the external interface default netmask, the command sets the netmask and derives the broadcast address.

Configuring SNAT automapping

Link Controller includes a feature called SNAT automapping. When you map a SNAT automatically, rather than manually, you enable the Link Controller to choose the translation IP address. You also enable the Link Controller to map that translation address from any of the following:

- One or more specified node address
- One or more VLANs
- A combination of specific node addresses and VLANs
- All node addresses (known as a default SNAT)

SNAT automapping eliminates the need for you to specifically define an IP address as the translation address.

The SNAT automapping feature is useful in the following cases:

- Where there is a need to ensure that outbound traffic returning through ISPs or NAT-less firewalls returns through the same ISP or firewall.
- Where a traditional single SNAT address would quickly exhaust the number of ephemeral ports available. As long as there is more than one eligible self IP address, SNAT automapping can increase the number of simultaneous connections possible by using the same ephemeral port on multiple addresses.
- When the *equivalent* of a default SNAT is required for Link Controller units in active-active mode. (The conventional default SNAT does not work in active-active mode.)

Adding an automapped default SNAT

The Link Controller allows you to take advantage of the SNAT automapping feature when adding a default SNAT. When you add a default SNAT, you are enabling the Link Controller to map every node on the internal network to a default translation address. With the automapping feature, you do not need to define a specific translation address to which all nodes on the network will be mapped.

To add the automapped default SNAT using the Configuration utility

1. In the navigation pane, click **NATs**.
The NATs screen displays.
2. Click the SNATs tab.
3. Click the **Add Default** button.
The Add Default SNAT screen opens.
4. Click the **Automap** button.
5. Click **Done**.

To add the automapped default SNAT from the command line

To add a default SNAT using the automapping feature, type the **bigpipe snat** command as follows:

```
b snat map default to auto
```

Adding automapped SNATs for standard (active-standby) configurations

When enabling SNAT automapping for VLANs, the Link Controller handles the SNATs in the following ways:

- If you create a SNAT on an internal VLAN, a SNAT is performed on any connection made from that VLAN.
- If you enable **snat automap** on a single self IP address, the translation address is that self IP address.
- If you enable **snat automap** on more than one self IP address, (implying more than one IP network), the following rules apply:
  - If the connection is handled by a non-forwarding virtual server, the translation address is the self IP address that matches the IP network of the node selected by load balancing.
  - If the connection is handled by a forwarding virtual server or no virtual server, the translation address is the self IP address that matches the IP network of the next hop to the destination.
  - If there are no self addresses that match the IP network of the node or the next hop, any self IP address on the VLAN is eligible.

To add a SNAT using the automapping feature, you must complete two procedures:

- Enable the **snat automap** attribute on any self IP addresses.
- Add the SNAT, specifying the **Automap** feature.

The following sections explain these procedures.

To enable the **snat automap** attribute on a self IP address from the command line

When you enable automapping to add a SNAT, the translation address that the Link Controller maps to an individual node or a VLAN is the self IP address. Thus, prior to enabling automapping for the node or VLAN, you must enable the **snat automap** attribute on the self IP address. This is done from the command line, using the following syntax:

```
b self <self IP address> snat automap enable
```

For example, if you have the two self IP addresses **192.168.217.14** and **192.168.217.15**, the following commands enable the **snat automap** attribute on those self IP addresses:

```
b self 192.168.217.14 snat automap enable
```

```
b self 192.168.217.15 snat automap enable
```

Later, when you add a SNAT using automapping, the Link Controller maps either of those self IP addresses to the original node (or VLAN) that you specify.

As another example, the following command enables the **snat automap** attribute on the self IP address **10.0.0.1**, for the VLAN named **external**:

```
b self 10.0.0.1 vlan external snat automap enable
```

For more information, see *To add an automapped SNAT from the command line*, on page 2-72.

To add an automapped SNAT using the Configuration utility

The Configuration utility allows you to define one SNAT for one or more original IP addresses, where the original IP address can be either a specific node address or a VLAN name.

1. In the navigation pane, click **NATs**.
The **NATs** screen displays.
2. Click the **SNATs** tab.
3. Click the **Add** button.
The Add SNAT screen opens.
4. In the Translation Address dialog area, click **Automap**.
5. If you want to map the translation address from one or more specific nodes, enter each node's IP address into the **Original Address:** box and move the address to the **Current List:** box, using the Move button (>>). Also, verify that the option **choose** appears in the **VLAN** box.
6. If you want to map the translation address to a VLAN, select the VLAN name from the **VLAN** box and move the selection to the **Current List:** field, using the Move button (>>).
7. Click **Done**.

To add an automapped SNAT from the command line

The **bigpipe snat** command defines one SNAT for one or more original IP addresses, where the original IP address can be either a specific node address, or a VLAN name.

For example, to define an automapped SNAT for two individual node addresses:

```
b snat map 10.1.1.1 10.1.1.2 to auto
```

In the preceding example, the translation address to which the nodes **10.1.1.1** and **10.1.1.2** will be mapped is the self IP address, assuming that you enabled the **snat automap** attribute on that self IP address prior to using the **bigpipe snat** command. For more information, see *To enable the snat automap attribute on a self IP address from the command line*, on page 2-71.

To define an automapped SNAT for a VLAN named **internal**:

```
b snat map internal to auto
```

To define an automapped SNAT for both a node address and a VLAN:

```
b snat map 192.168.75.50 internal2 to auto
```

◆ Note

*When adding automapped SNATs, you must also enable the **snat automap** attribute on the self IP address that the Link Controller will use as the translation address. For more information, see *To enable the snat automap attribute on a self IP address from the command line*, on page 2-71.*

ISPs and NAT-less firewalls

Link Controller handles ISPs and NAT-less firewalls in the following manner:

- If multiple external interfaces are available, the inside addresses of the firewalls in the load balancing pool may each be connected to different interfaces and assigned to different VLANs.
- A SNAT is then enabled on each VLAN.
- A SNAT must also be enabled on the internal VLAN.

For example, if the internal VLAN is named **internal** and the external VLANs are named **external1** and **external2**, you would type the following commands:

```
b snat internal to auto
b snat external1 to auto
b snat external2 to auto
```

- If multiple external interfaces are not available, the ISP routers or firewalls are assigned to different IP networks. This will already be the case for ISPs.
- For firewalls, the separate IP address ranges must be established on the inside and outside interfaces of each firewall. The separate networks are then assigned separate self addresses, for example, **10.0.0.1** and **11.0.0.1**.

Thus, if the internal and external VLANs are named **internal** and **external**, you type the following commands:

```
b self 10.0.0.1 vlan external snat automap enable
b self 11.0.0.1 vlan external snat automap enable
b snat internal to auto
```

Disabling SNATs and NATs for a pool

When configuring a pool, you can specifically disable SNAT or NAT translations on any connections that use that pool. By default, this setting is enabled. For information on how to disable SNAT and NAT connections for a pool, see *Disabling SNAT and NAT connections*, on page 2-36.

Disabling ARP requests

By default, the Link Controller responds to ARP requests for the SNAT address and sends a gratuitous ARP request for router table update. If you want to disable the SNAT address for ARP requests, you must specify **arp disable**.

Additional SNAT configuration options

The following procedures allow you to further configure SNATs.

To delete SNAT addresses

The following syntax deletes a specific SNAT:

```
b snat <snat_ip> | default delete
```

To show SNAT mappings

The following **bigpipe** command shows mappings:

```
b snat [<snat_ip> ...] show
b snat default show
```

The value of the **<snat_ip>** variable can be either the translated or the original IP address of the SNAT, or a SNAT-enabled VLAN name.

The following command shows the current SNAT connections:

```
b snat [<snat_ip> ...] dump [ verbose ]
b snat default dump [ verbose ]
```

The optional **verbose** keyword provides more detailed output.

The following command prints the global SNAT settings:

```
b snat globals show
```

To enable mirroring for redundant systems

The following example sets SNAT mirroring for all SNAT connections originating at **192.168.225.100**:

```
b snat 192.168.225.100 mirror enable
```

To clear statistics

You can reset statistics by node address, SNAT address, or VLAN name. Use the following syntax to clear all statistics for one or more nodes:

```
b snat <node_ip> ... stats reset
```

Use the following syntax to clear all statistics for one or more SNAT addresses:

```
b snat <snat_ip> ... stats reset
```

Use the following command to reset the statistics to zero for the default:

```
b snat default stats reset
```

NATs

A *network translation address* (NAT) provides a routable alias IP address that a node can use as its source IP address when making or receiving connections to clients on the external network. (This distinguishes it from a SNAT, which can make outbound connections but refuses inbound connections.) You can configure a unique NAT for each node address included in a virtual server mapping.

Note

Note that NATs do not support port translation, and are not appropriate for protocols that embed IP addresses in the packet, such as FTP, NT Domain or CORBA IIOP. You cannot define any NATs if you configure a default SNAT.

Table 2.14 shows the attributes you can configure for a NAT.

NAT Attributes	Description
Original address	The original address is the node IP address of a host that you want to be able to connect to through the NAT.
Translated address	The translated address is an IP address that is routable on the external network of the Link Controller. This IP address is the NAT address.
Disabled VLAN list	VLANs to which the NAT is not to be mapped can be explicitly disabled, as when there is more than one internal VLAN.
Unit ID	You can specify a unit ID for a NAT if the Link Controller is configured to run in active-active mode.

Table 2.14 *The attributes you can configure for a NAT*

The IP addresses that identify nodes on the Link Controller internal network need not be routable on the external network. This protects nodes from illegal connection attempts, but it also prevents nodes (and other hosts on

the internal network) from receiving direct administrative connections, or from initiating connections to clients, such as mail servers or databases, on the Link Controller external interfaces.

Using network address translation resolves this problem. Network address translations (NATs) assign to a particular node a routable IP address that the node can use as its source IP address when connecting to servers on the Link Controller external interfaces. You can use the NAT IP address to connect directly to the node through the Link Controller, rather than having the Link Controller send you to a random node according to the load balancing mode.

◆ **Note**

In addition to these options, you can set up forwarding virtual servers that allow you to selectively forward traffic to specific addresses. The Link Controller maintains statistics for forwarding virtual servers.

Defining a network address translation (NAT)

When you define standard network address translations (NATs), you need to create a separate NAT for each node that requires a NAT. You also need to use unique IP addresses for NAT addresses; a NAT IP address cannot match an IP address used by any virtual or physical servers in your network. You can configure a NAT with the Configuration utility or from the command line.

To configure a NAT using the Configuration utility

1. In the navigation pane, click **NATs**.
The NATs screen opens.
2. Click the **Add** button.
The Add NAT screen opens.
3. In the Add NAT screen, fill in the fields to configure the NAT. For additional information configuring a NAT, click the **Help** button.
4. Click **Done**.

To configure a NAT from the command line

A NAT definition maps the IP address of a node **<orig_addr>** to a routable address on the external interface **<trans_addr>**. Use the following syntax to define a NAT:

```
b nat <orig_addr> to <trans_addr> [vlans <vlan_list> disable | enable] [unit <unit ID>]
```

The **vlans <vlan_list>** parameter is used to disable the specified VLANs for translation. By default, all VLANs are enabled.

Use the **unit <unit ID>** parameter to specify the Link Controller to which this NAT applies in an active-active redundant system.

The following example shows a NAT definition:

```
b nat 10.10.10.10 to 10.12.10.10
```

To delete NATs

Use the following syntax to delete one or more NATs from the system:

```
b nat <orig_addr> [...<orig_addr>] delete
```

To display status of NATs

Use the following command to display the status of all NATs included in the configuration:

```
b nat show
```

Use the following syntax to display the status of one or more selected NATs (see Figure 2.23).

```
b nat <orig_addr> [...<orig_addr>] show
```

```
NAT { 10.10.10.3 to 9.9.9.9 }
      (pkts,bits) in = (0, 0), out = (0, 0)
NAT { 10.10.10.4 to 12.12.12.12
      netmask 255.255.255.0 broadcast 12.12.12.255 }
      (pkts,bits) in = (0, 0), out = (0, 0)
```

Figure 2.23 Output when you display the status of a NAT

To reset statistics for a NAT

Use the following command to reset the statistics for an individual NAT:

```
b nat [<orig_addr>] stats reset
```

Use the following command to reset the statistics for all NATs:

```
b nat stats reset
```

Disabling SNATs and NATs for a pool

When configuring a pool, you can specifically disable any SNAT or NAT connections that use that pool. By default, this setting is enabled. For information on how to disable SNAT and NAT connections for a pool, see *Disabling SNAT and NAT connections*, on page 2-36.

Disabling ARP requests

By default, the Link Controller responds to ARP requests for the NAT address and sends a gratuitous ARP request for router table update. If you want to disable the NAT address for ARP requests, you must specify **arp disable**.

Additional restrictions

The **nat** command has the following additional restrictions:

- The IP address defined in the **<orig_addr>** parameter must be routable to a specific server behind the Link Controller.
- You must delete a NAT before you can redefine it.

- The interface for a NAT can only be configured when the NAT is first defined.

IP forwarding

IP forwarding is an alternate way of allowing nodes to initiate or receive direct connections from the Link Controller external network. IP forwarding directly exposes all of the node IP addresses to the external network, making them routable on that network. If your network uses the NT Domain or CORBA IIOP protocols, IP forwarding is an option for direct access to nodes.

◆ Tip

*Use of SNATs and NATs, as well as forwarding pools and forwarding virtual servers, is preferable to global IP forwarding. For more information on forwarding pools and forwarding virtual servers, see **Forwarding pools**, on page 2-38 and **Forwarding virtual servers**, on page 2-47.*

IP forwarding is a global setting that exposes the IP address of all internal nodes to the Link Controller external network, and clients can use it as a standard routable address. When you turn IP forwarding on, the Link Controller acts as a router when it receives connection requests for node addresses. You can use the IP filter feature to implement a layer of security that can help protect your nodes.

Table 2.15 shows options associated with IP forwarding.

Option	Description
Enable IP forwarding globally	You can enable IP forwarding globally for the Link Controller, either with the Configuration utility or by turning on the sysctl variable net.inet.ip.forwarding . To protect your nodes with this feature, we recommend that you use IP filters, which add a layer of security.
Address routing issues	If you enable IP forwarding, you need to route packets to the node addresses through the Link Controller.
Configure the forwarding attribute for a pool	Instead of enabling IP forwarding globally or creating a forwarding virtual server, you can create a pool with no members that forwards traffic instead of load balancing it. For more information, see <i>Forwarding pools</i> , on page 2-38.
Enable IP forwarding for a virtual server	Instead of enabling IP forwarding globally, you can create a special virtual server with IP forwarding enabled. For information on creating a forwarding virtual server, see <i>Forwarding virtual servers</i> , on page 2-47.

Table 2.15 The attributes you can configure for IP forwarding

Enabling IP forwarding globally

You can enable IP forwarding globally on the Link Controller system. To set up IP forwarding globally, you need to complete two tasks:

- **Turn IP forwarding on**
The Link Controller uses a system control variable to control IP forwarding, and its default setting is **off**.
- **Verify the routing configuration**
You probably have to change the routing table for the router on the Link Controller external network. The router needs to direct packets for nodes to the Link Controller, which in turn directs the packets to the nodes themselves.

To set global IP forwarding using the Configuration utility

1. In the navigation pane, click **System**.
The Network Map screen opens.
2. Click the Advanced Properties tab.
The Advanced Properties screen opens.
3. Check the **Allow IP Forwarding** box.
4. Click **Apply**.

To set global IP forwarding from the command line

Use the **bigpipe global ip_forwarding** command to set the variable. The default setting for the variable is **disabled**. You should change the setting to **enabled**:

```
b global ip_forwarding enabled
```

Addressing routing issues for IP forwarding

Once you turn on IP forwarding, you probably need to change the routing table on the default router. Packets for the node addresses need to be routed through the Link Controller. For details about changing the routing table, refer to your router's documentation.

Configuring the forwarding attribute for a pool

You can configure IP forwarding so that it is done by a pool, rather than globally by the Link Controller or by an individual virtual server. For more information, see *Forwarding pools*, on page 2-38.

Enabling IP forwarding for a virtual server

You can configure IP forwarding so that it is done by a virtual server, rather than globally by the Link Controller or by a specific pool. For more information, see *Forwarding virtual servers*, on page 2-47.

Health monitors

Health monitors verify connections and services on nodes that are members of load balancing pools. The monitor checks the node at a set interval. If the node does not respond within a specified timeout period, the node is marked **down** and traffic is no longer directed to it.

By default, an **icmp** (Internet Control Message Protocol) monitor is associated with every node that is a member of a load balancing pool. This monitor is of the simplest type, checking only the node address and checking only for a ping response. To change the interval and timeout values of this default check, or to check specific services on a node, you need to configure a custom monitor or monitors to add to the default monitor. The Link Controller provides a variety of service-specific monitors in template form. Some of these monitors are usable as is (assuming their default values are acceptable) and may be put in service simply by associating them with the nodes to be monitored. In most cases, however, the template is used purely as a template for configuring custom monitors. Configuring custom monitors and placing them in service is a three-step process:

- Selecting the template
- Configuring the monitor from the template
- Associating the monitor with the node or nodes

For example, for the default **icmp** monitor, we selected the **icmp** monitor template, as shown in Figure 2.24.

```
monitor type icmp {  
    interval 5  
    timeout 16  
    dest *  
}
```

Figure 2.24 The **icmp** monitor template

The **icmp** monitor template has three attributes, **interval**, **timeout**, and **dest**, each with a default value. (All monitor templates have these three basic attributes. Other monitor templates have additional attributes as required by the service type.) These attributes are inherited by the custom monitor when it is configured and can be left at their default values or assigned new values as required.

For the default monitor, template **icmp** is used as is, that is, as monitor **icmp** with its default attribute values. To change any of these default values, you need to create a custom monitor based upon **icmp**, for example, **my_icmp**. Only the values that are actually to be changed would need to be specified in the definition of the custom monitor. Therefore, if you want to change the timeout values only, you define the custom monitor as follows:

```
b monitor my_icmp '{ use icmp timeout 20 }'
```

This creates a new monitor in `/config/bigip.conf`, as shown in Figure 2.25. You can display this monitor using the command **b monitor my_icmp show**.

```
monitor my_icmp{
  #type icmp
  "icmp"
  interval 5
  timeout 20
}
```

Figure 2.25 Custom *icmp* monitor

Once the custom monitor exists, you associate it with a node or nodes using the Configuration utility or the **bigpipe node** command as follows.

```
b node 11.11.11.1 11.11.11.2 11.11.11.3 monitor use my_icmp
```

◆ **Note**

*The nodes are identified by IP address only. **icmp** can ping addresses only, not specific ports on addresses. This creates three instances of monitor **my_icmp**, one for each address. You can display the instances using the command **b node monitor my_icmp show**.*

```
+ - NODE ADDRESS 11.11.11.1 UP
|
| +- icmp
| 11.11.11.1 up enabled
|
+ - NODE ADDRESS 11.11.11.2 UP
|
| +- icmp
| 11.11.11.2 up enabled
|
+ - NODE ADDRESS 11.11.11.3 UP
|
| +- icmp
| 11.11.11.3 up enabled
```

Figure 2.26 Output for the command **b node monitor show**

Note that each instance takes as its destination the same node it is associated with. This is because the **dest** value in **my_icmp** was left at the default *****, which tells the instance to use the associated node as its destination.

Assigning a specific address to **dest**, such as **11.11.11.1**, would cause the monitor to verify all three addresses by checking that one address, making **11.11.11.2** and **11.11.11.3** dependent on **11.11.11.1**.

Selecting the monitor template

Selecting a template is straightforward. Like **icmp**, each of the templates has a **type** based on the type of service it checks, for example, **http**, **https**, **ftp**, **pop3**, and takes that type as its name. (Exceptions are port-specific templates, like **https_443**, and the **external** template, which calls a user-supplied program.) To select a template, simply select the one that corresponds in name and/or type to the service you want to check. If more than one service is to be checked, for example **http** and **https**, more than one monitor can be placed on the node. (This creates a rule, namely that the node will not be considered up unless both monitors run successful checks.) You may not want to check all services available on a node specifically. If you want to verify only that the destination IP address is alive, or that the path to it through a transparent node is alive, use one of the simple templates, **icmp** or **tcp_echo**. If you want to verify TCP only, use the monitor template **tcp**.

All monitor templates are contained in the read-only file **/etc/base_monitors.conf**. The following sections describe each of the monitor templates, its function, and the information required to configure a monitor from it. The templates are divided into three groups based on the types of monitors they support: simple monitors, ECV (Extended Content Verification) monitors, and EAV (Extended Application Verification) monitors. Also described are the port-specific monitor templates, which are derived from the other types.

Working with templates for simple monitors

Simple monitors are those that check node addresses only and verify simple connections only. Templates for these monitors are **icmp** and **tcp_echo**.

◆ Note

*The templates **icmp** and **tcp_echo** are both usable as is, that is, they may be associated with nodes. It is important to understand, however, that using a template as is means that you are using the default attribute values. To change any of these values, you have to configure a custom monitor based on the template.*

Using icmp

The **icmp** template uses Internet Control Message Protocol to make a simple node check. The check is successful if a response to an ICMP_ECHO datagram is received. **icmp** has no attributes other than the standard **interval**, **timeout**, and **dest**.

```
monitor icmp {
    #type icmp
    interval 5
    timeout 16
    dest *
}
```

Figure 2.27 The **icmp** monitor template

Using `tcp_echo`

The **tcp_echo** template uses Transmission Control Protocol. The check is successful if a response to a TCP ECHO message is received. The **tcp_echo** monitor also supports *transparent* mode. In this mode, the node with which the monitor is associated is pinged through to the destination node. (For more information about transparent mode, refer to *Using transparent and reverse modes*, on page 2-93.)

To use **tcp_echo**, you must ensure that TCP ECHO is enabled on the nodes being monitored.

```
monitor tcp_echo {  
    type tcp_echo  
    interval 5  
    timeout 16  
    dest *      //transparent  
}
```

Figure 2.28 The *tcp_echo* monitor template

Working with templates for ECV monitors

ECV monitors attempt to retrieve explicit content from nodes using **send** and **recv** statements. These include **http**, **https** and **tcp**.

◆ **Note**

*The templates **http**, **https**, and **tcp** are all usable as is, and you may associate them with nodes. It is important to understand, however, that using a template as is means that you are using the default attribute values. To change any of these values, you have to configure a custom monitor based on the template.*

Using `tcp`

The **tcp** template is for Transmission Control Protocol. A **tcp** monitor attempts to receive specific content. The check is successful when the content matches the **recv** expression. A **tcp** monitor takes a **send** string and a **recv** expression. If the **send** string is left blank, the service is considered up if a connection can be made. A blank **recv** string matches any response. Both **transparent** and **reverse** modes are options. (For more information about transparent and reverse modes, refer to *Using transparent and reverse modes*, on page 2-93.)

```
monitor tcp {
  #type tcp
  interval 5
  timeout 16
  dest *:*
  send ""
  recv ""
  //reverse
  //transparent
}
```

Figure 2.29 The *tcp* monitor template

Using http

The **http** template is for HyperText Transfer Protocol. Like a **tcp** monitor, an **http** monitor attempts to receive specific content from a web page, and unlike a **tcp** monitor, sends a user name and password. The check is successful when the content matches the **recv** expression. An **http** monitor uses a **send** string, a **recv** expression, **username**, **password**, and optional **get**, **url**, **transparent** and **reverse** statements. (If there is no password security, use blank strings [""], for **username** and **password**.) The optional **get** statement replaces the send statement, automatically filling in the string "GET". Thus the following two statements are equivalent:

```
send "GET/"
get "/"
```

The optional **url** statement takes the HTTP URL as a value and automatically fills in the **dest** value with the address the URL resolves to. (For more information about transparent and reverse modes, refer to *Using transparent and reverse modes*, on page 2-93.) Both **transparent** and **reverse** modes are also options. (For more information about the **get** and **url** statements, refer to *Using send, receive, url, and get statements*, on page 2-92.)

```
monitor http {
  #type http
  interval 5
  timeout 16
  dest *:*
  send "GET /"
  recv ""
  username ""
  password ""
  //get
  //url
  //reverse
  //transparent
}
```

Figure 2.30 The *http* monitor template

Using https

The **https** template is for Hypertext Transfer Protocol Secure. An **https** monitor attempts to receive specific content from a web page protected by SSL security. The check is successful when the content matches the **recv** expression. An **https** monitor uses a **send** string, a **recv** expression, and a **username** and **password** (If there is no password security, use blank strings [""], for **username** and **password**.) The optional **get** statement replaces the **send** statement, automatically filling in the string "GET". Thus, the following two statements are equivalent:

```
send "GET/"
get "/"
```

The optional **url** statement takes the HTTPS URL as a value and automatically fills in the **dest** value with the address the URL resolves to.

```
monitor https {
  #type https
  interval 5
  timeout 16
  dest *.*
  send "GET /"
  recv ""
  //get
  //url
  username ""
  password ""
}
```

Figure 2.31 The https monitor template

Working with templates for EAV monitors

EAV monitors verify applications on the node by running those applications remotely, using an external service checker program located in the directory **/user/local/lib/pingers**. These include **ftp**, **pop3**, **smtp**, **sq**, **nntp**, **imap**, **ldap**, and **radius**. Also included is the template **external**, which has a **run** attribute to specify a user-added external monitor.

Using ftp

The **ftp** template is for File Transfer Protocol. The monitor attempts to download a specified file to the **/var/tmp** directory. The check is successful if the file is retrieved. The **ftp** monitor takes a **get** statement, **username**, and **password**. The **get** statement takes the full path to the file as a value. The optional **url** statement may be used in place of **get**. The **url** takes the FTP URL as a value and automatically fills in the **dest** value with the address the URL resolves to. (For more information about the **get** and **url** statements, refer to *Using send, receive, url, and get statements*, on page 2-92.)

```
monitor ftp {  
  #type ftp  
  interval 5  
  timeout 16  
  dest *:*  
  username ""  
  password ""  
  get ""  
  //url  
}
```

Figure 2.32 The ftp monitor template

Using pop3

The **pop3** template is for Post Office Protocol. The check is successful if the monitor is able to connect to the server, log in as the indicated user, and log out. The **pop3** monitor requires **username** and **password**.

```
monitor pop3 {  
  #type pop3  
  interval 5  
  timeout 16  
  dest *:*  
  username ""  
  password ""  
}
```

Figure 2.33 The pop3 monitor template

Using smtp

The **smtp** template is for Simple Mail Transport Protocol servers. An **smtp** monitor is an extremely simple monitor that checks only that the server is up and responding to commands. The check is successful if the mail server responds to the standard SMTP **HELO** and **QUIT** commands. An **smtp** monitor requires a **domain** name.

```
monitor smtp {  
  #type smtp  
  interval 5  
  timeout 16  
  dest *:*  
  domain ""  
}
```

Figure 2.34 The smtp monitor template

Using *snmp_dca*

The **snmp_dca** template is used for load balancing traffic to servers that are running an SNMP agent, such as UC Davis or Windows 2000. In addition to defining ratio weights for CPU, memory, and disk use, you can also define weights for use by users. Figure 2.35 shows the **snmp_dca** monitor template.

```
monitor type snmp_dca {
    #type snmp_dca
    interval 10
    timeout 30
    dest *:161
    agent_type "UCD"
    cpu_coefficient "1.5"
    cpu_threshold "80"
    mem_coefficient "1.0"
    mem_threshold "70"
    disk_coefficient "2.0"
    disk_threshold "90"
}
```

Figure 2.35 *snmp_dca* monitor template

For detailed information on using the **snmp_dca** template, see *Configuring SNMP servers*, on page 2-17.

Using *snmp_dca_base*

Like the **snmp_dca** template, the **snmp_dca_base** template is for load balancing traffic to servers that are running an SNMP agent, such as UC Davis or Windows 2000. However, this template should be used only when you want the load balancing destination to be based solely on user data, and not CPU, memory, or disk use. Figure 2.36 shows the **snmp_dca_base** monitor template.

```
monitor type snmp_dca_base {
    #type snmp_dca_base
    interval 10
    timeout 30
    dest *:161
}
```

Figure 2.36 *snmp_dca_base* monitor template

For detailed information on using the **snmp_dca_base** template, see *Configuring SNMP servers*, on page 2-17.

Using nntp

The **nntp** template is for Usenet News. The check is successful if the monitor retrieves a newsgroup identification line from the server. An **nntp** monitor requires a **newsgroup** name (for example, "**alt.cars.mercedes**") and, if necessary, **username** and **password**.

```
monitor nntp {  
    #type nntp  
    interval 5  
    timeout 16  
    dest *:*  
    username ""  
    password ""  
    newsgroup ""  
}
```

Figure 2.37 The nntp monitor template

Using sql

The **sql** template is for service checks on SQL-based services such as Microsoft SQL Server versions 6.5 and 7.0, and also Sybase. The service checking is accomplished by performing an SQL login to the service. An executable program, **tdslogin** performs the actual login. The check is successful if the login succeeds.

An **sql** monitor requires a **database** (for example, "**server_db**"), **username**, and **password**.

```
monitor sql {  
    #type sql  
    interval 5  
    timeout 16  
    dest *:*  
    username ""  
    password ""  
    database ""  
}
```

Figure 2.38 The sql monitor template

Using imap

The **imap** template is for Internet Message Access Protocol. The **imap** monitor is essentially a **pop3** monitor with the addition of the attribute **folder**, which takes the optional key **message_num**. The check is successful if the specified message number is retrieved. An **imap** monitor requires **username**, **password**, and a **folder**. It also takes an optional message number, **message_num**.

```
monitor imap {
    #type imap
    interval 5
    timeout 16
    dest *.*
    username ""
    password ""
    folder ""
    /message_num ""
}
```

*Figure 2.39 The **imap** monitor template*

Using radius

The **radius** template is for Remote Access Dial-in User Service servers. The check is successful if the server authenticates the requesting user. A **radius** monitor requires a **username**, a **password**, and a shared secret string **secret** for the code number.

◆ **Note**

*Servers to be checked by a **radius** monitor typically require special configuration to maintain a high level of security while also allowing for monitor authentication.*

```
monitor radius {
    #type radius
    interval 5
    timeout 16
    dest *
    username ""
    password ""
    secret ""
}
```

*Figure 2.40 The **radius** monitor template*

Using ldap

The **ldap** template is for Lightweight Directory Access Protocol, which implements standard X.500 for e-mail directory consolidation. A check is successful if entries are returned for the **base** and **filter** specified. An **ldap** monitor requires a **username**, a **password**, and a **base** and a **filter** string. The **username** is a distinguished name, that is, an LDAP-format user name. The **base** is the starting place in the LDAP hierarchy from which to begin the query. The **filter** is an LDAP-format key of what is to be searched for.

Note

*Servers to be checked by an **imap** monitor typically require special configuration to maintain a high level of security while also allowing for monitor authentication.*

```
monitor ldap {
  #type ldap
  interval 5
  timeout 16
  dest *:*
  username ""
  password ""
  base ""
  filter ""
}
```

Figure 2.41 A Sample monitor template

Using external

The **external** template is for a user-supplied monitor. An **external** monitor requires the executable name (**run**) of that monitor and any command line arguments (**args**) required.

```
monitor external {
  #type external
  interval 5
  timeout 16
  dest *:*
  run ""
  args ""
}
```

Figure 2.42 The external monitor template

Configuring a monitor

The second step in creating a monitor and placing it in service is to configure the monitor from the monitor template. Configuring a monitor consists of giving it a name distinct from the monitor template name and assigning values to all attributes that are not to be left at their default values (and adding any optional attributes that are not present by default, like **reverse** or **transparent**). You can do this using the Configuration utility or at the command line using the **bigpipe monitor** command.

To configure a monitor using the Configuration utility

1. In the navigation pane, click **Monitors**.
The Network Monitors screen opens.
2. Click the **Add** button.
The Add Monitor screen opens.

3. In the Add Monitor screen, type in the name of your monitor (it must be different from the monitor template name), and select the monitor template you want to use.
4. Click the **Next** button and you are guided through the configuration of your monitor.
5. When you have finished configuring the monitor, click **Done**.

To configure a monitor from the command line

Use the **bigpipe monitor** command to configure the monitor at the command line. If you are defining the monitor with all attributes set to their default values, type:

```
b monitor <name> '{ use <template_name> }'
```

If you want to set one or more attributes to a new value, specify only those attributes and their values. For example, to create a **tcp_echo** monitor **my_tcp_echo** in **bigpipe** using the default values for the attributes **interval**, **timeout**, and **dest**, you would type:

```
b monitor my_tcp '{ use tcp_echo }'
```

If you are changing any of the default values, you need to specify only these changes. For example:

```
b monitor my_tcp-echo '{ use tcp_echo interval 10 timeout 20 }'
```

If you are using an optional attribute, such as **transparent**, add it to the list:

```
b monitor my_tcp-echo '{ use tcp_echo interval 10 timeout 20\
transparent dest 198.192.112.13:22}'
```

◆ Note

*If you are configuring an **snmp_dca** or **snmp_dca_base** monitor, see also **Configuring SNMP servers**, on page 2-17.*

Monitor attributes

Table 2.16 provides a summary of the monitor attributes and their definitions. For more information on the monitor templates and attributes, refer to *Selecting the monitor template*, on page 2-81.

Attribute	Definition
interval <seconds>	Ping frequency time interval in seconds.
timeout <seconds>	Ping timeout in seconds.
dest <node_addr>	Ping destination node. <node_address> Usually *:* for simple monitors, *:* for all others, causing the monitor instance to ping the address or address:port for which it is instantiated. Specifying address and/or port forces the destination to that address/port.
send <string>	Send string for ECV. Default send and recv values are empty (""), matching any string.
recv <string>	Receive expression for ECV. Default send and recv values are empty (""), matching any string.
get <string>	For the http and https monitors get replaces the recv statement, automatically filling in "GET" . For the ftp monitor get can be used to specify a full path to a file. This automatically fills in dest .
url	For the http and https , and ftp monitors, url replaces the recv statement, supplies a URL and automatically fills in dest with the URL address.
reverse	A mode that sets the node down if the received content matches the recv string.
transparent	A mode that forces pinging through the node to the dest address for transparent nodes, such as firewalls.
run <program>	An external user-added EAV program.
args <program_args>	List of command line arguments for external program. The args are quoted strings set apart by spaces.
username <username>	User name for services with password security. For ldap this is a distinguished , that is, LDAP-format user name.
password <password>	Password for services with password security.
newsgroup <newsgroup>	Newsgroup, for type nntp EAV checking only
database <database>	Database name, for type sql EAV checking only.
domain <domain_name>	Domain name, for type smtp EAV checking only
secret	Shared secret for radius EAV checking only.
folder	Folder name for imap EAV checking only.
message_num	Optional message number for imap EAV.
base	Starting place in the LDAP hierarchy from which to begin the query, for ldap EAV checking only.
filter	LDAP- format key of what is to be searched for, for ldap EAV checking only.

Table 2.16 Monitor attributes

Entering string values

You should enter all attribute values, except for **interval**, **timeout**, and **dest**, as quoted strings, even if they are numeric, as in the case of code numbers.

Setting destinations

By default, all **dest** values are set to the wildcard "*" or ".*.*". This causes the monitor instance created for a node to take that node's address or address and port as its destination. An explicit **dest** value is used only to force the instance destination to a specific address and/or port which may not be that of the node. For more information about setting destinations, refer to *Associating the monitor with a node or nodes*, on page 2-96.

Using send, receive, url, and get statements

The ECV monitor templates **http**, **https**, and **tcp** have the attributes **send** and **recv** for the send string and receive expression, respectively.

The most common send string is "GET /" which simply retrieves a default HTML page for a web site. To retrieve a specific page from a web site, simply enter a fully qualified path name:

```
"GET /www/support/customer_info_form.html"
```

The receive expression is the text string the monitor looks for in the returned resource. The most common receive expressions contain a text string that would be included in a particular HTML page on your site. The text string can be regular text, HTML tags, or image names.

The sample receive expression below searches for a standard HTML tag.

```
"<HEAD>"
```

You can also use the default null **recv** value "". In this case, any content retrieved is considered a match. If both **send** and **recv** are left empty, only a simple connection check is performed.

For **http** and **ftp**, the special attributes **get** or **url** may be used in place of **send** and **recv** statements. The attribute **get** takes the full path to the file as a value and automatically fills in the **dest** value with the address the path resolves to. The following two statements are equivalent:

```
send "GET/"  
get "/"
```

The attribute **url** takes the URL as a value and automatically fills in the **dest** value with the address the URL resolves to. The URL is then resolved to supply the **dest** address automatically. The third statement below is equivalent to the first two combined:

```
dest 198.192.112.13:22  
get "/"  
url "ftp://www.my_domain.com/"
```

Using transparent and reverse modes

The ECV monitors have optional keywords **transparent** and **reverse**. (The keyword **transparent** may also be used by **tcp_echo**.) The normal and default mode for a monitor is to ping the **dest** node by an unspecified route and to mark the node **up** if the test is successful. There are two other modes, transparent and reverse.

In transparent mode, the monitor is forced to ping *through* the node it is associated with, usually a firewall, to the **dest** node. (In other words, if there are two firewalls in a load balancing pool, the destination node will always be pinged through the one specified and not through the one picked by the load balancing method.) In this way, the transparent node is tested as well: if there is no response, the transparent node is marked **down**. For more information about transparent mode, refer to *Using transparent mode*, on page 2-99.

In reverse mode, the monitor marks the node **down** when the test is successful. For example, if the content on your web site home page is dynamic and changes frequently, you may want to set up a reverse ECV service check that looks for the string "**Error**". A match for this string would mean that the web server was down. Transparent mode can also be used with **tcp_echo**.

Transparent and reverse modes cannot be used on the same monitor.

Testing SQL service checks

SQL service checks may require manual testing before being implemented in a monitor, as follows:

```
cd /usr/local/lib/pingers
./tdslogin 192.168.1.1 1433 mydata user1 mypass1
```

Replace the IP address, port, database, user, and password in this example with your own information.

You should receive the message:

```
Login succeeded!
```

If you receive the connection refused message, verify that the IP and port are correct.

If you are still having trouble, you should verify that you can log in using another tool. For example, if you have Microsoft NT SQL Server version 6.5, there is a client program **ISQL/w** included with the SQL software. This client program performs simple logins to SQL servers. Use this program to test whether you can login using the ISQL/w program before attempting logins from the Link Controller.

On the SQL Server, you can run the SQL Enterprise Manager to add logins. When first entering the SQL Enterprise Manager, you may be prompted for the SQL server to manage.

You can register servers by entering the machine name, user name, and password. If these names are correct, the server will be registered and you will be able to click an icon for the server. When you expand the subtree for the server, there will be an icon for Logins.

Underneath this subtree, you can find the SQL logins. Here, you can change passwords or add new logins by right-clicking the Logins icon. Click this icon to open an option to **Add login**. After you open this option, type the user name and password for the new login, as well as which databases the login is allowed to access. You must grant the test account access to the database you specify in the EAV configuration.

Running user-added EAVs

You may add your own monitors to those contained in `/user/local/lib/pingers`. For running these added programs, the monitor template **external** is used. The executable program is specified as the value of the attribute **run**. By default, the monitor looks for the run program in `/user/local/lib/pingers`. If the program resides elsewhere, a fully qualified path name must be entered. Any command line arguments to be used with the program are entered as **args** values. For example, suppose the program **my_pinger** is to be run with a **-q** option, so that it would be entered on the command line as follows:

```
my_pinger -q
```

This monitor might be specified as follows:

```
b monitor custom '{ use external run "my_pinger" args "-q" }'
```

Alternatively, you may pass arguments to the external monitor as environment variables. For example, you might want to enter this command:

```
/var/my_pinger /www/test_files/first_test
```

This could be specified in the conventional manner:

```
b monitor custom '{ use external run "/var/my_pinger" args "www/test_files/first_test" }'
```

It could also be specified in this way:

```
b monitor custom '{ use external run "/var/my_pinger" DIRECTORY "www/test_files" FILE "first_test" }'
```

This defines the monitor as shown in Figure 2.43.

```
monitor custom {
  use external
  run "/var/my_pinger"
  DIRECTORY "www/test_files"
  FILE "first_test" }
```

Figure 2.43 Monitor template for an external monitor

This frees the monitor definition from the rigidity of a strictly ordered command line entry. The arguments are now order-independent and may be used or ignored by the external executable.

Showing, disabling, and deleting monitors

You can show, disable, and delete monitors using the Configuration utility or from the command line. Deleting a monitor removes it from the `/config/bigip.conf` file. Disabling a monitor instance simply removes that instance from service until it is re-enabled. Disabling a monitor (which can be performed only at the command line) disables all instances of the monitor. All monitor instances are enabled by default.

To show or delete a monitor using the Configuration utility

1. In the navigation pane, click **Monitors**.
A screen opens that lists monitors in two columns, System Supplied and User Defined.
2. To show a monitor, simply click the monitor name.
3. To delete a monitor, click the **Delete** button for the monitor. Note that only user-defined monitors can be deleted.

To show a monitor from the command line

You can display a selected monitor or all monitors using the **bigpipe monitor show** command:

```
b monitor <name> show
b monitor show all
```

To delete a monitor from the command line

You can delete a selected monitor using the **bigpipe monitor delete** command:

```
b monitor <name> delete
```

To disable a monitor instance using the Configuration utility

1. In the navigation pane, click **Monitors**.
The Monitors screen opens.
2. Click the appropriate tab for the monitor instances: Basic Associations, Node Associations, or Node Address Associations.
The resulting screen shows the existing associations (monitor instances).
3. Click the node you want to disable.
The Properties screen for that node opens.
4. In the Monitor Instances portion of the screen, clear the **Enable** check box.
5. Click **Apply**.
The monitor instance is now disabled.

To disable a monitor or monitor instance from the command line

To disable a monitor, use the **bigpipe monitor <name> disable** command:

b monitor <name> disable

This has the effect of disabling all instances of the monitor, as shown in Figure 2.44.

```
+ - NODE 11.12.11.20:80 UP
|
| +- http
|   11.12.11.20:80 up disabled
|
+ - NODE 11.12.11.21:80 UP
|
| +- http
|   11.12.11.21:80 up disabled
|
+ - NODE 11.12.11.22:80 UP
|
| +- http
|   11.12.11.22:80 ip disabled
```

Figure 2.44 All monitor instances disabled

To disable a monitor instance, use the **bigpipe monitor instance <addr:port> disable** command:

b monitor instance <addr:port> disable

Disabled monitors and instances may be re-enabled as follows:

b monitor <name> enable

b monitor instance <addr:port> enable

To delete a monitor with no node associations from the command line

You can delete a monitor if it has no existing node associations and no references in a monitor rule. To delete a monitor, use the **bigpipe monitor <name> delete** command:

b monitor my_http delete

If the monitor has instances, the instances must first be deleted using the **bigpipe node <addr:port> monitor delete** command. (Refer to *Showing and deleting associations*, on page 2-101.)

Associating the monitor with a node or nodes

Now that your monitor exists, the final step is to associate it with the nodes to be monitored. This creates an instance of the monitor for each node. At the command line, association is done using **bigpipe node** command:

b node <addr_list> monitor use <name>

For example, to associate monitor **http** with nodes **11.12.11.20:80**, **11.12.11.21:80**, and **11.12.11.22:80**, use the following **bigpipe node** command:

```
b node 11.12.11.20:80 11.12.11.21:80 11.12.11.22:80 monitor use\
http
```

This creates a monitor instance of **http** for each of these nodes. You can verify this association using the **bigpipe monitor show** command:

```
b node monitor show
```

This would produce the output shown in Figure 2.45.

```
+ - NODE  11.12.11.20:80  UP
|         |
|         +- http
|         11.12.11.20:80 up enabled
|
+ - NODE  11.12.11.21:80  UP
|         |
|         +- http
|         11.12.11.21:80 up enabled
|
+ - NODE  11.12.11.22:80  UP
|         |
|         +- http
|         11.12.11.22:80 ip enabled
```

*Figure 2.45 The output of the **b node monitor show** command*

The actual monitor instance for each node is represented by the output lines highlighted with bold text in Figure 2.45.

Reviewing types of association

While the term ***node association*** is applied generally, there are three types of association based on whether the monitor is associated with an address and a port, address(es) only, or port only. These are node association, address association, and port association.

- ***Node association***, strictly defined, is the association of a monitor with an address and port.
- ***Address association*** is the association of a monitor with an address only.
- ***Port association*** is the association of a monitor with a port only. For a port association, a wildcard character (*) is used to represent all addresses.

Once a monitor has been associated with a node, address, or port, no other monitor can be associated with the same node, address or port. However, an address association does not prevent a monitor from being associated with a node of the same address, or the reverse.

Using a simple association

The **http** example given above is the simplest kind of association, a node association performed using a monitor with a **dest** value of ***.***. It can be seen in Figure 2.30, on page 2-83, that in each case the instance destination node is identical to the node the monitor has been associated with. This is because the template **http**, shown in Figure 2.30, was used as is, with a **dest** value of ***.***. Either or both wildcard symbols can be replaced by an explicit **dest** value by creating a new monitor based on **http**. This is referred to as **node** and **port aliasing**, described in the following section.

Using node and port aliasing

Usually the health of a node is checked by pinging that node. For this reason the **dest** attribute in the monitor template is always set to **"*"** or **"*.*"**. This causes the monitor instance created for a node to take that node's address or address and port as its destination. An explicit **dest** value forces the instance destination to a specific address and/or port which may not be that of the node. This causes the monitor to ping that forced destination by an unspecified path. Suppose, for example, that the association performed using **http** instead used a monitor **my_http** with a **dest** value of ***:443**. The node association command would be identical except that **http** is now replaced with **my_http**:

```
b node 11.12.11.20:80 11.12.11.21:80 11.12.11.22:80 monitor use my_http
```

This creates three instances of the monitor with the following **dest** values as shown in Figure 2.46.

```
+ - NODE  11.12.11.20:80  UP
|
|   +- my_http
|      11.12.11.20:443 up enabled
|
+ - NODE  11.12.11.21:80  UP
|
|   +- my_http
|      11.12.11.21:443 up enabled
|
+ - NODE  11.12.11.22:80  UP
|
|   +- my_http
|      11.12.11.22:443 up enabled
```

Figure 2.46 Node ports aliased

This is referred to as **port aliasing**. The node itself can also be aliased, by assigning an explicit address to **dest**. For example, **dest** could set to **11.11.11.1:80**. This is called **node aliasing** and for the nodes **11.12.11.20:80**, **11.12.11.21:80**, and **11.12.11.22:80** it would produce the following instances, (which are in fact one instance associated with three different nodes) as shown in Figure 2.47.

```

+- NODE 11.12.11.20:80 ADDR UP
|
| +- my_http
|   11.11.11.1:80 checking enabled
|
+- NODE 11.12.11.21:80 ADDR UP
|
| +- my_http
|   11.11.11.1:80 checking enabled
|
+- NODE 11.12.11.22:80 ADDR UP
|
| +- my_http
|   11.11.11.1:80 checking enabled

```

Figure 2.47 Node addresses aliased

Using transparent mode

Sometimes it is necessary to ping the aliased destination through a transparent node. Common examples are checking a router, or checking a mail or FTP server through a firewall. For example, you might want to check the router address **10.10.10.53:80** through a transparent firewall **10.10.10.101:80**. To do this, you would specify **10.10.10.53:80** as the monitor **dest** address (a node alias) and add the flag **transparent**:

```
b monitor http_trans '{ use http dest 10.10.10.53:80 transparent }'
```

Then you would associate the monitor **http_trans** with the transparent node:

```
b node 10.10.10.101:80 monitor use http_trans
```

This causes the address **10.10.10.53:80** to be checked through **10.10.10.101:80**. (In other words, the check of **10.10.10.53:80** is routed through **10.10.10.101:80**.) If the correct response is not received from **10.10.10.53:80**, then **10.10.10.101:80** is marked **down**.

◆ Note

*Transparent mode applies only to the ECV monitors and to **tcp_echo**.*

Using logical grouping

In the preceding examples, only one monitor has been associated with the nodes. You may associate more than one monitor with a node or nodes by joining them with the Boolean operator **and**. This creates a rule, and the node is marked as **down** if the rule evaluates to false, that is, if not all the checks are successful. The most common example is the use of an HTTP monitor and an HTTPS monitor:

```
b node 11.12.11.20:80 monitor use my_http and my_https
```

The monitors themselves must be configured with the grouping in mind. For example, if the **dest** values of both monitors were set to ***:***, then both monitor instances would try to ping the default port **80**. This would both

defeat the purpose of the HTTPS monitor and cause an automatic failure, since two monitors would be trying to ping the same address and port simultaneously.

Instead, monitor **my_http** should be given a **dest** value of ***:*** and monitor **my_https** should be given a **dest** value of ***:443**. This causes only the **my_http** monitor instances to default to **80**. The **my_https** monitor instances are forced to the explicit port **443**, avoiding a conflict as shown in Figure 2.48.

```
MONITOR my_http and https_443
|
+- NODE 11.12.11.20:80 UP
|
|      +- my_http
|      | 11.12.11.20:80 up enabled
|      |
|      +- https_443
|      | 11.12.11.20:443 up enabled
```

Figure 2.48 Use of a monitor rule

Using wildcards to specify addresses

The wildcard ***** can be used to specify addresses. A wildcard address association creates instances of the monitor for all nodes load balanced by the Link Controller:

```
b node * monitor use my_tcp_echo:
```

A wildcard with a port association creates instances of the monitor for every node configured to service that port:

```
b node *:80 monitor use my_http:
```

To associate a monitor using the Configuration utility

1. In the navigation pane, click **Monitors**.
The Network Monitors screen opens.
2. Click one of three tabs:
  - If you are associating the monitor with a *node* (the IP address plus the port), click the Node Associations tab.
  - If you are associating the monitor with a *node address* only (the IP address minus the port), click the Node Address Associations tab.
  - If you are associating the monitor with a port only (the port minus the IP address), click the Port Associations tab.
3. Regardless of the selection you made in step 2, a dialog box opens with the boxes **Choose Monitor** and **Monitor Rule**. Type the monitor name or select one from the list.
4. If you want to associate more than one monitor, click the Move button (>>) to add the monitor name to the **Monitor Rule** box.

5. Repeat the previous two steps for each monitor you want to associate with a node.
6. Click **Apply** to associate the monitor(s).
For additional information associating a monitor, click the **Help** button.

Showing and deleting associations

There are node commands for showing, and deleting node associations.

To show or delete associations using the Configuration utility

1. In the navigation pane, click **Monitors**.
The Network Monitors screen opens.
2. Click one of three tabs:
  - If you are showing or deleting a *node* association (a node is the IP address plus the port) click the Node Associations tab.
  - If you are showing or deleting a *node address* association (the IP address minus the port), click the Node Address Associations tab.
  - If you are showing or deleting a *port* association (the port minus the IP address), click the Port Associations tab.
3. Regardless of the selection you made in step 2, a dialog box opens showing existing associations, and with a **Delete Existing Associations** check box.
4. Delete associations by checking the box, then clicking **Apply**.

◆ Note

For wildcard address associations, the wildcard (*) association itself is shown in addition to each of the individual associations it produces. To delete all these associations, it is necessary to delete the wildcard association.

To show associations from the command line

You can display a selected node association or all node associations using the **bigpipe node monitor show** command:

```
b node monitor show
```

```
b node <addr:port> monitor show
```

To delete associations from the command line

You can delete a selected node association or all node associations using the **bigpipe node monitor delete** command:

```
b node <addr:port> monitor delete
```

In deleting specific monitor instances, it is important to consider how the association was made. If a monitor instance was created using a wildcard address, the wildcard must be deleted. For example, if multiple associations were created by entering **b node *:80 monitor use my_tcp_echo**, you would delete it by typing:

```
b node *:80 monitor delete
```

If multiple associations were created by entering **b node * monitor use my_tcp_echo**, you delete it by typing:

```
b node * monitor delete
```




3

The big3d Agent

- Working with the big3d agent
- Overview of communications protocols

Working with the big3d agent

The **big3d** agent collects performance information on behalf of the Link Controller. Before the **big3d** agents can communicate with the Link Controllers in the network, you need to configure the appropriate ports and tools to allow communication between the devices running the **big3d** agent and Link Controllers in the network. These planning issues are discussed in *Communications for remote administration of the Link Controller*, on page 3-3.

Collecting path data and server performance metrics

A **big3d** agent collects the following types of performance information used for load balancing.

- ◆ **Virtual server availability**
The **big3d** agent queries virtual servers to verify whether they are up and available to receive connections. For name resolution, the Link Controller uses only those virtual servers that are **up**.
- ◆ **Network path round trip time**
The **big3d** agent calculates the round trip time for the network path between the Link Controller and the client's LDNS server that is making the resolution request. The round trip time is used to determine the best virtual server to answer the request when you use the Round Trip Times or the Quality of Service load balancing modes.
- ◆ **Network path packet loss**
The **big3d** agent calculates the packet completion percentage for the network path between the data center and the client's LDNS server that is making the resolution request. Packet completion is used to determine the best virtual server to answer the request when you use the Completion Rate or the Quality of Service load balancing modes.
- ◆ **Router hops along the network path**
The **big3d** agent calculates the number of intermediate systems transitions (router hops) between the data center and the client's LDNS server. Hops are used to determine the best virtual server to answer the request when you use the Hops or the Quality of Service load balancing modes.
- ◆ **Server performance**
The **big3d** agent calculates server metrics, such as the packet rate for Link Controller virtual servers. Packet rate is used to determine the best virtual server to answer the request when you use the Packet Rate or the Quality of Service load balancing modes.
- ◆ **Virtual server performance**
The **big3d** agent calculates the number of connections to virtual servers defined on the Link Controller. The number of virtual server connections is used to determine the best virtual server when using the Least Connections load balancing mode.

Understanding factories run by big3d agents

To gather performance information, the **big3d** agent uses different types of factories. A *factory* is a process that collects different types of data. The **big3d** agent currently supports the following factory types:

◆ **Prober factory**

A prober factory collects several types of information using ICMP, TCP, UDP, DNS_DOT, or DNS_REV protocols. This factory queries local DNS servers. For local DNS servers, the prober factory uses the response time to calculate the round trip time and packet loss between the LDNS and the Link Controller.

◆ **Hops factory**

A hops factory uses the **tracert** utility to calculate the number of intermediate systems transitions (or router hops) along the network path between the Link Controller and a client LDNS.

◆ **SNMP factory**

An SNMP factory queries the SNMP agents that run on routers to collect performance metrics for the router.

◆ **Note**

You cannot configure the factories on a Link Controller.

Tracking LDNS probe states with the big3d agent

The Link Controller tracks the state of path data collection for each LDNS that has ever requested a name resolution from the system. Table 3.1 shows the states that can be assigned to an LDNS. Note that you can view the state of LDNS servers in the Local DNS Statistics screen in the Configuration utility.

State	Description
Needs Probe	The big3d agent has never collected data for the LDNS, or the data has expired.
Idle	The big3d agent successfully collected data for the LDNS, and is waiting for the next collection request.
In Probe	The big3d agent is currently collecting data for the LDNS.

Table 3.1 Probe and discovery states for individual client LDNS servers

Overview of communications protocols

The Link Controller supports the following communications protocols.

Communications for remote administration of the Link Controller

The Configuration utility is a browser-based utility through which you can administer the Link Controller from a remote workstation. Table 3.2 shows the port that is used for remote administrative connections to the Link Controller web server, which hosts the Configuration utility.

From	To	Protocol	Port	Purpose
Configuration utility on a remote workstation	Link Controller	https over TCP	443	Connection to secure web server

Table 3.2 Communications between Link Controllers and remote workstations

Communications for resolving DNS requests

Table 3.3 shows the ports on which the Link Controller receives and responds to DNS resolution requests issued by local DNS servers for inbound load balancing.

From	To	Protocol	From Port	To Port	Purpose
LDNS	Link Controller	UDP	53 or >1024	53	DNS resolution requests
Link Controller	LDNS	UDP	53	53 or >1024	DNS resolution responses

Table 3.3 DNS communications on the Link Controller

Communications for collecting path data

Table 3.4 shows the protocols and ports that the **big3d** agent uses when collecting path data for local DNS servers. The Link Controller uses path data for dynamic load balancing modes.

From	To	Protocol	From Port	To Port	Purpose
big3d	LDNS	ICMP	N/A	N/A	Probe using ICMP pings
big3d	LDNS	TCP	>1023	53	Probe using TCP (Cisco routers: allow establish)
LDNS	big3d	TCP	53	>1023	Replies using TCP (Cisco routers: allow establish)

Table 3.4 Communications between **big3d** agents and local DNS servers

From	To	Protocol	From Port	To Port	Purpose
big3d	LDNS	UDP	53	33434	Probe using UDP or traceroute utility
LDNS	big3d	ICMP	N/A	N/A	Replies to ICMP, UDP pings, or traceroute probes
big3d	LDNS	dns_rev dns_dot	>1023	53	Probe using DNS version or DNS dot
LDNS	big3d	dns_rev dns_dot	53	>1023	Replies to DNS version or DNS dot probes

Table 3.4 Communications between **big3d** agents and local DNS servers



4

Internet Link Evaluator

- Overview of the Internet Link Evaluator
- Interpreting the Internet Link Evaluator data

Overview of the Internet Link Evaluator

The Internet Link Evaluator, a statistics screen in the Configuration utility, is a unique tool with which to evaluate the following data about the Internet:

- The average round trip time between the local DNS servers on a particular continent and the links in your network
- The average completion rate between the local DNS servers on a particular continent and the links in your network
- The average number of router hops between the local DNS servers on a particular continent and the links in your network

The data displayed in the Internet Link Evaluator is based on path data, which is collected when you use a dynamic load balancing mode, such as Round Trip Times or Quality of Service. For more information on dynamic load balancing modes, see *Using dynamic load balancing modes*, on page 5-4.

To view the Internet Link Evaluator statistics screen using the Configuration utility

1. In the navigation pane, expand the **Link Statistics** item.
2. Click **Link Evaluator**.
The Internet Link Evaluator Statistics screen opens.
3. For information on working with the Internet Link Evaluator Statistics screen, view the online help.

The round trip time and completion rate data, on the Internet Link Evaluator Statistics screen, are based on path data. The router hops data is based on information collected by the **traceroute** utility.

Working with the Average Round Trip Time table

In the Average Round Trip Time table on the Internet Link Evaluator Statistics screen, you can view the following information:

- The average round trip time for each link to each continent
- For each link, the best average round trip time to the local DNS servers on a particular continent. This value is indicated by **bold** text within the table.
- For each continent, the best average round trip time from the links. This value is indicated by underlined text within the table.

If you hold the mouse pointer over the Information button (), you can view the following additional information:

- For a particular link, the number of local DNS servers used to calculate the average round trip time

- For all the local DNS servers that have been probed by a particular link, the percentage of those local DNS servers that are located on a particular continent
- For all the local DNS servers on a particular continent, the percentage of those local DNS servers that have been probed by a particular link

Working with the Average Completion Rate table

In the Average Completion Rate table on the Internet Link Evaluator Statistics screen, you can view the following information:

- The average completion rate for each link to each continent
- For each link, the best average completion rate to the local DNS servers on a particular continent. This value is indicated by **bold** text within the table.
- For each continent, the best average completion rate from the links. This value is indicated by underlined text within the table.

If you hold the mouse pointer over the Information button (), you can view the following additional information:

- For a particular link, the number of local DNS servers used to calculate the average completion rate
- For all the local DNS servers that have been probed by a particular link, the percentage of those local DNS servers that are located on a particular continent
- For all the local DNS servers on a particular continent, the percentage of those local DNS servers that have been probed by a particular link

Working with the Average Router Hops table

In the Average Router Hops table on the Internet Link Evaluator Statistics screen, you can view the following information:

- The average number of router hops between each link and each continent
- For each link, the best average number of router hops to the local DNS servers on a particular continent. This value is indicated by **bold** text within the table.
- For each continent, the best average number of router hops from the links. This value is indicated by underlined text within the table.

If you hold the mouse pointer over the Information button (), you can view the following additional information:

- For a particular link, the number of local DNS servers used to calculate the average number of router hops
- For all the local DNS servers that have been probed by a particular link, the percentage of those local DNS servers that are located on a particular continent

- For all the local DNS servers on a particular continent, the percentage of those local DNS servers that have been probed by a particular link

Interpreting the Internet Link Evaluator data

In choosing an ISP provider, enterprises have very few ways to evaluate the quality of the connectivity provided. By measuring the round trip times, completion rate, and average router hops over the configured links on the Link Controller, the Internet Link Evaluator provides a thorough view of link and ISP performance. By comparing link performance over time, you can monitor the actual performance that an ISP delivers to your customers.

The two data points that help you determine which link has the best performance are the RTT response time (lower is better), and the Completion Rate (higher is better). The Round Trip Time measurement calculates the packet speed of a request from each geographically-distributed user. The Completion Rate measurement is an important metric for evaluating the quality of a link, because it represents the percentage of dropped packets. Completion rate is also important for measuring line quality for applications such as streaming media.

◆ Note

One easy way to compare link performance over time is to print a screen shot of the Internet Link Evaluator at a certain time every day.



5

Inbound Load Balancing

- Working with load balancing modes for inbound traffic
- Understanding inbound load balancing on the Link Controller
- Configuring inbound load balancing
- Setting up load balancing for services that require multiple ports

Working with load balancing modes for inbound traffic

The Link Controller uses load balancing modes to distribute DNS name resolution requests, sent by local DNS servers, to the best available link and virtual server in the Link Controller configuration. This is known as ***inbound load balancing***. This chapter first describes how inbound load balancing works on the Link Controller, explains the various static and dynamic load balancing modes, and then describes how to configure them.

Understanding inbound load balancing on the Link Controller

When the Link Controller receives a name resolution request from a local DNS server, the system uses a load balancing mode to select the best available link by evaluating the virtual servers in a wide IP pool. Once the Link Controller selects the best virtual server, it constructs the DNS answer and sends the answer back to the requesting client's local DNS server, using the best available link. The DNS answer, or ***resource record***, is an **A** record that contains one or more virtual server IP addresses.

The Link Controller chooses a virtual server from a wide IP pool using either a static load balancing mode or a dynamic load balancing mode. A ***static load balancing mode*** selects a virtual server based on a pre-defined pattern. A ***dynamic load balancing mode*** selects a virtual server based on current performance metrics.

Inbound load balancing load balances the virtual servers within a pool. Within each pool, you specify two different load balancing modes that the system uses in sequential order: the preferred method or the alternate method. The ***preferred*** method is the first load balancing mode that the Link Controller uses for load balancing. If the preferred method fails, the system then uses the alternate method for load balancing. If this load balancing mode fails, then the Link Controller randomly picks an available virtual server.

The Link Controller supports several modes for inbound load balancing. Table 5.1 shows a complete list of the modes and indicates where you can use each mode. The sections following the table describe how each load balancing mode works.

Load Balancing mode	Use for preferred method	Use for alternate method	Static mode	Dynamic mode
Completion Rate	X			X
Global Availability	X	X	X	
Hops	X			X
Kilobytes/Second	X			X
Least Connections	X		X	
Packet Rate	X	X		X
Quality of Service	X			X
Random	X	X	X	
Ratio	X	X	X	
Round Robin	X	X	X	
Round Trip Time	X			X
Static Persist	X	X	X	
VS Capacity	X	X		X

Table 5.1 Inbound load balancing modes and usage

Static load balancing modes distribute connections across the network according to predefined patterns, and take server availability into account. **Dynamic load balancing modes** distribute connections to servers that show the best current performance. The performance metrics taken into account depend on the particular dynamic mode you are using.

Using static load balancing modes

The Link Controller supports the following static load balancing modes:

- Global Availability
- Random
- Ratio
- Round Robin
- Static Persist

The static load balancing modes perform load balancing as described in the following sections.

Global Availability mode

The Global Availability load balancing mode uses the virtual servers included in the pool in the order in which they are listed. For each connection request, this mode starts at the top of the list and sends the connection to the first available virtual server in the list. Only when the current virtual server is full or otherwise unavailable does Global Availability mode move to the next virtual server in the list. Over time, the first virtual server in the list receives the most connections and the last virtual server in the list receives the least number of connections.

Random mode

The Random load balancing mode sends connections to virtual servers in a random, uniform distribution pattern. The Random mode is useful for certain test configurations.

Ratio mode

The Ratio load balancing mode distributes connections among a pool of virtual servers as a weighted Round Robin. For example, you can configure the Ratio mode to send twice as many connections to a T1 link, and only half as many connections to a DSL link.

The Ratio load balancing mode requires that you define a ratio weight for each virtual server in a pool. The default ratio weight for a server or a pool is set to **1**. Figure 5.1 shows a sample connection distribution for Ratio mode.

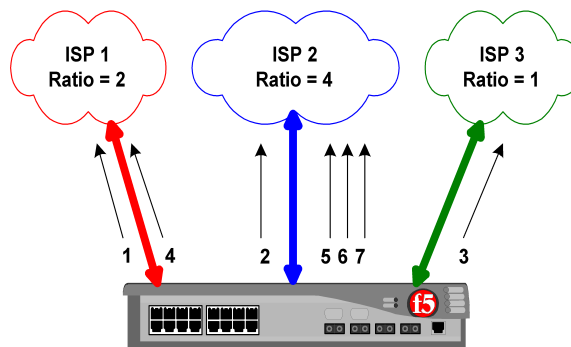


Figure 5.1 Ratio mode

Round Robin mode

The Round Robin load balancing mode distributes connections in a circular and sequential pattern among the virtual servers in a pool. Over time, each virtual server receives an equal number of connections.

Figure 5.2 shows a sample of the connection distribution pattern for Round Robin mode.

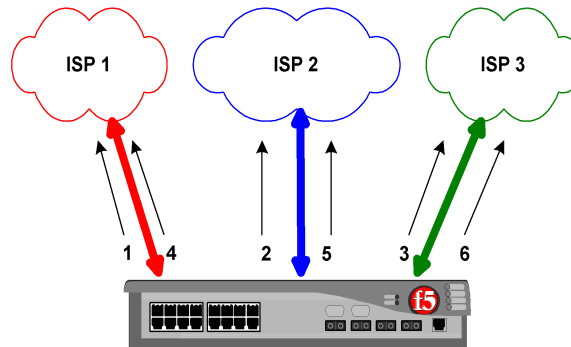


Figure 5.2 Round Robin mode

Static Persist mode

The Static Persist load balancing mode provides static persistence of local DNS servers to virtual servers; it consistently maps an LDNS IP address to the same available virtual server for the duration of the session. This mode guarantees that certain transactions are routed through a single transaction manager (for example, a BIG-IP or other server array manager); this is beneficial for transaction-oriented traffic, such as e-commerce shopping carts, online trading, and online banking.

Using dynamic load balancing modes

All dynamic load balancing modes make load balancing decisions based on the metrics collected by the **big3d** agents running in each data center. The **big3d** agents collect the information at set intervals that you define when you set the global timer variables. If you want to use the dynamic load balancing modes, you must run one or more **big3d** agents in each of your data centers, to collect the required metrics.

The Link Controller supports the following dynamic load balancing modes:

- Completion Rate
- Hops
- Kilobytes/Second
- Least Connections
- Packet Rate
- Round Trip Times (RTT)
- Quality of Service (QOS)
- VS Capacity

Completion Rate mode

The Completion Rate load balancing mode selects the virtual server that currently maintains the least number of dropped or timed-out packets during a transaction between a Link Controller and the client LDNS.

Figure 5.3 shows a sample connection distribution pattern for the Completion Rate mode.

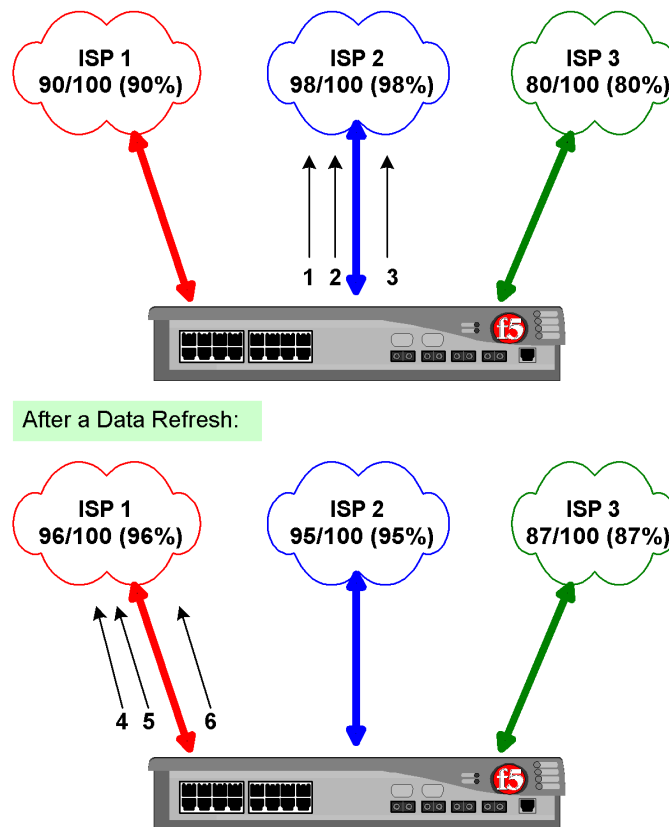


Figure 5.3 Completion Rate load balancing mode

Hops mode

The Hops load balancing mode is based on the **tracert** utility, and tracks the number of intermediate system transitions (router hops) between a client LDNS and each Link Controller. Hops mode selects a virtual server in the data center that has the fewest router hops from the LDNS.

Kilobyte/Second mode

The Kilobytes/Second load balancing mode selects a virtual server that is currently processing the fewest number of kilobytes per second.

Least Connections mode

The Least Connections load balancing mode selects a virtual server on the Link Controller that currently hosts the fewest connections.

Packet Rate mode

The Packet Rate load balancing mode selects a virtual server that is currently processing the fewest number of packets per second.

Figure 5.4 shows a sample connection distribution for the Packet Rate mode.

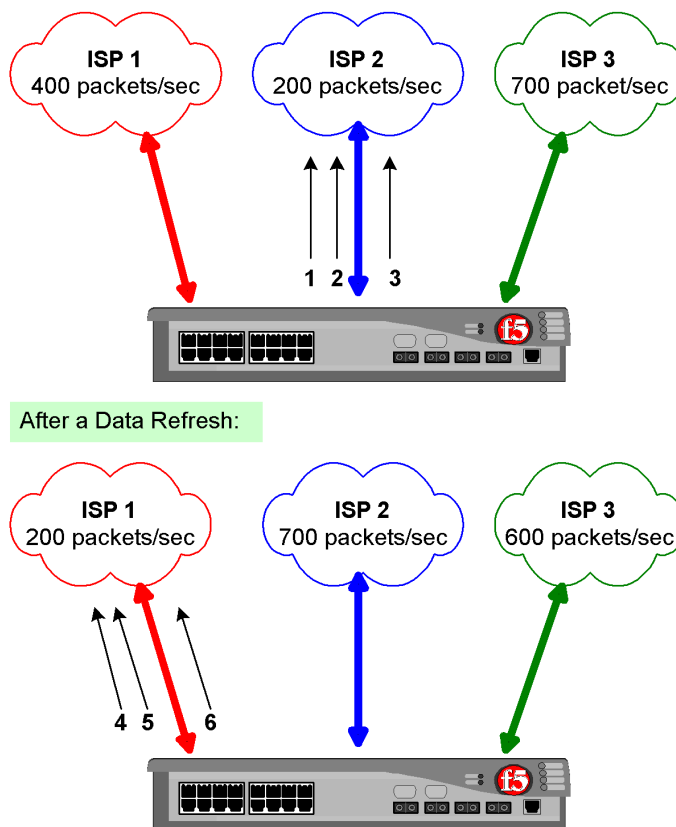


Figure 5.4 Packet Rate mode

Round Trip Times mode

The Round Trip Times (RTT) load balancing mode selects the virtual server with the fastest measured round trip time between a Link Controller and a client LDNS.

Figure 5.5 shows a sample connection distribution for the Round Trip Times mode.

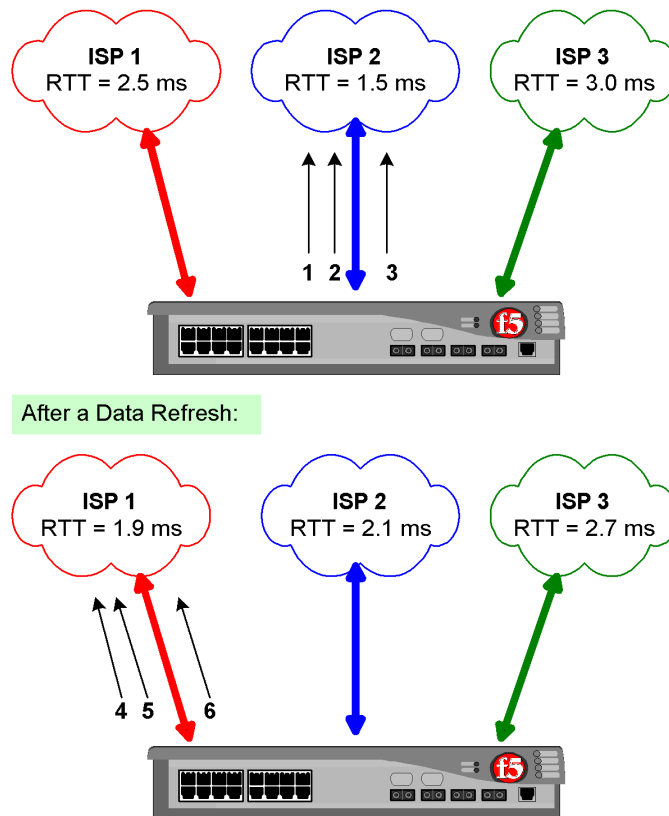


Figure 5.5 Round Trip Times mode

Quality of Service mode

The Quality of Service load balancing mode uses current performance information to calculate an overall score for each virtual server, and then distributes connections based on each virtual server's score. The performance factors that the Link Controller takes into account include:

- Round trip time
- Hops
- Completion rate
- Packet rate
- Link Capacity
- VS Capacity
- Kilobytes/Second

The Quality of Service load balancing mode is a customizable load balancing mode. For simple configurations, you can easily use this load balancing mode with its default settings. For more advanced configurations, you can specify different weights for each performance factor in the equation.

You can also configure the Quality of Service load balancing mode to use the dynamic ratio feature. With the dynamic ratio feature turned on, the Quality of Service mode becomes similar to the Ratio mode, where the connections are distributed in proportion to ratio weights assigned to each virtual server. The ratio weights are based on the QOS scores: the better the score, the higher the percentage of connections the virtual server receives.

Understanding QOS coefficients

Table 5.2 lists each Quality of Service (QOS) coefficient, its scale, a likely upper limit for each, and whether a higher or lower value is more efficient.

Coefficient	How measured	Default value	Example upper limit	Higher or lower?
Round trip time	Microseconds	50	2,000,000	Lower
Completion rate	Percentage of successfully transferred packets (0-100%)	5	100%	Higher
Hops	Number of intermediate systems transitions (hops)	0	64	Lower
Packet rate	Packets per second	1	700	Lower
Link Capacity	Link bandwidth and cost analysis	30	2,000,000	Lower
VS capacity	Number of nodes <i>up</i>	0	20	Higher
Kilobytes/second	Kilobytes per second throughput	3	15000	Lower

Table 5.2 QOS coefficients: default values, ranges, and limits

If you change the default QOS coefficients, keep the following issues in mind.

- ◆ **Scale**

The raw metrics for each coefficient are not on the same scale. For example, completion rate is measured in percentages, while packet rate is measured in packets per second.

- ◆ **Normalization**

The Link Controller normalizes the raw metrics to values in the range of 0 to 10. As the QOS value is calculated, a high measurement for completion rate is good, because a high percentage of completed

connections are being made, but a high value for packet rate is not desirable because the packet rate load balancing mode attempts to find a virtual server that is not overly taxed at the moment.

- **Emphasis**

You can adjust coefficients to emphasize one normalized metric over another. For example, by changing the coefficients to the values shown in Figure 5.6, you are putting the most emphasis on link capacity.

Modify Virtual Servers   Modify Alias   Port List   List Pools   List Wide IPs   Help

Modify Load Balancing for Pool

external.wip.net

Enter settings and click "Update"

Edit Pool	
Load Balancing Modes	Preferred: Quality Of Service
	Alternate: Round Robin
Use Dynamic Ratio	☐
Use Round Robin LDNS	☐
Round Robin LDNS Limit	0
Check Static Depends	☒
Check Dynamic Depends	☒
Quality Of Service	Round Trip Time: 50
	Completion Rate: 5
	Hops: 0
	Packet Rate: 1
	Link Capacity: 100
	VS Capacity: 0
	Kilobytes/Second: 3
Update Reset	

Figure 5.6 QOS coefficients emphasizing Link Capacity

In Figure 5.6, if the link capacity scores for two virtual servers are not close, the virtual server with the most available capacity is chosen. If the link capacity scores for both virtual servers are close, the round trip time (RTT) breaks the tie. In this example, the metrics for Hops, VS Capacity, and Kilobytes/Second modes are not used in determining how to distribute connections.

Customizing the QOS equation

You can customize the QOS equation for individual wide IPs to tailor Quality of Service load balancing to best match your load balancing requirements.

To modify QOS coefficients using the Configuration utility

1. In the navigation pane, click **Wide IPs**.
2. In the Wide IP column, click a wide IP name.
The Modify Wide IP screen opens.
3. On the toolbar, click **Modify Pool**.
The Modify Wide IP Pools screen opens.
4. In the Pool Name column, click **Pool**.
The Modify Load Balancing for Pool screen opens.
5. Define the QOS coefficients in the **Round Trip Time**, **Completion Rate**, **Hops**, **BIG-IP Packet Rate**, **Link Capacity**, **VS Capacity**, and **Kilobytes/Second** boxes.
6. Click **Update**.

Using the Dynamic Ratio option

When the Dynamic Ratio option is turned on, the Link Controller treats QOS scores as ratios, and it uses each server in proportion to the ratio determined by the QOS calculation. When the Dynamic Ratio option is turned off (the default), the Link Controller uses only the server with the highest QOS score for load balancing, (in which case it is a winner takes all situation) until the metrics information is refreshed.

To turn on the Dynamic Ratio option using the Configuration utility

1. In the navigation pane, click **Wide IPs**.
2. In the Wide IP column, click a wide IP name.
The Modify Wide IP screen opens.
3. On the toolbar, click **Modify Pool**.
The Modify Wide IP Pools screen opens.
4. In the Pool Name column, click **Pool**.
The Modify Load Balancing for Pool screen opens.
5. Check the **Use Dynamic Ratio** box.
6. Click **Update**.

◆ Note

We recommend that you turn on dynamic ratio, so that the Link Controller uses any bandwidth cost information you have configured as part of the Link Capacity coefficient.

VS Capacity mode

The VS Capacity load balancing mode creates a list of the virtual servers, weighted by capacity, then picks one of the virtual servers from the list. The virtual servers with the greatest capacity are picked most often, but over time all virtual servers are returned. If more than one virtual server has the same capacity, then the Link Controller load balances using the Random mode among those virtual servers.

◆ Note

We recommend that you do not use the VS Capacity load balancing mode on the Link Controller because it manages only its own virtual servers.

Configuring inbound load balancing

This section describes how to configure inbound load balancing on the Link Controller. You configure inbound load balancing at the wide IP level.

When you define a wide IP, you specify the preferred and alternate load balancing methods to use in selecting a virtual server within the wide IP pool. You can find instructions on how to configure these load balancing methods in the section, *Defining a wide IP*, on page 5-12.

Understanding wide IPs

After you configure the virtual servers they manage, you need to group the configured virtual servers into wide IPs. A **wide IP** is a mapping of a fully-qualified domain name (FQDN) to a set of virtual servers that host the domain's content, such as a web site, an e-commerce site, or a CDN.

Before defining the first wide IP, you should do the following:

- Gather your configuration information for the Link Controller so you can easily see which virtual servers have the content you want to map to an FQDN.
- Decide which load balancing modes to use for the wide IP pool.

There may be situations (for example, e-commerce, and other sites with multiple services) where you need to configure a wide IP so that connections are not sent to a given address unless multiple ports or services are available. You configure this behavior after you define the wide IP. For more information, see *Setting up load balancing for services that require multiple ports*, on page 5-13.

Understanding wide IP pools

A wide IP contains a pool definition. A *pool* is a group of virtual servers that the Link Controller load balances. The virtual servers on a Link Controller are associated with a specific link, and they load balance inbound requests over the links.

Defining a wide IP

After you determine which virtual servers you should place in which wide IP pools, you are ready to add the first wide IP to the configuration. Note that you configure only one pool in the wide IP.

To define a wide IP using the Configuration utility

1. In the navigation pane, click **Wide IPs**.
The Wide IP List screen opens.
2. On the toolbar, click **Add Wide IP**.
The Add a New Wide IP screen opens.
3. Add the wide IP settings, and click **Next**.
The Configure Load Balancing for New Pool screen opens.
4. Add the pool settings, and click **Next**.
The Select Virtual Servers screen opens.
5. Check the virtual servers that you want to add to the pool, and click **Finish**.
The wide IP is added to your configuration.

Repeat this process for each wide IP you want to add. For help on defining wide IPs and pools, click **Help** on the toolbar.

Using wildcard characters in wide IP names

The Link Controller supports wildcard characters in both wide IP names and wide IP aliases. You can use the wildcard characters to simplify your maintenance tasks if you have a large quantity of wide IP names and/or wide IP aliases. The wildcard characters you can use are: the question mark (?), and the asterisk (*). The guidelines for using the wildcard characters are as follows:

◆ The question mark (?)

- You can use the question mark to replace a single character, with the exception of dots (.).
- You can use more than one question mark in a wide IP name or alias.
- You can use both the question mark and the asterisk in the same wide IP name or alias.

◆ **The asterisk (*)**

- You can use the asterisk to replace multiple consecutive characters, with the exception of dots (.).
- You can use more than one asterisk in a wide IP name or alias.
- You can use both the question mark and the asterisk in the same wide IP name or alias.

The following examples are all valid uses of the wildcard characters for the wide IP name, **www.mydomain.net**.

- **???.mydomain.net**
- **www.??domain.net**
- **www.my*.net**
- **www.??*.net**
- **www.my*.***
- **???.my*.***
- ***.*.net**
- **www.*.???**

Using the LDNS round robin wide IP attribute

LDNS round robin is an attribute that you can use in conjunction with any load balancing mode. The LDNS round robin attribute allows the Link Controller to return a list of available virtual servers, instead of a single virtual server. Certain browsers keep the answer returned by DNS servers. By enabling this attribute, the Link Controller returns a maximum of 16 virtual servers as the answer to a DNS resolution request. This provides browsers with alternate answers if a virtual server becomes unavailable.

Setting up load balancing for services that require multiple ports

Certain types of network traffic, such as FTP traffic or e-commerce traffic, require that more than one port be available in order for the client's requests to be properly handled. When you set up a load balancing configuration, you can define a port list for a wide IP. Before the Link Controller selects a virtual server to receive a connection, it verifies that the virtual server is **up** and available to receive connection requests. When the Link Controller receives a query, all of the ports in the port list must be available for each virtual server in the wide IP. If a virtual server does not have all ports in the port list available, the Link Controller marks it as unavailable for load balancing.

To configure multiple ports for a wide IP using the Configuration utility

1. In the navigation pane, click **Wide IPs**.
2. In the Wide IP column, click a wide IP name.
The Modify Wide IP screen opens.
3. On the toolbar, click **Port List**.
The Wide IP Port List screen opens.
4. Type a port number in the box or select a service from the list, then click the **Add** button.
5. Repeat step 4 for each port or service you need to add, then click **Update**.
The port list is added to the wide IP configuration.



6

Monitoring and Administration

- Monitoring and administration utilities
- Using the bigpipe utility as a monitoring tool
- Using the Configuration utility for administration and monitoring
- Working with the Syslog utility
- Powering down the Link Controller
- Removing and returning items to service
- Viewing system statistics and log files
- Printing the connection table
- Changing passwords
- Working with the BIG/stat utility

Monitoring and administration utilities

The Link Controller platform provides several utilities for monitoring and administration of the Link Controller. You can monitor system statistics, as well as statistics specific to virtual servers and nodes, such as the number of current connections, and the number of packets processed since the last reboot.

The Link Controller platform provides the following monitoring and configuration and administration utilities:

- ◆ **bigpipe**
If you type certain **bigpipe** commands, such as **bigpipe virtual** or **bigpipe node**, and use the **show** keyword in the command, the command displays statistical information about the elements that you configure using that command. You can also use **bigpipe** commands to selectively reset any statistic collected by the Link Controller.
- ◆ **The Configuration utility**
You can use the Configuration utility to configure any feature on the Link Controller. You can reset any statistic, or all statistics, for virtual servers, nodes, NATs, and SNATs in the Configuration utility.
- ◆ **BIG/stat**
This utility is provided specifically for statistical monitoring of virtual servers, nodes, NATs, SNATs, and services. One benefit of using BIG/stat is that you can customize the display of statistical information.
- ◆ **Syslog**
Syslog is the standard UNIX system logging utility, which monitors critical system events, as well as configuration changes made on the Link Controller.

Using the bigpipe utility as a monitoring tool

Using the **bigpipe** utility, you can view information about the Link Controller itself, as well as elements such as virtual servers, virtual addresses, virtual ports, nodes, and node addresses. Typically, the **bigpipe** utility provides the following statistics:

- Current number of connections
- Maximum number of concurrent connections
- Total number of connections since the last system reboot
- Total number of bits (inbound, outbound, total)
- Total number of packets (inbound, outbound, total)

Monitoring the Link Controller

The **bigpipe summary** command displays performance statistics for the Link Controller itself. You can use these statistics to monitor and troubleshoot your Link Controller.

The output of the **bigpipe summary** command includes current usage statistics, such as the amount of time a Link Controller has been running since the last reboot. To display a summary of the performance statistics for the Link Controller, type the following command:

b summary

If you need to reboot the Link Controller but want to retain the current summary information, you can save the information in a file prior to rebooting. For more information, see *Retaining current statistics during reboot*, on page 6-8.

The performance statistics display in the format shown in Figure 6.1 (the output has been truncated for this example).

```

BIG-IP total uptime          = 1 (day) 4 (hr) 40 (min) 8 (sec)
BIG-IP total uptime (secs)   = 103208
BIG-IP total # connections   = 0
BIG-IP total # pkts          = 0
BIG-IP total # bits          = 0
BIG-IP total # pkts(inbound) = 0
BIG-IP total # bits(inbound) = 0
BIG-IP total # pkts(outbound) = 0
BIG-IP total # bits(outbound) = 0
BIG-IP error no nodes available = 0
BIG-IP tcp port deny          = 0
BIG-IP udp port deny          = 0
BIG-IP virtual tcp port deny  = 0
BIG-IP virtual udp port deny  = 0
BIG-IP max connections deny   = 0
BIG-IP virtual duplicate syn ssl = 0
BIG-IP virtual duplicate syn wrong dest = 0
BIG-IP virtual duplicate syn node down = 0
BIG-IP virtual maint mode deny = 0
BIG-IP virtual addr max connections deny = 0
BIG-IP virtual path max connections deny = 0
BIG-IP virtual non syn        = 0
BIG-IP error not in out table  = 0
BIG-IP error not in in table   = 0
BIG-IP error virtual fragment no port = 0
BIG-IP error virtual fragment no conn = 0
BIG-IP error standby shared drop = 0
BIG-IP dropped inbound        = 0
BIG-IP dropped outbound       = 0
BIG-IP reaped                 = 0
BIG-IP ssl reaped             = 0
BIG-IP persist reaped         = 0
BIG-IP udp reaped             = 0
BIG-IP malloc errors          = 0
BIG-IP bad type               = 0
BIG-IP mem pool total 96636758 mem pool used 95552 mem percent used 0.10

```

Figure 6.1 The *bigpipe summary* display screen

Explanation of summary statistics

Table 6.1 contains descriptions of each individual statistic included in the **bigpipe summary** display screen.

Statistic	Description
total uptime	Total time elapsed since the Link Controller was last booted. Output is displayed in days, hours, minutes, and seconds.
total uptime (secs)	Total uptime since the last reboot, displayed in seconds.
total # connections	Total number of connections handled by the Link Controller since the last reboot. In this case, a connection is a data path between a client machine, a virtual server, NAT or SNAT, and a node. Connections last until timed out by either a treaper or UDP timer. This is important to note, because a Link Controller with short time-outs is likely to show more connections than a Link Controller with long time-outs.
total # pkts	Total number of packets handled by the Link Controller since the last reboot. This value equals the sum of total # pkts (inbound) and total # pkts (outbound) . By dividing this value by total uptime (secs) , you can determine the average packets per second since the last reboot.
total # bits	Total number of bits handled by the Link Controller since the last reboot. This value should equal the sum of total # of bits (inbound) and total # bits (outbound) . By dividing this value by total uptime (secs) , you can determine the average bits per second since the last reboot.
total # pkts (inbound)	Total number of incoming packets handled by the Link Controller since the last reboot.
total # bits (inbound)	Total number of incoming bits handled by the Link Controller since the last reboot.
total # pkts (outbound)	Total number of outgoing packets handled by the Link Controller since the last reboot. Outgoing packets are packets that originated at the Link Controller.
total # bits (outbound)	Total number of outgoing bits handled by the Link Controller since the last reboot.
error no nodes available	The number of times the Link Controller tried to make a connection to a node, but no nodes were available. Because no nodes were available, the incoming packet was dropped. This is a serious error. If this number is non-zero, then one or more of the virtual servers is, or has been, unavailable. For example, the following shows sample statistics for systems A, B, and C. System A: BIG-IP error no nodes available = 0 System B: BIG-IP error no nodes available = 11409162 System C: BIG-IP error no nodes available = 42 In this example, the statistics show that System A is healthy. System B, however, is unhealthy; in slightly over a month of up time, 11,000,000 instances of an unavailable server have occurred. System C is also unhealthy; although the number of times a virtual server has become unavailable is minimal, this averages to one such event per day of up time on this system. It should be possible to keep this number to zero.

Table 6.1 *bigpipe monitoring statistics*

Statistic	Description
tcp port deny	The number of times a client attempted to connect to an unauthorized TCP port on the Link Controller. This statistic indicates that a connection was attempted, but the TCP port requested was not open. Because the port was not open, the incoming packet was discarded. This applies only to administrative and alias addresses on the Link Controller, and not virtual servers, NATs, or SNATs. If the sysctl bigip.verbose_log_level is set to 2 or 15, the offending port and source IP address are logged in the file /var/log/bigip. Alternatively, the location of this log can be set in the Advanced Properties section of the GUI. For example, the following shows sample statistics for systems A, B, and C: System A: BIG-IP tcp port deny = 0 System B: BIG-IP tcp port deny = 16358 System C: BIG-IP tcp port deny = 58 In these examples, there is a wide disparity in port denials from system to system. System A is behind a firewall and has been available for a relatively short amount of time. System B is not behind a firewall, and because of this, the system is intercepting many packets that would otherwise be filtered out by a firewall. System C is behind a firewall, but a few packets appear not to have been discarded. In this case, the administrator should examine the Link Controller logs to determine the port and source address of these packets and then filter them out at the firewall.
udp port deny	The number of times a client attempted to connect to an unauthorized UDP port on the Link Controller. This statistic indicates that a connection was attempted, but the UDP port requested was not open. Because the port was not open, the incoming packet was discarded. This applies only to administrative and alias addresses on the Link Controller, and not virtual servers, NATs, or SNATs. If the sysctl bigip.verbose_log_level is set to 1 or 15, the offending port and source IP address are logged in the file /var/log/bigip. Alternatively, the location of this log can be set in the Advanced Properties section of the GUI. For example, the following shows sample statistics for systems A, B, and C: System A: BIG-IP udp port deny = 0 System B: BIG-IP udp port deny = 16358 System C: BIG-IP udp port deny = 58 In these examples, there is a wide disparity in port denials from system to system. System A is behind a firewall and has been available for a relatively short amount of time. System B is not behind a firewall, and because of this, the system is intercepting many packets that would otherwise be filtered out by a firewall. System C is behind a firewall, but a few packets appear not to have been discarded. In this case, the administrator should examine the Link Controller logs to determine the port and source address of these packets and then filter them out at the firewall.

Table 6.1 *bigpipe monitoring statistics*

Statistic	Description
virtual tcp port deny	The number of times a client attempted to connect to an unauthorized TCP port on a virtual IP address. This statistic indicates that a connection was attempted to a virtual IP address, but used a TCP port that was not open. Because the port was not enabled for this virtual server, the incoming packet was discarded. If the sysctl bigip.verbose_log_level is set to 8 or 15, the offending port and source IP address are logged in the file /var/log/bigip. Alternatively, the location of this log can be set in the Advanced Properties section of the GUI. For example, the following shows sample statistics for systems A, B, and C: System A: BIG-IP virtual tcp port deny = 67 System B: BIG-IP virtual tcp port deny = 221313359 System C: BIG-IP virtual tcp port deny = 447 This is another statistic where a critical factor is whether or not the Link Controller is located behind a firewall. Systems A and C are placed behind a firewall, while System B is not. Because System B is not behind a firewall, it has had to discard many more stray packets (over 221,000,000) than the systems protected by a firewall. Although Systems A and C are located behind firewalls, some packets are still slipping through. This indicates that the firewall administrator has probably allowed all ports for each virtual IP address. By tightening security on the firewalls, it should be possible to reduce this statistic to zero. This example applies also to the next section.
virtual udp port deny	The number of times a client attempted to connect to an unauthorized UDP port on a virtual IP address. This statistic indicates that a connection was attempted to a virtual IP address, but used a TCP port that was not open. Because the port was not enabled for this virtual server, the incoming packet was discarded. If the sysctl bigip.verbose_log_level is set to 4 or 15, the offending port and source IP address are logged in the file /var/log/bigip. Alternatively, the location of this log can be set in the Advanced Properties section of the GUI. For example, the following shows sample statistics for systems A, B, and C: System A: BIG-IP virtual tcp port deny = 67 System B: BIG-IP virtual tcp port deny = 221313359 System C: BIG-IP virtual tcp port deny = 447 This is another statistic where a critical factor is whether or not the Link Controller is located behind a firewall. Systems A and C are placed behind a firewall, while System B is not. Because System B is not behind a firewall, it has had to discard many more stray packets (over 221 million) than the systems protected by a firewall. Although Systems A and C are located behind firewalls, some packets are still slipping through. This indicates that the firewall administrator has probably allowed all ports for each virtual IP address. By tightening security on the firewalls, it should be possible to reduce this statistic to zero.
max connections deny	The total number of connections denied because the maximum number of connections allowed was exceeded. This error indicates that the maximum number of connections was received on a port that has a connection limit configured. If port connection limits are configured, it is a good idea to monitor this statistic. The statistic indicates the exact amount of traffic being discarded by the port limit. You can set port connection limits using the following bigpipe service syntax: bigpipe service <port> limit <seconds>

Table 6.1 *bigpipe* monitoring statistics

Statistic	Description
virtual duplicate syn ssl	This error is obsolete and therefore should always be zero.
virtual duplicate syn wrong dest	The number of duplicate connection attempts from the same client (IP address and port combination) to a different virtual server. This error indicates that a connection arrived from a source port and IP address for which a connection already existed on the Link Controller; however, this connection requested a different destination. The existing connection was torn down and a new connection was created. This is most likely to occur with packets coming from busy proxies where source ports are constantly being reused. These errors are common and should not be a cause for alarm. For example, the following shows sample statistics for systems A, B, and C: System A: BIG-IP virtual duplicate syn wrong dest = 0 System B: BIG-IP virtual duplicate syn wrong dest = 2673815 System C: BIG-IP virtual duplicate syn wrong dest = 324 In this example, System A has probably not encountered this error due to its limited up time. System B has a very large value because the HTML contains links to another virtual server on the Link Controller. When these links are followed and coincide with reuse of a source port, the error is generated. System C has a fairly average value. It is receiving some source port reuse, probably from large proxies such as AOL.
virtual duplicate syn node down	The number of duplicate connection attempts to a server that is unavailable when a connection to the server was made previously. This error indicates that a syn packet arrived and that a connection to a node was created. Later, another syn packet arrived but the node was unavailable. The existing connection was severed and a new connection to a different node was created to service the request. This error is common when nodes are unreliable or over-burdened. The seriousness of this error varies from site to site. In sites where there are a large number of low-capacity nodes, you see this error as nodes become over-burdened and then unavailable. This error is also more common with some load balancing modes, where certain nodes receive more connections than others. For example, the following shows sample statistics for systems A, B, and C: System A: BIG-IP virtual duplicate syn node down = 0 System B: BIG-IP virtual duplicate syn node down = 7442773 System C: BIG-IP virtual duplicate syn node down = 1 In this example, Systems A and C are healthy, while System B appears to have a serious problem. System B nodes frequently appear to be marked unavailable.
virtual maint mode deny	The number of times a connection to a virtual server was denied while the Link Controller was in maintenance mode. Current maintenance status can be determined by using the bigpipe maint command.
virtual addr max connections deny	The number of virtual IP address connections that were dropped because the maximum number of connections configured was exceeded. A connection limit on a virtual IP address can be configured using the bigpipe virtual command, as shown in the following example: bigpipe virtual 172.16.1.1 limit 100

Table 6.1 *bigpipe monitoring statistics*

Statistic	Description
virtual path max connections deny	The number of virtual path connections that were dropped because the maximum number of connections configured was exceeded. A connection limit on a virtual server can be configured using the bigpipe virtual command, as shown in the following example: <pre>bigpipe virtual 172.16.1.1:80 limit 100</pre>
virtual non syn	The number of packets received that are not connection requests, and are destined to a virtual IP address but not a valid virtual server (port). This error indicates that a packet arrived with a destination of a virtual server. However, this packet was not a syn and there was no existing connection. The packet was discarded because Link Controller only creates a new connection to a virtual server when it receives a syn packet. This error is common and could occur for a variety of reasons. It is typical for this number to grow very large. In most cases, this error occurs when packets arrive out of sequence or when packets are dropped before arriving at the Link Controller. For example, the following shows sample statistics for systems A, B, and C: <pre>System A: BIG-IP virtual non syn = 2 System B: BIG-IP virtual non syn = 1609400253 System C: BIG-IP virtual non syn = 10366</pre> In this example, System A and System C show fairly typical values based on their up times. System B shows a very large value which may indicate a problem. One possibility is that System B has a heavy traffic load and that a significant portion of this traffic must traverse a very long distance before arrival (European traffic to the US west coast, for example). Long physical distance seems to contribute to the kind of packet loss that results in this error. Another possibility is that a device on the local network is dropping packets. Large values in this statistic should be investigated at least enough to ensure that a local device is not the cause of the problem.
error virtual fragment no port	The number of IP fragments for which there is no port. This error indicates that a packet fragment arrived, but it was not the first fragment. Since the first fragment contains the port number and is required to set up a new connection, Link Controller discards the fragment. This error is similar to virtual non syn, but refers to fragmented, rather than intact, packets. As with virtual non syn, this probably occurs because fragments arrive out of sequence or not at all. This is a common error and not usually a cause for concern.
error virtual fragment no conn	The number of IP fragments for which there is no connection. This error indicates that a packet fragment arrived, but it was not the first fragment. Link Controller previously received the first fragment, but it was not a syn packet and did not match an existing connection, causing it to be discarded. This error is most likely to occur in conjunction with a virtual non syn. Thus, when Link Controller received the first packet, it found that the packet met the criteria for virtual non syn and dropped it; this error simply indicates that the Link Controller is dropping the following fragments of that packet. As with virtual non syn and virtual fragment no port, this error is relatively common and should not cause much concern.

Table 6.1 *bigpipe monitoring statistics*

Statistic	Description
error standby shared drop	The number of packets destined to the shared IP address in a redundant system that are received and ignored by the standby system. This error indicates that a packet arrived at the standby Link Controller with a destination of the shared alias. Because the Link Controller was in standby mode, this packet was dropped. This is most likely to occur when a Link Controller has failed over, but the arp cache of another device has not yet been updated. Because that device is caching an invalid arp request, the device continues to send packets to the previously active Link Controller.
dropped inbound	The total number of inbound packets dropped by the Link Controller because the source IP address is other than a Link Controller address.
dropped outbound	The total number of outbound packets dropped by the Link Controller because the source IP address is the Link Controller address.
reaped	The total number of TCP connections that timed-out and were therefore deleted by the Link Controller.
ssl reaped	The total number of SSL persistence records based on SSL session IDs that timed-out and were therefore deleted by the Link Controller. The timeout for SSL persistence is configured when creating the virtual server.
persist reaped	The total number of non-SSL persistence records that timed-out and were therefore deleted by the Link Controller. Persistence timeouts are configured when creating the virtual server.
udp reaped	The total number of UDP connections that timed-out and were therefore deleted by the UDP timer of the Link Controller. UDP timers are configured with the bigpipe command, using the following syntax: <pre>bigpipe udp <port> <timeout></pre>
malloc errors	The number of times a connection could not be created due to insufficient memory. If this error is occurring, the Link Controller is overloaded and the configuration should either be streamlined, or the software and/or chassis should be upgraded.
mem pool total	The total amount of memory available in all combined memory pools.
mem pool used	The total amount of memory, in all combined memory pools, in use by the Link Controller.
mem percent used	The total percentage of memory in use by all combined memory pools.

Table 6.1 *bigpipe* monitoring statistics

Retaining current statistics during reboot

When you reboot the Link Controller, all existing summary statistics are deleted. If you need to reboot the Link Controller but want to save the existing summary statistics, you can create a shell script that saves the

statistics and copies them to another file. Figure 6.2 shows an example of this script, named **restart**. Note that this script must reside in the **/sbin** directory.

```
# restart script
bigpipe summary >/var/log/summary.`date +%d%m%y%H%M`
bigpipe ms >/var/log/ms.`date +%d%m%y%H%M`
echo "Reboot by /sbin/restart at `date +%d%m%y%H%M`" >/var/log/bigip
/bin/sync;/bin/sync
/sbin/reboot
```

Figure 6.2 Script for saving summary statistics

Resetting statistics on the Link Controller

The **bigpipe** commands allow you to selectively reset any statistic on the Link Controller. The statistics you can reset selectively include:

- Virtual address
- Virtual server
- Node address
- Node server
- Virtual port
- Network address translations (NATs)
- Secure network address translations (SNATs)
- Global statistics

When you reset one of these items, the packets in, packets out, bytes in, and bytes out counters of the target item are reset to zero. The maximum connection count counter is also reset. The current connections counter is not reset, and the total connections counter is set equal to the number of current connections.

◆ Note

The statistics are reset for the specified items only. Statistics for dependent items, such as node servers for a given virtual address, are not modified by these commands. The only exception is the global statistics reset option which resets traffic statistics for all items. After an item-level reset, statistics for all other dependent items do not add up.

You can create an audit trail for reset events by setting an optional system control variable. You can set this variable to generate a syslog log entry. To set this variable, type the following command:

```
b internal set verbose_log_level=4
```

To reset statistics for virtual servers and virtual addresses

Use the following syntax to reset statistics for the virtual address specified by the IP address **<virtual ip>**.

```
b virtual <virtual_ip> stats reset
```

For example, if you want to reset statistics for the virtual address **172.20.1.100**, type the following command:

```
b virtual 172.20.1.100 stats reset
```

If you want to reset statistics for a list of virtual addresses, type the command with a list of addresses separated by spaces:

```
b virtual 172.20.1.100 172.20.1.101 172.20.1.102 stats reset
```

If you want to reset statistics for all virtual servers, use the following command:

```
b virtual stats reset
```

Use the following syntax to reset statistics for the virtual server IP:port combination **<virtual_ip>:<port>**.

```
b virtual <virtual_ip>:<port> stats reset
```

For example, if you want to reset statistics for the virtual address/port combination **172.20.1.100:80**, type the following command:

```
b virtual 172.20.1.100:80 stats reset
```

If you want to reset statistics for a list of virtual address/port combinations, type the command with the list of addresses separated by spaces:

```
b virtual 172.20.1.100:80 172.20.1.100:23 172.20.1.101:80 stats reset
```

To reset statistics for node servers and node addresses

Use the following syntax to reset statistics for all node addresses and node servers:

```
b node stats reset
```

You can reset statistics for the node address specified by the IP address **<node_ip>**:

```
b node <node_ip> stats reset
```

For example, to reset the statistics for the node address 10.1.1.1, use the following command:

```
b node 10.1.1.1 stats reset
```

If you want to reset statistics for a list of node addresses, type the command with the list of addresses separated by spaces:

```
b node 10.1.1.1 10.1.1.2 10.1.1.3 stats reset
```

Use the following syntax to reset statistics for the node server specified by the IP:port combination **<node_ip>:<port>**:

```
b node <node_ip>:<port> stats reset
```

For example, to reset the statistics for the node server **10.1.1.1:80**, use the following command:

```
b node 10.1.1.1:80 stats reset
```


If you want to reset statistics for a list of node server addresses, type the command with the list of addresses separated by spaces:

```
b node 10.1.1.1:80 10.1.1.2:23 stats reset
```

To reset statistics for virtual ports

Use the following syntax to reset statistics for all virtual ports:

```
b service stats reset
```

Use the following syntax to reset statistics for the virtual port **<port>**. You can specify a list of virtual ports separated by spaces:

```
b service <port> stats reset
```

For example, to reset the statistics for the virtual port **80**, use the following command:

```
b service 80 stats reset
```

To reset the statistics for a list of virtual ports, use the following command:

```
b service 23 80 443 stats reset
```

To reset statistics for network address translations (NATs)

Use the following syntax to reset statistics for all NATs:

```
b nat stats reset
```

Use the following syntax to reset statistics for the NAT for the IP address **<orig_ip>**.

```
b nat <orig_ip> stats reset
```

For example, to reset the statistics for the NAT **172.20.3.101**, use the following command:

```
b nat 172.20.3.101 stats reset
```

To reset the statistics for a list of origin IPs, use the following command where addresses are separated by spaces:

```
b nat 172.20.3.101 172.20.3.102 stats reset
```

To reset statistics for secure network address translations (SNATs)

Use the following syntax to reset statistics for all SNATs:

```
b snat stats reset
```

Use the following syntax to reset statistics for the SNAT for IP address **<snat_ip>**:

```
b snat <snat_ip> stats reset
```

For example, to reset the statistics for the SNAT **172.20.3.101**, use the following command:

```
b snat 172.20.3.101 stats reset
```

To reset the statistics for a list of SNAT origin addresses, use the following command where addresses are separated by spaces:

```
b snat 172.20.3.101 172.20.3.102 stats reset
```

To reset global statistics

Use the following command to reset all statistics for all items:

```
b global stats reset
```

To reset any statistic in the Configuration utility

A **Reset** button is located in the Configuration utility in each of the following tables:

- Virtual address
- Virtual server
- Node address
- Node server
- Virtual port
- Network address translations (NATs)
- Global statistics

To reset a statistic for a particular item, click the **Reset** button next to the item in one of these tables.

Monitoring virtual servers, virtual addresses and services

You can use different variations of the **bigpipe virtual** command, as well as the **bigpipe port** command, to monitor information about virtual servers, virtual addresses, and services managed by the Link Controller.

Displaying information about virtual servers and virtual addresses

The **bigpipe virtual** command displays the status of virtual servers (**up**, **down**, **unchecked**, or **disabled**), the current number of connections to each virtual server, and the status of the member nodes that are included in each virtual server mapping. The status for individual member nodes includes whether the node is **up**, **down**, **unchecked**, or **disabled** and also includes the cumulative count of packets and bits received and sent by the node on behalf of the virtual server. The Link Controller displays the statistics as shown in Figure 6.3.

```

virtual +-----> 11.11.11.50          UNIT 1
|                (cur, max, limit, tot) = (0, 8, 0, 370)
|                (pkts,bits) in = (10704, 8744872), out = (21480, 230874016)
+----+--> PORT http                    UP
|                (cur, max, limit, tot) = (0, 8, 0, 370)
|                (pkts,bits) in = (10704, 8744872), out = (21480, 230874016)
POOL appgen_11.11.11.50.80
MEMBER 11.12.11.100:http                UP
|                (cur, max, limit, tot) = (0, 8, 0, 370)
|                (pkts,bits) in = (10704, 8744872), out = (21480, 230874016)

virtual +-----> 11.11.11.101         UNIT 1
|                (cur, max, limit, tot) = (0, 2, 0, 4)
|                (pkts,bits) in = (4532, 2090768), out = (6824, 82113984)
+----+--> PORT http                    UP
|                (cur, max, limit, tot) = (0, 2, 0, 4)
|                (pkts,bits) in = (4532, 2090768), out = (6824, 82113984)
POOL my_website_pool
MEMBER 11.12.11.100:http                UP
|                (cur, max, limit, tot) = (0, 2, 0, 4)
|                (pkts,bits) in = (4532, 2090768), out = (6824, 82113984)

```

Figure 6.3 Virtual server statistics

If you want to view statistical information about one or more specific virtual servers, simply include the virtual servers in the **bigpipe virtual show** command as shown below:

```
b virtual <virt addr>:<port> ... <virt addr>:<port> show
```

If you want to view statistical information about traffic going to one or more virtual addresses, specify only the virtual address information in the command:

```
b virtual <virt addr> ... <virt addr> show
```

Displaying information about services

The **bigpipe port show** command allows you to display information about specific virtual ports managed by the Link Controller. You can use the command to display information about all virtual services, or you can specify one or more particular virtual services.

To view information about all virtual services, use the following syntax:

```
b service show
```

To view statistical information about one or more specific virtual services, simply include the service names or port numbers as shown below:

```
b service <port> ... <port> show
```

Monitoring nodes and node addresses

The **bigpipe node** command displays the status of all nodes configured on the Link Controller. The information includes whether the specified node is **up**, **down**, **disabled**, or **unchecked**, and the number of cumulative packets and bits sent and received by each node on behalf of all virtual servers. The Link Controller displays the statistical information as shown in Figure 6.4.

```

NODE 11.12.11.100      UP
|      (cur, max, limit, tot) = (0, 8, 0, 374)
|      (pckts,bits) in = (15236, 10835640), out = (28304, 312988000)
+-      PORT http      UP
      (cur, max, limit, tot) = (0, 8, 0, 374)
      (pckts,bits) in = (15236, 10835640), out = (28304, 312988000)

```

Figure 6.4 Node statistics screen

If you want to view statistical information about one or more specific nodes, simply include the nodes in the **bigpipe node show** command as shown below:

b node <node addr>:<port> ... <node addr>:<port> show

If you want to view statistical information about traffic going to one or more node addresses, specify only the node address information in the command:

b node <node addr> ... <node addr> show

Monitoring NATs

The **bigpipe nat show** command displays the status of the NATs configured on the Link Controller. The information includes the number of cumulative packets and bits sent and received by each NAT.

To display NAT status from the command line

Use the following command to display the status of all NATs included in the configuration:

b nat show

Use the following syntax to display the status of one or more selected NATs:

b nat <node addr> [...<node addr>] show

An example of the output for this command is shown in Figure 6.5.

```

NAT { 10.10.10.3 to 9.9.9.9 }
  (pckts,bits) in = (0, 0), out = (0, 0)
NAT { 10.10.10.4 to 12.12.12.12
  netmask 255.255.255.0 broadcast 12.12.12.255 }
  (pckts,bits) in = (0, 0), out = (0, 0)

```

Figure 6.5 NAT statistics

Monitoring SNATs

The **bigpipe snat show** command displays the status of the SNATs configured on the Link Controller. The information includes connections and global SNAT settings.

To show SNAT details from the command line

Use the following **bigpipe** command to show SNAT mappings:

```
b snat [<SNAT addr>] [...<SNAT addr>] show  
b snat show
```

Use the following command to show the current SNAT connections:

```
b snat [<snat_ip>...] dump [ verbose ]  
b snat dump [ verbose ]
```

The optional **verbose** keyword provides more detailed output.

The following command prints the global SNAT settings:

```
b snat globals show
```

Viewing the status of the interface cards

The **bigpipe interface** command displays the current status and the settings for external and internal interface cards. You can also use the **bigpipe interface** command to view information for a specific interface card, using the following command syntax:

```
b interface <ifname> -show
```

Using the Configuration utility for administration and monitoring

You can use the Configuration utility System Admin screen to add users, customize the user interface, configure SNMP, and save and restore a current configuration.

You can use the Configuration utility to allow access to the SNMP agent and to set SNMP properties. For more information on configuring SNMP, refer to Chapter 9, *Configuring SNMP*.

Adding a user

You can add a user to the Link Controller using the Configuration utility.

To add a user to the Link Controller using the Configuration utility

1. In the navigation pane, click **System Admin**.
The System Admin tabs appear.
2. Click the User Administration tab.
The Add User screen opens. This screen contains a list of current users.
3. In the appropriate field, type the User ID, password, and access level for the user.
For more information on the Add User screen, click the **Help** button.

Customizing the Configuration utility

You can customize the appearance of the Configuration utility.

To customize the Configuration utility

1. In the navigation pane, click **System Admin**.
The System Admin tabs appear.
2. Click the Web UI Administration tab.
The Web UI Administration screen opens.
3. Select the options you want to configure.
For more information about the options available on this screen, click the **Help** button.

Configuring SNMP

For information on configuring SNMP, refer to Chapter 9, *Configuring SNMP*.

Working with the Syslog utility

The Link Controller supports logging using the **Syslog** utility. The logs are generated automatically, and saved in user-specified files. These logs contain all changes made to the Link Controller configuration, such as those made with the **bigpipe virtual** command, or other **bigpipe** commands, as well as all critical events that occur in the system.

◆ **Note**

You can configure the Syslog utility to send email or activate pager notification based on the priority of the logged event.

The Syslog log files track system events based on information defined in the `/etc/syslog.conf` file. You can view the log files in a standard text editor, or with the `less` file page utility.

Sample log messages

Table 6.2 shows sample log messages to give you an idea of how the Syslog utility tracks events that are specific to the Link Controller.

Sample message	Description
bigd: allowing connections on port 20	A user specifically allowed connections on virtual port 20.
bigd: node 192.168.1.1 detected up	The 192.168.1.1 node address was successfully pinged by the Link Controller.
bigd: added service port 20 to node 192.168.1.1	A user defined a new node, 192.168.1.1:20.
kernel: security: port denial 207.17.112.254:4379 -> 192.168.1.1:23	A client was denied access to a specific port. The client is identified as coming from 207.17.112.254:4379, and the destination node is 192.168.1.1:23.

Table 6.2 Sample Syslog messages

Powering down the Link Controller

If you want to power down, or turn off, the Link Controller, you need to complete two tasks. The first task is to shut down the Link Controller software. After you shut down the Link Controller software, you can turn the power to the system off.

To shut down the Link Controller software from the command line

To complete the first task to shut down the Link Controller software, type the following command:

```
halt
```

After the system halts, you can turn off the power to the system.

Removing and returning items to service

Once you have completed the initial configuration on the Link Controller, you may want to temporarily remove specific items from service for maintenance purposes. For example, if a specific network server needs to be

upgraded, you may want to disable the nodes associated with that server, and then enable them once you finish installing the new hardware and bring the server back online.

If you specifically disable the nodes associated with the server, the Link Controller allows the node to go down only after all the current connections are complete. During this time, the Link Controller does not attempt to send new connections to the node. Although the Link Controller monitoring features do eventually determine that the nodes associated with the server are down, specifically removing the nodes from service can prevent interruptions on long duration client connections.

You can remove the entire Link Controller from service, or you can remove the following individual items from service:

- Virtual servers
- Virtual addresses
- Virtual ports
- Nodes
- Node addresses

Removing the Link Controller from service

The Link Controller platform offers a Maintenance mode, which allows you to remove the Link Controller from network service. This is useful if you want to perform hardware maintenance, or make extensive configuration changes. When you activate Maintenance mode, the Link Controller no longer accepts connections to the virtual servers it manages. However, the existing connections are allowed to finish processing so that current clients are not interrupted.

The **bigpipe maint** command toggles the Link Controller into or out of Maintenance mode. Use the following command to put the Link Controller in maintenance mode:

```
b maint
```

If the Link Controller runs in Maintenance mode for less than 20 minutes and you return the machine to the normal service, the Link Controller quickly begins accepting connections. However, if the Link Controller runs in Maintenance mode for more than 20 minutes, returning the unit to service involves updating all network ARP caches. This process can take a few seconds, but you can speed the process up by reloading the **/config/bigip.conf** file using the following command:

```
b -f /config/bigip.conf
```

To activate maintenance mode using the Configuration utility

1. In the navigation pane, click **System**.
The Network Map screen opens.

2. Click the Properties tab.
The Properties screen opens.
3. Check the **Maintenance Mode** box.
4. Click the **Apply** button.

Removing individual virtual servers, virtual addresses, and ports from service

The Link Controller also supports taking only selected virtual servers, addresses, or ports out of service, rather than removing the Link Controller itself from service. Each **bigpipe** command that defines virtual servers and their components supports **enable** and **disable** keywords, which allow you to remove or return the elements from service.

When you remove a virtual address or a virtual port from service, it affects all virtual servers associated with the virtual address or virtual port. Similarly, if you remove a node address from service, it affects all nodes associated with the node address.

Enabling and disabling virtual servers and virtual addresses

The **bigpipe virtual** command allows you to enable or disable individual virtual servers, as well as virtual addresses.

To enable or disable a virtual server from the command line

To enable or disable a virtual server, type the appropriate command:

```
b virtual <virtual addr>:<virtual port> enable
```

```
b virtual <virtual addr>:<virtual port> disable
```

To enable or disable a virtual address, type the appropriate command:

```
b virtual <virtual addr> enable
```

```
b virtual <virtual addr> disable
```

Enabling and disabling virtual ports

The **bigpipe port** command allows you to allow or deny traffic on a virtual port.

To allow or deny traffic on a virtual port from the command line

Use the following syntax to allow or deny traffic on a virtual port.

```
b service <virtual port> enable
```

```
b service <virtual port> disable
```

Removing individual nodes and node addresses from service

You can enable or disable individual and node addresses from the command line.

To enable and disable nodes and node addresses from the command line

The **bigpipe node** command allows you to enable or disable individual nodes, as well as node addresses.

To enable or disable a *node*, type the appropriate command:

```
b node <node addr>:<node port> enable
```

```
b node <node addr>:<node port> disable
```

To enable or disable a *node address*, type the appropriate command:

```
b node <node addr> enable
```

```
b node <node addr> disable
```

Viewing the currently defined virtual servers and nodes

When used with the **show** parameter, **bigpipe** commands typically display currently configured elements. For example, the **bigpipe virtual show** command displays all currently defined virtual servers, and the **bigpipe node** command displays all nodes currently included in virtual server mappings. For additional information about using **bigpipe** commands on the Link Controller, see Appendix A, *bigpipe Command Reference*.

Viewing system statistics and log files

The Configuration utility allows you to view a variety of system statistics and system log files. Note that from each statistics screen, you can access property settings for individual virtual servers, nodes, IP addresses, and ports by selecting the individual item in the statistics table.

Viewing system statistics

The Configuration utility allows you to view the following statistical information:

- Link Controller system statistics, including the elapsed time since the last system reboot, the number of packets and connections handled by the system, and the number of dropped connections
- Virtual servers, including virtual servers, virtual address only, or virtual ports only
- Nodes, including nodes, node addresses only, or node ports only
- NAT statistics, such as the number of packets handled by each NAT

- SNAT statistics, such as SNAT mappings
- IP filter statistics, including the number of packets accepted and rejected by individual IP filters
- Rate filter statistics, including the number of bits passed through, delayed, and dropped by individual rate filters
- Information about illegal connection attempts, such as the source IP addresses from which the illegal connection is initiated

Statistics are displayed in real-time. You can specify the update frequency by setting an interval (in seconds), and then clicking **Update**.

Viewing log files

The Configuration utility allows you to display three different log files:

- The Link Controller system log, which displays standard UNIX system events
- The Link Controller log, which displays information specific to Link Controller events, such as defining a virtual server
- The Pinger log, which displays status information determined by each node ping issued by the Link Controller

Printing the connection table

The **bigpipe** command line utility offers a useful diagnostic tool that prints the list of current connections. Normally, the **bigpipe conn** command prints the client, virtual server, and node addresses.

Changing passwords

When you run the Setup utility, you define a password that allows remote access to the Link Controller, and you also define a password for the Link Controller web server. You can change these passwords at any time.

To change the Link Controller password

1. At the Link Controller command line prompt, log on as the root user and use the **passwd** command.
2. At the password prompt, enter the password you want to use for the Link Controller and press Enter.
3. To confirm the password, retype it and press Enter.

Changing passwords and adding users for the Configuration utility

You can create new users for the Link Controller web server by using the Configuration utility.

The user accounts you create in the Configuration utility can have full, partial, or read-only access to the Link Controller.

To create user accounts using the Configuration utility

1. In the navigation pane, click **User Admin**.
The User Administration screen opens.
2. In the Add User section, type the following information.
  - **User ID**
Type the user ID you want to assign the user.
  - **Password**
Type the password you want to assign the user.
  - **Retype Password**
Retype the password you want to assign the user.
3. In the Current Users list, select the access level for the user. The access levels available are:
  - **Read Only**
This access level allows the user only to view information in the Configuration utility. Users with this access level do not have access to **Add** buttons, certain tab items, **Apply** buttons, or **Remove** buttons.
  - **Partial Read/Write**
In addition to allowing the user to view information, a Partial Read/Write user can also change the status of node addresses to either **enabled** or **disabled**.
  - **Full Read/Write**
This access level provides the user with full access to all administrative tasks.
4. After you select the access level for the user, click the **Add** button.

The Current User list on the User Administration screen contains all users configured to access the Configuration utility. You can delete any user added through the Configuration utility by clicking the **Remove** button next to the user in the list. The Link Controller web server administrator account you created with the Setup utility shows up in this list. However, you cannot edit or delete this account from the Configuration utility. To edit this account, you must run the Setup utility and choose the **Configure Web servers** menu option. For more information about this utility, see Chapter 2, *Using the Setup Utility*, in the **BIG-IP Link Controller Solutions Guide**.

Working with the BIG/stat utility

BIG/stat™ is a utility that allows you to quickly view the status of the following elements:

- Virtual servers
- Services (**cur**, **max**, **limit**, **tot**) (**pckts**,**bits**) in out
- Nodes (**cur**, **max**, **limit**, **tot**) (**pckts**,**bits**) in out
- Ports
- Network address translations (NATs)

You can customize the BIG/stat utility statistics display. For example, you can customize your output to display statistics for a single element, or for selected elements. You can set the display to automatically update at time intervals you specify.

The **bigstat** command accepts one or more options, which allow you to customize the statistical display. When you use the **bigstat** command without specifying any options, the BIG/stat utility displays the list of virtual servers, services, nodes, NATs, and SNATs only one time. The basic command syntax is:

```
bigstat [ options... ]
```

Table 6.3 describes the options that you can use in the **bigstat** command.

Option	Description
-bigip	Displays totals for the Link Controller overall.
-c <count>	Sets the interval at which new information is displayed.
-h and -help	Displays the help options.
-n	Displays data in numeric format.
-nat	Displays network address table (NAT) entries only.
-no_virtualtot	Removes virtual server totals from the display.
-no_nodetot	Removes node totals from the display.
-node	Displays nodes only.
-port	Displays ports only.
-v	Displays version information.
-virtual	Displays virtual servers only.

Table 6.3 The *bigstat* command options

Figure 6.6 contains an example of the output from the **bigstat** command. Table 6.4 contains descriptions of each of the items in this example.

```
bigip springbank
    (cur, max, tot) = (0, 8, 374)
    (pckts,bits) in = (15310, 10860064), out = (28363, 313009048)
virtual 11.11.11.50
    (cur, max, limit, tot) = (0, 8, 370, 370)
    (pckts,bits) in = (10704, 8744872), out = (21480, 230874016)
virtual 11.11.11.50:http UP
    (cur, max, limit, tot) = (0, 8, 370, 370)
    (pckts,bits) in = (10704, 8744872), out = (21480, 230874016)
virtual 11.11.11.101
    (cur, max, limit, tot) = (0, 2, 4, 4)
    (pckts,bits) in = (4532, 2090768), out = (6824, 82113984)
virtual 11.11.11.101:http UP
    (cur, max, limit, tot) = (0, 2, 4, 4)
    (pckts,bits) in = (4532, 2090768), out = (6824, 82113984)
node 11.12.11.100 UP
    (cur, max, limit, tot) = (0, 8, 374, 374)
    (pckts,bits) in = (15236, 10835640), out = (28304, 312988000)
node 11.12.11.100:http UP
    (cur, max, limit, tot) = (0, 8, 374, 374)
    (pckts,bits) in = (15236, 10835640), out = (28304, 312988000)
port WILDCARD PORT
    (cur, max, limit, tot, reaped) = (0, 0, 0, 0, 0)
    (pckts,bits) in = (0, 0), out = (0, 0)
port 80:http
    (cur, max, limit, tot, reaped) = (0, 8, 374, 374, 6)
    (pckts,bits) in = (15236, 10835640), out = (28304, 312988000)
```

Figure 6.6 Sample output of the **bigstat** command

The following table contains descriptions of each of the metrics collected for the Link Controller.

BIG/stat Item	Description
BIG-IP	cur - Shows the number of current connections handled by the Link Controller max - Shows the maximum number of connections handled by the Link Controller tot - Shows the total number of connections handled by the Link Controller pckts,bits in - Shows the total number of packets and bits coming into the Link Controller pckts,bits out - Shows the total number of packets and bits going out of the Link Controller
virtual server	cur - Shows the number of current connections handled by the virtual server max - Shows the maximum number of connections handled by the virtual server limit - Shows the connection limit reached by the virtual server tot - Shows the total number of connections handled by the virtual server pckts,bits in - Shows the total number of packets and bits coming into the virtual server pckts,bits out - Shows the total number of packets and bits going out of the virtual server
service	cur - Shows the number of current connections handled by the service max - Shows the maximum number of connections handled by the service limit - Shows the connection limit reached by the service tot - Shows the total number of connections handled by the Link Controller service pckts,bits in - Shows the total number of packets and bits coming into the service pckts,bits out - Shows the total number of packets and bits going out of the service
nodes	cur - Shows the number of current connections handled by the node max - Shows the maximum number of connections handled by the node limit - Shows the connection limit reached by the node tot - Shows the total number of connections handled by the Link Controller node pckts,bits in - Shows the total number of packets and bits coming into the node pckts,bits out - Shows the total number of packets and bits going out of the node
ports	cur - Shows the number of current connections handled by the port max - Shows the maximum number of connections handled by the port limit - Shows the connection limit reached by the port tot - Shows the total number of connections handled by the Link Controller port reaped - Shows the number of connections reaped on the port pckts,bits in - Shows the total number of packets and bits coming into the port pckts,bits out - Shows the total number of packets and bits going out of the port

Table 6.4 Data displayed by the *bigstat* utility



7

Additional Setup Options

- Overview of additional setup options
- Defining additional host names
- Using the MindTerm SSH Console
- Downloading the SSH client to your administrative workstation
- Addressing general networking issues
- Using a serial terminal with the Link Controller
- Configuring RADIUS or LDAP authentication

Overview of additional setup options

This chapter contains details about additional setup options you may want to configure for the Link Controller. The options described in this chapter include:

- Defining additional host names
- Preparing workstations for command line access
- Addressing general networking issues
- Using a serial terminal with the Link Controller
- Configuring RADIUS and LDAP authentication

Defining additional host names

Once you complete the Setup utility, you may want to insert additional host names and IP addresses for network devices into the `/etc/hosts` file to allow for more user-friendly system administration. In particular, you may want to create host names for the IP addresses that you will assign to virtual servers. You may also want to define host names for standard devices such as your routers, network interface cards, and the servers or other equipment that you are load balancing.

The `/etc/hosts` file, as created by the Setup utility, is similar to the example shown in Figure 7.1.

```
# BIG-IP(R) Hosts Table    Generated by Setup utility on Thu May 16 11:03:03 PDT 2002

# localhost entry
127.1    localhost

# default gateway entry
11.11.11.10    router

# Local name
11.11.11.2     bigip1.mynet.net

# Peer name (state mirror)
11.12.11.1     peer

#
# vlans
#
11.11.11.2     external
11.12.11.2     internal

#
# VIPS and NODES ( add below - do not delete this line )
#
```

Figure 7.1 The `/etc/hosts` file created by the Setup utility

This sample **hosts** file lists the IP addresses for the default router, the internal VLAN, and the external VLAN, and it contains placeholders for both the virtual servers and the content servers that the Link Controller will manage.

◆ **WARNING**

---

*If you have modified the **/etc/hosts** file with something other than the Setup utility, such as **vi** or **pico**, be aware that your changes may be lost when you run the Setup utility (**config**). The Setup utility overwrites the **/etc/hosts** file and **openssl.conf**, but it does not warn you before doing so.*

Using the MindTerm SSH Console

With the MindTerm SSH Console, you can open an SSH session for the Link Controller from the Configuration utility. Use the MindTerm SSH client to enable secure command line administration. You can perform any of the command line tasks in a popup console screen.

◆ **WARNING**

The MindTerm SSH client requires a Java virtual machine to operate. If you are unable to run the MindTerm SSH client, make sure that you have a Java virtual machine installed and that your browser has Java enabled in the Preferences, or Options, section. For more information on Java virtual machines and download options, visit your web browser manufacturer's web site.

To open the MindTerm SSH Console using the Configuration utility

1. In the navigation pane, click **MindTerm SSH Console**.
A popup console opens.
2. When you see the command prompt, press Enter.
3. Log in to the Link Controller as you normally would.

◆ **Note**

When you use the MindTerm SSH Console, you can only administer the local Link Controller. If you wish to administer remote systems, you do so using an SSH or Telnet session from the command line. For information about installing an SSH client on the administrative workstation, see the following section.

Downloading the SSH client to your administrative workstation

From Link Controller units that support encrypted communications, you can download the SSH client to your administrative workstation in preparation for remote command line access. In addition to running Link Controller command line utilities, you can also use the SSH suite for file transfer to and from the Link Controller, as well as for remote backups.

The SSH client is available for both Windows and UNIX platforms, and you can download your preferred client either from the web server or using an FTP connection. You can find detailed information about the SSH client in the documentation provided on the web server or on the Documentation and Software CD-ROM.

◆ Note

If your Link Controller does not support encrypted connections, you can use a Telnet shell for remote command line access.

Downloading the SSH client from the web server

Connect to the Link Controller using **https://** rather than **http://** in the URL. In the Additional Software Downloads section, click the **SSH Clients** link. From the SSH Clients page, you can choose the SSH Client appropriate to your operating system.

Setting up the SSH client on a Windows 95 or Windows NT workstation

The SSH client installation file for Windows platforms is compressed in ZIP format. You can use standard ZIP tools, such as PKZip or WinZip to extract the file.

To unzip and install the SSH client

1. Log on to the Windows workstation.
2. Navigate to the directory to which you transferred the installation file. Run **PKZip** or **WinZip** to extract the files.
3. The set of files extracted includes a Setup program. Run the Setup program to install the client.
4. Start the SSH client.
5. In the SSH Client window, from the Edit menu choose **Properties**. The Properties dialog box opens.
6. In the Connection tab, in the Remote Host section, type the following items:

- In the **Host Name** box, type the Link Controller IP address or host name.
  - In the **User Name** box, type the root user name.
7. In the Options section, check **Compression** and set the Cipher option to **Blowfish**.
  8. Click the **OK** button.

Setting up the SSH client on a UNIX workstation

The installation file for UNIX platforms is compressed in **tar/gzip** format.

To untar and install the SSH client

1. Log on to the workstation and navigate to the directory into which you transferred the SSH client tar file.
2. Untar the file and follow the instructions in the **install** file to build the SSH client for your workstation.
3. Start the SSH client.
4. Open a connection to the Link Controller:

```
ssh -l root [Link Controller IP address]
```
5. Type the root password and press the Enter key.

Addressing general networking issues

You must address several network issues when you place a Link Controller in your network. These networking issues include routing, DNS configuration, and special e-mail considerations. You need to address these issues based on the type of hardware and software in your network. This section describes the following networking issues:

◆ Addressing routing issues

There are a variety of routing configuration issues that you need to address. If you did not create a default route with the Setup utility, you must now configure a default route for the Link Controller. You also must set up routes for the nodes that the Link Controller manages. You may also want to configure GateD, which allows dynamic routing information to automatically be updated on the Link Controller.

◆ Configuring DNS on the Link Controller

You may need to configure the Link Controller for DNS resolution or for DNS proxy, and you may even need to convert from rotary or round robin DNS.

◆ Configuring email on the Link Controller

There are some special requirements that you need to take into account when configuring email on the Link Controller.

Addressing routing issues

The Link Controller must communicate properly with network routers, as well as with the servers, firewalls, and other routers that it manages. Because there are a variety of router configurations, and varying levels of direct control an administrator has over each router, you need to carefully review the router configurations in your own network. You may need to change some routing configurations before you put the Link Controller into production.

The Link Controller supports static route configurations, dynamic routing (by way of BGP4, RIP1, RIP2, and OSPF), and subnetting. However, the Link Controller is also designed to eliminate the need for you to modify routing tables on a router that routes to a Link Controller. Instead, the Link Controller uses Address Resolution Protocol (ARP) to notify routers of the IP addresses that it uses on each interface, as well as on its virtual servers.

The following sections address these common routing issues:

- Routing from a Link Controller to a gateway to the external network
- Routing from content servers to the Link Controller
- Routing between a Link Controller to content servers that are on different logical networks
- Setting up dynamic routing with GateD
- Configuring static routes in **/config/routes**

Routing from a Link Controller to a gateway to the external network

The Link Controller needs a route to the external network. For most configurations, this should be configured as the default gateway pool on the Link Controller.

During installation, you were prompted to configure a default route for the Link Controller. If you need to change the default route at this time, you can set a new default route by editing the default gateway pool.

To change the default route from the Setup utility

1. From the command line, type **setup**.
The Setup utility menu opens.
2. Choose the Default Gateway Pool option.
3. Type the IP address of the gateway you want to add to the default gateway pool.
4. Save and exit.

To change the default route using the Configuration utility

1. In the navigation pane, click **System**.
The System Properties screen opens.

2. Click the System tab.
Look in the **Default Gateway Pool** list for the name of the default gateway pool. Make sure you have the pool name before proceeding to step 3.
3. In the navigation pane, click Pools.
The Pools screen opens.
4. In the list of pools, click the name of the default gateway pool.
The pool properties page for that pool opens.
5. In the Resources section of the screen, add or remove gateway IP addresses.
6. Click the **Apply** button.

Routing from content servers to the Link Controller

The content servers being load balanced by the Link Controller need to have a default route set to the internal shared floating IP alias of the Link Controller. For most configurations, this should be configured as the default route on the content server.

For information about setting the default route for your content servers, refer to the product documentation for your server.

Routing between a Link Controller and content servers on different logical networks

If you need to configure the Link Controller to use one or more nodes that actually sit on a different logical network from the Link Controller, you need to assign one or more additional routes to get to those nodes. Set each node's default route so that traffic goes back through the Link Controller internal interface.

In the following examples, the nodes are on **192.168.6.0/24** and the Link Controller internal interface is on **192.168.5.0/24**. There are two possible situations which you may have to address:

- **192.168.5.0/24** and **192.168.6.0/24** are on the same LAN (either sharing media or with a switch or hub between them).
- **192.168.5.0/24** and **192.168.6.0/24** are on two different LANs with a router between them.

Case 1: Same LAN

If the nodes are on the same LAN as the Link Controller, you simply need to add an interface route for **192.168.6.0/24** to an interface on the internal network. You can add this route to the bottom of the **/etc/rc.local** file using the following syntax, where **<ip addr>** is the IP address on the internal interface:


```
route add -net 192.168.6 -interface <ip addr>
```

◆ Note

*You must have the interface defined correctly in the **/etc/hosts** file in order to use this syntax.*

Case 2: Different LANs

If you have nodes on different LANs from the Link Controller, you need to add a static gateway route on the Link Controller itself. If, for example, the router that connects the **192.168.5** network and the **192.168.6** network has IP addresses **192.168.5.254** and **192.168.6.254**, then you could use the following command to create the necessary static route on the Link Controller:

```
route add -net 192.168.6.0 -gateway 192.168.5.254
```

You should add this command to the end of the file **/etc/netstart** so that it runs each time the Link Controller boots.

You may also need to set the default route on the nodes to point to the router between the LANs. For example:

```
route add default -gateway 192.168.6.254
```

Finally, you need to set the default route on the router between the LANs to the shared alias on the Link Controller. For example, type the command:

```
route add default -gateway 192.168.5.200
```

◆ Note

These examples assume you are using a UNIX-based router. The exact syntax for your router may be different.

It is not necessary to set the default route for nodes directly to the Link Controller, as long as the default path eventually routes through the Link Controller.

Setting up dynamic routing with GateD

The GateD daemon allows the Link Controller to exchange dynamic routing updates with your routers. Setting up the GateD daemon is a three-part task:

- You need to create the GateD configuration file, **/config/gated.conf**.
- You need to start the GateD daemon.
- You need to edit the **/etc/netstart** file.

◆ Tip

You are not required to configure GateD on the Link Controller. The Link Controller can meet most routing requirements without using GateD.

◆ Note

Additional documentation for GateD is available through the web server on the Link Controller.

To create the GateD configuration file

GateD relies on a configuration file, typically named `/config/gated.conf`, which can be relatively simple, or can be very complex, depending on the routing needs of your network. The Link Controller web server includes the GateD online documentation (in the Configuration utility home screen, under the **Online Documentation** section, click **GateD**). Note that the GateD configuration guide details the process of creating the GateD configuration file, and also provides samples of common protocol configurations.

To immediately start the GateD daemon on the Link Controller

Once you create the GateD configuration file, you need to start the GateD daemon on the command line using the following command:

```
bigip# gated
```

Configuring static routes in `/config/routes`

You can create the file `/config/routes` on the Link Controller for configuring static route information. The information you add to `/config/routes` is synchronized between units in a Link Controller redundant pair. When you upgrade, the route information is saved and reinstalled when the upgrade is complete.

You can add routes to `/config/routes` using the format in Figure 7.2.

```
route add -net 10.1.10.0 -netmask 255.255.255.0 -gateway 10.1.30.254
route add -net 10.1.20.0 -netmask 255.255.255.0 -gateway 10.1.30.254
```

Figure 7.2 Example entries in `/config/routes`

Configuring DNS on the Link Controller

If you plan to use DNS in your network, you can configure DNS on the Link Controller. There are three different DNS issues that you may need to address when setting up the Link Controller:

- Configuring DNS resolution on the Link Controller
- Configuring DNS proxy
- Converting from rotary or round robin DNS

Configuring DNS resolution

When entering virtual addresses, node addresses, or any other addresses on the Link Controller, you can use the address, host name, or fully qualified domain name (FQDN).

The Link Controller looks up host names and FQDNs in the `/etc/hosts` file. If it does not find an entry in that file, then it uses DNS to look up the address. In order for this to work, you need to create an `/etc/resolv.conf` file. The file should have the following format:

```
nameserver <DNS_SERVER_1>
search <DOMAIN_NAME_1> <DOMAIN_NAME_2>
```

In place of the `<DNS_SERVER_1>` parameter, use the IP address of a properly configured name server that has access to the Internet. You can specify additional name servers as backups by inserting an additional **nameserver** line for each backup name server.

If you configure the Link Controller itself as a DNS proxy server, then we suggest that you choose its loopback address (**127.0.0.1**) as the first name server in the `/etc/resolv.conf` file.

Replace the `<DOMAIN_NAME_1>` and `<DOMAIN_NAME_2>` parameters with a list of domain names to use as defaults. The DNS uses this list to resolve hosts when the connection uses only a host name, and not an FQDN. When you enter domain names in this file, separate each domain name with a space, as shown in Figure 7.3.

```
; example /etc/resolv.conf
nameserver 127.0.0.1
nameserver 127.16.112.2 ;ip address of main DNS server
search mysite.com store.mysite.com
```

Figure 7.3 Sample `/etc/resolv.conf` file

You can also configure the order in which name resolution checks are made by configuring the `/etc/irs.conf` file. You should set this file so that it checks the `/etc/hosts` file first, and then checks for DNS entries. See Figure 7.4, for an example of how to make the entry in the `/etc/irs.conf` file.

hosts	local	continue
hosts	dns	

Figure 7.4 Sample entry for the `/etc/irs.conf` file

Configuring DNS proxy

The Link Controller is automatically configured as a DNS proxy or forwarder. This is useful for providing DNS resolution for servers and other equipment load balanced by the Link Controller. This can be set in the Setup utility.

To re-configure DNS proxy, you simply edit the `/etc/named.boot` file that contains these two lines:

forwarders <DNS_SERVERS>

options forward-only

In place of the <DNS_SERVERS> parameter, use the IP addresses of one or more properly configured name servers that have access to the Internet.

You can also configure the Link Controller to be an authoritative name server for one or more domains. This is useful when DNS is needed in conjunction with internal domain names and network addresses for the servers and other equipment behind the Link Controller. Refer to the BIND documentation for more details.

Converting from rotary or round robin DNS

If your network is currently configured to use rotary DNS, your node configuration may not need modification. However, you need to modify your DNS zone tables to map to a single IP address instead of to multiple IP addresses.

For example, if you had two Web sites with domain names of **www.SiteOne.com** and **www.SiteTwo.com**, and used rotary DNS to cycle between two servers for each Web site, your zone table might look like the one in Figure 7.5.

www.SiteOne.com	IN A 192.168.1.1
	IN A 192.168.1.2
www.SiteTwo.com	IN A 192.168.1.3
	IN A 192.168.1.4

Figure 7.5 Sample zone table with two Web sites and four servers

In the Link Controller configuration, the IP address of each individual node used in the original zone table becomes hidden from the Internet. We recommend that you use the Internet reserved address range as specified by RFC 1918 for your nodes. In place of multiple addresses, simply use a single virtual server associated with your site's domain name.

Using the above example, the DNS zone table might look like the zone table shown in Figure 7.6.

www.SiteOne.com	IN A 192.168.100.231
www.SiteTwo.com	IN A 192.168.100.232

Figure 7.6 Sample zone table with two Web sites and two servers.

Configuring email

Another optional feature you can set up when you configure the Link Controller is email. You can configure the Link Controller to send email notifications to you, or to other administrators. The Link Controller uses Sendmail as its mail transfer agent. The Link Controller includes a sample

Sendmail configuration file that you can use to start with, but you will have to customize the Sendmail setup for your network environment before you can use it.

Before you begin setting up Sendmail, you may need to look up the name of the mail exchanger for your domain. If you already know the name of the mail exchanger, continue with the following section, *Setting up Sendmail*.

Setting up Sendmail

When you actually set up Sendmail, you need to open and edit a couple of configuration files. Note that the Link Controller does not accept email messages, and that you can use the **crontab** utility to purge unsent or returned messages, and that you can send those messages to yourself or another administrator.

To set up and start Sendmail

1. Copy **/config/sendmail.cf.off** to **/config/sendmail.cf**.
2. To set the name of your mail exchange server, open the **/config/sendmail.cf** and set the DS variable to the name of your mail exchanger. The syntax for this entry is:
DS<MAILHUB_OR_RELAY>
3. Save and close the **/config/sendmail.cf** file.
4. If you want to allow Sendmail to flush outgoing messages from the queue for mail that cannot be delivered immediately:
  - a) Open the **/config/crontab** file, and change the last line of the file to read:
0,15,30,45 * * * * root /usr/sbin/sendmail -q > /dev/null 2>&1
  - b) Save and close the **/config/crontab** file.
5. If you want to prevent returned or undelivered email from going unnoticed:
  - a) Open the **/config/aliases** file and create an entry for **root** to point to you or another administrator at your site:
root: networkadmin@SiteOne.com
  - b) Save and close the **/config/aliases** file.
  - c) Run the **newaliases** command to generate a new aliases database that incorporates the information you added to the **/config/aliases** file.
6. To turn Sendmail on, either reboot the system or type the following command:
/usr/sbin/sendmail -bd -q30m

Using a serial terminal with the Link Controller

There are a couple of different ways to add a serial terminal to the Link Controller. You can add a serial terminal in addition to the console, or you can add a serial terminal as the console. The difference between the two is:

- ◆ A serial terminal configured as a terminal displays a simple login. You can log in and run commands and edit files. In this case, you can use the serial terminal in addition to the keyboard and monitor.
- ◆ A serial terminal configured as the console displays system messages and warnings in addition to providing a login prompt. In this case, the serial terminal replaces the keyboard and monitor.

To connect the serial terminal to the Link Controller

Connect a serial line cable between the terminal device and the Link Controller. On the back of Link Controller is a male, 9-Pin RS232C connector labeled **Terminal**. (Be sure not to confuse this with the fail-over connection which is also a male, 9-pin connector.)

◆ WARNING

Do not use the fail-over cable to connect the serial terminal to the Link Controller. A null modem cable is required

The connector is wired as a DTE device, and uses the signals described in Table 7.1.

Pin	Source	Usage
1	External	Carrier detect
2	External	Received data
3	Internal	Transmitted data
4	Internal	Data terminal ready
5	Both	Signal ground
7	Internal	Request to send
8	External	Clear to send

Table 7.1 Serial line cable signals

The connector is wired for direct connection to a modem, with receipt of a Carrier Detect signal generating transmission of a login prompt by the Link Controller. If you are planning to connect to a terminal or to connect a PC

and utilize a terminal emulation program such as HyperTerminal™, you need a null modem cable with the wiring to generate the signals shown in Table 7.1.

◆ **Note**

*You can achieve acceptable operation by wiring pin 7 to 8, and pin 1 to 4 on the Link Controller (and turning hardware flow control **off** in your terminal or terminal emulator).*

Configuring a serial terminal in addition to the console

You can configure a serial terminal for the Link Controller in addition to the standard console.

To configure the serial terminal in addition to the console

1. Connect the serial terminal to the Link Controller.
2. Configure the serial terminal settings in your terminal or terminal emulator or modem as follows:
  - 9600 baud
  - 8 bits
  - 1 stop bit
  - No parity
3. Open the `/etc/ttys` file and find the line that reads **tty00 off**. Modify it as shown here:

```
# PC COM ports (tty00 is DOS COM1)
tty00 "/usr/libexec/getty default" vt100 in secure
```
4. Save the `/etc/ttys` file and close it.
5. Reboot the Link Controller.

Configuring a serial terminal as the console

You can configure the serial terminal as the console.

To configure the serial terminal as the console

1. Disconnect the keyboard from the Link Controller.
2. Connect the serial terminal to the Link Controller. When there is no keyboard connected to the Link Controller, the Link Controller defaults to using the serial port for the console.
3. Configure the serial terminal settings in your terminal or terminal emulator or modem as follows:
  - 9600 baud

- 8 bits
  - 1 stop bit
  - No parity
4. Reboot the Link Controller.

Forcing a serial terminal to be the console

In the case where you have not yet connected the serial terminal or it is not active when the Link Controller is booted, as it might be if you are using a terminal server or dial-up modem, you can force the controller to use the serial terminal as a console. Note that you do not need to disconnect the keyboard if you use this procedure to force the serial line to be the console.

To force a serial terminal to be the console

1. Edit the `/etc/boot.default` file.
Find the entry `-console auto`. Change this entry to `-console com`.
2. Save the `/etc/boot.default` file and exit the editor.
3. Plug the serial terminal into the serial port on the Link Controller.
4. Turn on the serial terminal.
5. Reboot the Link Controller.

WARNING

Once you configure a serial terminal as the console for the Link Controller, the following conditions apply:

Keyboard/monitor access is disabled, and logging in is only possible through Secure Telnet (SSH), if configured, or the serial line.

If the `boot.default` file is corrupted, the system does not boot at all. Save a backup copy of the original file and keep a bootable CD-ROM on hand.

The `boot.default` file must contain either the line: `-console com` or the line: `-console auto`. Do not configure both settings. This could cause problems when you attempt to boot the system.

Configuring RADIUS or LDAP authentication

You can configure the Link Controller to use a RADIUS or LDAP server on your network to authenticate users attempting to access the controller with SSH. In this configuration, the RADIUS or LDAP server can function as a central repository of users that are allowed access to the Link Controller for administrative purposes.

Configuring RADIUS login support

Follow these steps to enable RADIUS authentication on Link Controller.

To configure RADIUS login support

1. Create the directory **/etc/raddb**.
2. Create the file **/etc/raddb/servers**. Each line should contain the host name of the radius server to connect to, and the secret used by that server (see the following example). For security reasons, we recommend that you use IP addresses instead of host names for the entries in this file. If you specify a host name for an entry, we recommend that you add the host name to the **/etc/hosts** file.

```
# this is the /etc/raddb/server file
# format is <radius server> <secret>
radius.test.net testing123
```

3. Edit the **/etc/login.conf** file. Locate these lines at the top of the file. Replace **my_radius_server** with the host name of your RADIUS server. The host name you specify must also exist in the **/etc/raddb/servers** file you created in step 2 (see the following example).

```
radius-defaults:auth=passwd:\
:auth-ssh=radius,passwd:\
:radius-server=my_radius_server:
```

4. Change the default configuration to include the **radius-default** section like this (see the following example):

```
default:\
:path=/bin /usr/bin /usr/contrib/bin:\
:datasize-cur=16M:\
:tc=radius-defaults:
```

5. Before logging out, test the configuration by using SSH to connect to the Link Controller. That way you can correct any configuration errors which could prevent you from logging in to the Link Controller.

Configuring LDAP login support

To configure the Link Controller for LDAP authentication, you need to modify the `/etc/login.conf` file. You can configure LDAP authentication on the Link Controller with LDAP servers that store passwords in encrypted or hashed format, or you can configure the Link Controller to handle LDAP servers that use plain text passwords.

To configure an LDAP server that stores encrypted passwords

In some LDAP servers, passwords are stored encrypted with DES, or stored as MD5 hashes. On these systems, it is best to bind to the server directly in order to let the LDAP server match the passwords. The `login_ldap` utility can be configured to bind directly to the server with the following settings in the `/etc/login.conf` file.

1. Edit `/etc/login.conf`. Locate these lines at the top of the file. Replace `my_ldap_server` with the host name of your LDAP server. Replace the value for `ldap-basedn` with the appropriate `basedn` for your LDAP server (see the following example).

```
ldap-defaults:auth=passwd:\
:auth-ssh=ldap,passwd:\
:ldap-server=my_ldap_server:\
:ldap-server-user=cn=Manager,dc=test,dc=net:\
:ldap-basedn=dc=test,dc=net:\
:ldap-user-bind=yes:
```

2. Locate the default authentication type. Change the `tc` value to point to the new `ldap-defaults` type (see the following example).

```
default:\
:path=/bin /usr/bin /usr/contrib/bin:\
:datasize-cur=16M:\
:tc=ldap-defaults:
```

To configure an LDAP server that stores plain text passwords

Other LDAP servers store user passwords in plain text. Because of this, these servers require the root LDAP user to log in to see these users. Use these instructions to configure Link Controller to authenticate to the server with the root user identity before each user authentication.

1. Edit the `/etc/login.conf` file. Locate the lines at the top of the file after the `auth-bsd-defaults` type. Replace `my-ldap-server` with the values from your LDAP configuration. Change `ldap-user-bind` to `no`. The `ldap-sever-user` may not be required by your configuration. If it is not, remove that line (see the following example).

```
ldap-defaults:auth=passwd:\
:auth-ssh=ldap,passwd:\
:ldap-server=my_ldap_server:\
:ldap-server-user=cn=Manager,dc=test,dc=net:\
:ldap-basedn=dc=test,dc=net:\
:ldap-user-bind=no:
```

2. Locate the **default** authentication type. Change the **tc** value to point to the new **ldap-defaults** type (see the following example).

```
default:\
:path=/bin /usr/bin /usr/contrib/bin:\
:datasize-cur=16M:\
:tc=ldap-defaults:
```

3. Create the file **/etc/ldapb/servers**. Add one line for the host name of the LDAP server to connect to, and the secret used by that server user (see the following example). For security reasons, we recommend that you use IP addresses instead of host names for the entries in this file. If you specify a host name for an entry, we recommend that you add the host name to the **/etc/hosts** file.

```
# this is the /etc/ldapdb/server file
# format is <ldap server> <secret>
ldap.test.net secret
```

4. Before logging out, test the configuration by running the login program, either on a virtual console or using Telnet. That way you can correct any configuration errors before that may prevent you from accessing the Link Controller.

Allowing multiple authentication styles

We recommend that you allow multiple authentication styles. This allows you to log in even if the LDAP or RADIUS server is not working properly. You can specify multiple authentication styles in the **auth** field such as **radius,passwd**. With the example **radius,passwd**, RADIUS is the default authentication style, but you still have the ability to override the style and force a login using the password file by appending **:passwd** after the login name. This is useful because you need to be able to log in even if the authentication server is down (or if its name gets changed and the **/etc/login.conf** file needs to be updated).

```
# login root:passwd
```

or

```
# ssh bigip -l "root:passwd"
```

Only the styles that you specify are accepted. For example, **root:ldap** would fail, since that style was not specified.

Requiring different authentication styles for different applications

You can configure the Link Controller authentication system to require different authentication styles for different applications. The following example (see Figure 7.7) would use password authentication by default (at the console), but would require RADIUS for FTP and LDAP for SSH, and would accept RADIUS, LDAP, or network password logins (such as Telnet).

```
my-defaults:\
:auth-ftp=radius:\
:radius-server=<my_radius_server>:\
:auth-ssh=ldap:\
:ldap-server=<my_ldap_server>:\
:ldap-basedn=ou=People,dc=<f5>,dc=<com>:\
:auth-network=radius, ldap, passwd:\
:auth=passwd:
default:\
:path=/bin /usr/bin /usr/contrib/bin:\
:datsize-cur=16M:\
:tc=my-defaults:
```

Figure 7.7 Example of password authentication with RADIUS required for other applications

---

◆ **Note**

RADIUS authentication through the Link Controller is based on the user name/password only. It does not support challenge-response authentication methods.



8

Configuring a Redundant System

- Overview of a redundant system
- Synchronizing configurations between units
- Configuring fail-safe settings
- Mirroring connection information
- Using gateway fail-safe
- Using network-based fail-over
- Setting a specific Link Controller to be the preferred active unit

Overview of a redundant system

A Link Controller redundant system consists of two identically configured Link Controller units, only one of which is active at a given time (unless a special active-active configuration is chosen). The inactive unit serves as a standby which becomes active only in case of failure of the active system, a process called *failover*.

Link Controller redundant systems have special settings that you need to configure, such as VLAN fail-safe settings. One convenient aspect of configuring a redundant system is that once you have configured one of the Link Controller units, you can simply copy the configuration to the other Link Controller in the system by using the configuration synchronization feature.

There are two basic aspects about working with redundant systems:

- Synchronizing configurations between two Link Controller units
- Configuring fail-safe settings for the VLANs

In addition to the simple redundant features available on the Link Controller, several advanced redundant features are available. Advanced redundant system features provide additional assurance that your content is available if a Link Controller experiences a problem. These advanced redundant system options include:

- Mirroring connection and persistence information
- Gateway fail-safe
- Network-based fail-over
- Setting a specific Link Controller to be the active unit

The attributes you can configure for redundant systems are shown in Table 8.1.

Attributes	Description
Synchronizing configurations	This feature allows you to configure one Link Controller and then synchronize the configuration with the other Link Controller.
Fail-safe for VLANs	Fail-safe for VLANs provides the ability to cause a Link Controller to fail over if a VLAN is no longer generating traffic.
Mirroring connections and persistence information	You can mirror connection and/or persistence information between redundant units. This enables you to provide seamless fail-over of client connections.
Gateway fail-safe	This feature allows you to fail-over between two gateway routers.
Network-based fail-over	You can configure the Link Controller to use the network to determine the status of the active unit.

Table 8.1 *The attributes you can configure for redundant systems*

Attributes	Description
Setting a dominant Link Controller	You can set up one unit in a pair to be the dominant active Link Controller. The unit you set up as the dominant Link Controller will always attempt to be active.
Active-active configuration	The default mode for a Link Controller redundant system is Active/Standby. However, you can configure both units to run in active mode.

Table 8.1 The attributes you can configure for redundant systems

Synchronizing configurations between units

Once you complete the initial configuration on the first unit in the system, you can synchronize the configurations between the active unit and the standby unit. When you synchronize a configuration, the following configuration files are copied to the other Link Controller:

- The common **BIG/db** keys
- All files in **/config** (except **bigip_base.conf**)

If you use command line utilities to set configuration options, be sure to save the current configuration to the file before you use the configuration synchronization feature. (Alternately, if you want to test the memory version on the standby unit first, use **bigpipe config sync running**.)

Use the following **bigpipe** command to save the current configuration:

```
b save
```

◆ Note

*The Link Controller software creates a file named **/usr/local/ucs/cs_backup.ucs** prior to installing a configuration file (UCS) from a remote machine.*

To synchronize the configuration using the Configuration utility

1. In the navigation pane, click **System**.
The Network Map screen opens.
2. Click the Redundant Properties tab.
The Redundant Properties screen opens.
3. Click the **Synchronize Configuration** button.

To synchronize the configuration from the command line

Synchronize the configuration from the command line using the **bigpipe config sync** command. Use the **bigpipe config sync** command without the **all** option to synchronize only the boot configuration file **/config/bigip.conf**.

The **bigpipe config sync all** command synchronizes the following configuration files:

- The common **BIG/db** keys
- All files in **/config** (except **bigip_base.conf**)

The **bigpipe config sync running** command synchronizes the running version of **/config/bigip.conf**, which is the image that resides in memory as the system runs. This file is written only to memory on the standby unit, it is not saved.

Configuring fail-safe settings

For maximum reliability, the Link Controller supports failure detection on both internal and external VLANs. When you arm the fail-safe option on a VLAN, the Link Controller monitors network traffic going through the VLAN. If the Link Controller detects a loss of traffic on a VLAN when half of the fail-safe timeout has elapsed, it attempts to generate traffic. A VLAN attempts to generate network traffic by issuing ARP requests to nodes accessible through the VLAN. Also, an ARP request is generated for the default route if the default router is accessible from the VLAN. Any traffic through the VLAN, including a response to the ARP requests, averts a fail-over.

If the Link Controller does not receive traffic on the VLAN before the timer expires, it initiates a fail-over, switches control to the standby unit, and reboots.

WARNING

You should arm the fail-safe option on a VLAN only after the Link Controller is in a stable production environment. Otherwise, routine network changes may cause fail-over unnecessarily.

Arming or disarming fail-safe on a VLAN

Each interface card installed on the Link Controller is typically mapped to a different VLAN, which you need to know when you set the fail-safe option on a particular VLAN. You can view VLAN names in the Configuration utility, or you can use the **bigpipe vlan show** command to view VLAN names from the command line.

To arm or disarm fail-safe on an interface using the Configuration utility

1. In the navigation pane, click **Network**.
The VLANs list opens and displays all VLANs.
2. Select a VLAN name.
The VLAN Properties screen opens.

3. Locate the **Arm Failsafe** box:
  - To arm fail-safe, check **Arm Failsafe**.
  - To disarm fail-safe, clear the **Arm Failsafe** box.
4. If you are arming fail-safe, in the **Timeout** box, type the maximum time allowed for a loss of network traffic before a fail-over occurs.
5. Click the **Apply** button.

To arm or disarm fail-safe on a VLAN from the command line

To look up the names of the existing VLANs, use the **bigpipe vlan** command with the **show** keyword:

```
b vlan show
```

To arm fail-safe on a particular VLAN, use the **bigpipe vlan** command with the **timeout** and **failsafe arm** keywords:

```
b vlan <vlan_name> timeout <seconds>
```

```
b vlan <vlan_name> failsafe arm
```

For example, you have an external VLAN named **vlan1** and an internal VLAN named **vlan2**. To arm the fail-safe option on both cards with a timeout of 30 seconds, you need to issue the following commands:

```
b vlan vlan1 timeout 30
```

```
b vlan vlan2 timeout 30
```

```
b vlan vlan1 failsafe arm
```

```
b vlan vlan2 failsafe arm
```

To disarm fail-safe on a particular VLAN, use the **bigpipe vlan** command with the **failsafe arm** keyword:

```
b vlan <vlan_name> failsafe disarm
```

Mirroring connection information

When the fail-over process puts the active Link Controller duties onto a standby unit, the connection capability of your site returns so quickly that you have little chance to see it. By preparing a redundant system for the possibility of fail-over, you effectively maintain your site's reliability and availability in advance. But fail-over alone is not enough to preserve the connections and transactions on your servers at the moment of fail-over; they would be dropped as the active unit goes down unless you have enabled mirroring.

The **mirror** feature on Link Controller units is the ongoing communication between the active and standby units that duplicates the active unit's real-time connection information state on the standby unit. If you have enabled mirroring, fail-over can be so seamless that file transfers can

proceed uninterrupted, customers making orders can complete transactions without interruption, and your servers can generally continue with whatever they were doing at the time of fail-over.

The mirror feature is intended for use with long-lived connections, such as FTP, Chat, and Telnet sessions. Mirroring is also effective for HTTP persistence connections.

◆ WARNING

If you attempt to mirror all connections, it may degrade the performance of the Link Controller.

Commands for mirroring

Table 8.2 contains the commands that support mirroring capabilities. For complete descriptions, syntax, and usage examples, see Appendix A, *bigpipe Command Reference*.

bigpipe command	Options
b global mirror	Options for global mirroring
b virtual mirror	Options for mirroring connection and persistence information on a virtual server
b snat mirror	Options for mirroring secure NAT connections

Table 8.2 *Mirroring commands in bigpipe*

To configure global mirroring

You must enable mirroring on a redundant system at the global level before you can set mirroring of any specific types of connections or information. However, you can set specific types of mirroring and then enable global mirroring to begin mirroring. The syntax of the command for setting global mirroring is:

b global mirror enable | disable | show

To enable mirroring on a redundant system, use the following command:

b global mirror enable

To disable mirroring on a redundant system, use the following command:

b global mirror disable

To show the current status of mirroring on a redundant system, use the following command:

b global mirror show

Mirroring virtual server state

Mirroring provides seamless recovery for current connections when a Link Controller fails. When you use the mirroring feature, the standby Link Controller maintains the same state information as the active unit. Transactions such as FTP file transfers continue as though uninterrupted.

Since mirroring is not intended to be used for all connections, it must be specifically enabled for each virtual server.

◆ Note

Mirroring cannot be used with SSL gateways.

To control mirroring for a virtual server, use the **bigpipe virtual mirror** command to enable or disable mirroring of persistence information, or connections, or both. The syntax of the command is:

```
b virtual <virt addr>:<service> \  
mirror [conn] enable | disable
```

Use **conn** to mirror connection information for the virtual server. To display the current mirroring setting for a virtual server, use the following syntax:

```
b virtual <virt addr>:<service> \  
mirror [conn] show
```

If you do not specify **conn** for connection information, the Link Controller assumes that you want to display this type of information.

Mirroring SNAT connections

SNAT connections are mirrored only if specifically enabled. You can enable SNAT connection mirroring by specific node address, and also by enabling mirroring on the default SNAT address. Use the following syntax to enable SNAT connection mirroring on a specific address:

```
b snat <node addr> [...<node addr>] mirror enable | disable
```

In the following example, the **enable** option turns on SNAT connection mirroring to the standby unit for SNAT connections originating from **192.168.225.100**.

```
b snat 192.168.225.100 mirror enable
```

Use the following syntax to enable SNAT connection mirroring the default SNAT address:

```
b snat default mirror enable | disable
```

Using gateway fail-safe

Fail-safe features on the Link Controller provide network failure detection based on network traffic. Gateway fail-safe monitors traffic between the active Link Controller and the gateway router, protecting the system from a loss of the internet connection by triggering a fail-over when the gateway is unreachable for a specified duration.

You can configure gateway fail-safe in the Configuration utility or in BIG/db. If you configure gateway fail-safe in BIG/db, you can toggle it on and off with **bigpipe** commands.

Adding a gateway fail-safe check

When you set up a gateway fail-safe check using the Configuration utility, you need to provide the following information:

- Name or IP address of the router (only one gateway can be configured for fail-safe)
- Time interval (seconds) between pings sent to the router
- Time-out period (seconds) to wait for replies before proceeding with fail-over

---

◆ **Note**

We recommend a gateway failsafe ping interval of 2 seconds with a timeout of 10 seconds. If this interval is too small, you can use any 1 to 5 ratio that works for you.

To configure gateway fail-safe using the Configuration utility

1. In the navigation pane, click **System**.
The Network map screen opens.
2. Click the Redundant Properties tab.
The Redundant Properties screen opens.
3. In the Gateway Fail-safe section of the screen, make the following entries:
  - Check the **Enabled** box.
  - In the **Router** box, type the IP address of the router you want to ping.
  - In the **Ping (seconds)** box, type the number of seconds you want the Link Controller to wait before it pings the router.
  - In the **Timeout (seconds)** box, type the timeout value, in seconds. If the router does not respond to the ping within the number of seconds specified, the gateway is marked **down**.
4. Click the **Apply** button.

To configure gateway fail-safe in BIG/db

To enable gateway fail-safe in BIG/db, you need to change the settings of three specific BIG/db database keys using the **bigpipe db** command. The keys set the following values:

- The IP address of the router
- The ping interval
- The timeout period

To set the IP address of the router, type the following entry, where **<gateway IP>** is the IP address, or host name, of the router you want to ping:

```
b db set Local.Bigip.GatewayPinger.Ipaddr=<gateway IP>
```

To set the ping interval, type the following entry, where **<seconds>** is the number of seconds you want the Link Controller to wait before pinging the router:

```
b db set Local.Bigip.GatewayPinger.Pinginterval=<seconds>
```

To set the timeout, type the following entry, where **<seconds>** is the number of seconds you want the Link Controller to wait before marking the router **down**:

```
b db set Local.Bigip.GatewayPinger.Timeout=<seconds>
```

After you make these changes, you must restart **bigd** to activate the gateway pinger:

```
bigstart reinit bigd
```

To enable gateway fail-safe from the command line

You can toggle Gateway fail-safe monitoring **on** or **off** from the command line using the **bigpipe gateway** command.

For example, arm the gateway fail-safe using the following command:

```
b global gateway failsafe arm
```

To disarm fail-safe on the gateway, enter the following command:

```
b global gateway failsafe disarm
```

To see the current fail-safe status for the gateway, enter the following command:

```
b global gateway failsafe show
```

Finding gateway fail-safe messages

The destination for gateway fail-safe messages is set in the standard **syslog** configuration (**/etc/syslog.conf**), which directs these messages to the file **/var/log/bigd**. Each message is also written to the Link Controller console (**/dev/console**).

Using network-based fail-over

Network-based fail-over allows you to configure a redundant Link Controller system to use the network to determine the status of the active unit. Network-based fail-over can be used in addition to, or instead of, hard-wired fail-over.

To configure network-based fail-over using the Configuration utility

1. In the navigation pane, click **System**.
The Network Map screen opens.
2. Click the Redundant Properties tab.
The Redundant Properties screen opens.
3. Check the **Network Failover Enabled** box.
4. Click the **Apply** button.

To configure network-based fail-over in BIG/db

To enable network-based fail-over, you need to change the settings of specific BIG/db database keys using the **bigpipe db** command. To enable network-based fail-over, the **Common.Sys.Failover.Network** key must be set to one (1). To set this value to one, type:

```
b db set Common.Sys.Failover.Network=1
b failover init
```

Other keys are available to lengthen the delay to detect the fail-over condition on the standby unit, and to lengthen the heart beat interval from the active unit. The default number of seconds required for the standby unit to notice a failure in the active unit is **3** seconds. To change the default setting, use the following syntax:

```
b db set Common.Bigip.Cluster.StandbyTimeoutSec=<value>
b failover init
```

The default heart beat interval is **1** second. To change it from the active Link Controller, change the following value using **b db**:

```
b db set Common.Bigip.Cluster.ActiveKeepAliveSec=<value>
b failover init
```

Setting a specific Link Controller to be the preferred active unit

Setting a preferred active unit means overlaying the basic behavior of a Link Controller with a preference toward being active. A Link Controller that is set as the active unit becomes active whenever the two units negotiate for active status.

To clarify how this differs from default behavior, contrast the basic behavior of a Link Controller in the following description. Each of the two Link Controller units in a redundant system has a built-in tendency to try to become the active unit. Each unit attempts to become the active unit at boot time; if you boot two Link Controller units at the same time, the one that becomes the active unit is the one that boots up first. In a redundant configuration, if the Link Controller units are not configured with a preference for being the active or standby unit, either unit can become the active unit by becoming active first.

The active or standby preference for the Link Controller is defined by setting the appropriate startup parameters for the fail-over mechanism in BIG/db.

To force a Link Controller to active or standby state

The following example shows how to set the Link Controller to standby:

```
b db set Local.Bigip.Failover.ForceStandby  
b failover init
```

A Link Controller that prefers to be standby can still become the active unit if it does not detect an active unit.

This example shows how to set a Link Controller to active:

```
b db set Local.Bigip.Failover.ForceActive  
b failover init
```

A Link Controller that prefers to be active can still serve as the standby unit when it is on a live redundant system that already has an active unit. For example, if an active Link Controller that preferred to be active failed over and was taken out of service for repair, it could then go back into service as the standby unit until the next time the redundant system needed an active unit, for example, at reboot.



9

Configuring SNMP

- Introducing SNMP on the Link Controller
- Downloading the MIBs
- Configuring SNMP using the Configuration utility
- SNMP configuration files
- Configuring snmpd to send responses out of different ports or addresses

Introducing SNMP on the Link Controller

This chapter covers the management and configuration tasks for the simple network management protocol (SNMP) agent and management information bases (MIBs) available with the Link Controller.

The Link Controller SNMP agent and MIBs allow you to manage the Link Controller by configuring traps for the SNMP agent or polling the Link Controller with your standard network management station (NMS).

You can use the Configuration utility to configure the Link Controller SNMP agent to send traps to your management system. You can also set up custom traps by editing several configuration files.

You can use SNMP security options to securely manage access to information collected by the Link Controller SNMP agent, including Community names, TCP wrappers, and View Access Control Mechanism (VACM).

This chapter is divided into three parts:

- ◆ **Downloading the MIBS**
This section shows how to download the SNMP MIBs.
- ◆ **Configuring SNMP using the Configuration utility**
This section shows how to set up SNMP for a remote administrative host.
- ◆ **SNMP configuration files**
This section describes the SNMP configuration files and their syntax.
- ◆ **Configuring snmpd to respond out of different ports and addresses**
This section describes how to configure **snmpd** to respond out of different ports and addresses

Downloading the MIBs

To set up SNMP for a remote network management station, you must download and install the product-specific MIB files. For all Link Controller units there are the following product-specific MIB files:

- ◆ **LOAD-BAL-SYSTEM-MIB.txt.**
This is an enterprise MIB that contains specific information for properties associated with specific Link Controller functionality (load balancing, NATs, and SNATs).
- ◆ **UCD-SNMP-MIB.txt.**
This is an enterprise MIB that contains information and metrics about memory, disk utilization and other information regarding the Link Controller operating system. It is fully documented in RFC 1213.
- ◆ **Etherlike-MIB.txt**
This is a standard MIB which describes statistics for the collection of ethernet interfaces attached to the system. It is fully documented in RFC-2665.

- ◆ **If-MIB.txt**

This MIB supports an extended version of the **ifTable** including 64-bit counters.

- ◆ **RMON-MIB.txt**

This is a standard MIB that describes real-time and historical statistics for the ethernet systems in the interface. This MIB also allows the setting of alerts and traps based on user defined thresholds of available metrics in the system. It is fully documented in RFC 2819s.

- ◆ **rfc1525.mib**

This is a standard MIB which describes objects for managing MAC bridges based on the IEEE 802.1D-1990 standard between Local Area Network (LAN) segments. It is fully documented in RFCs 1463 and 1525.

- ◆ **RFC1611.my**

This is the DNS MIB.

- ◆ **3dns.my**

This is an enterprise MIB which describes information and properties of objects associated with the functioning of 3-DNS (for the 3-DNS module only).

You can download these files from the **Additional Software Downloads** section of the Configuration utility home page, where they appear as the following hypertext entries:

- **BIG-IPMIB**
(LOAD-BAL-SYSTEM-MIB.txt and UCD-SNMP-MIB.txt)
- **Interface MIB** (If-MIB.txt)
- **RMON MIB** (RMON-MIB.tx)
- **BRIDGE** (rfc1525.mib)

You can also download these files directly from **/usr/local/share/snmp/mibs** on the Link Controller to your remote host using **ssh** and **scp**.

Configuring SNMP using the Configuration utility

To configure SNMP for a remote network management station, you must perform the following tasks:

- ◆ **Set up client access**

Configure the Link Controller to allow administrative access to the SNMP agent.

- ◆ **Configure system information**

Set the system information variables.

- ◆ **Configure Traps**

Enable traps and specify by community, port, and sink.

All three tasks are performed using the SNMP Administration screen, shown in Figure 9.1. To access this screen, simply click **System Admin** in the navigation pane, then click the SNMP Administration tab.

Enable:		☒
Client Access	Allow List:	IP Address or Network Address: Netmask: ____ / ____ Add (>>) Remove (<<) Current List: 127.0.0.1
System Information	System Contact:	Customer Name <admin@custom
	Machine Location:	Network Closet #3
	Community String:	public
Trap Configuration	Auth Trap Enable:	☒
	Trap List:	Community: ____ Service: Sink ____ Add (>>) Remove (<<) Current List:

Figure 9.1 SNMP Administration screen

Setting up client access

To set up client access, you enable access and specify the IP or network addresses (with netmasks as required) from which the SNMP agent can accept requests. (By default, SNMP is enabled only for the Link Controller loopback interface **127.0.0.1**.)

To allow access to the SNMP agent using the Configuration utility

1. At the top of the SNMP Administration screen, check the **Enable** box to allow access to the Link Controller SNMP agent.
2. In the Client Access Allow List section, type the following information:
  - **IP Address or Network Address**
Type in an IP address or network address from which the SNMP agent can accept requests. Click the Add (>>) button to add the address to the Current List. For a network address, type in a netmask.

- **Netmask**

If you type a network address in the **IP Address or Network Address** box, type the netmask for the network address in this box.

3. Click the Add (>>) button to add the network address to the **Current List**.

Configuring system information

System information includes certain traps, passwords, and general SNMP variable names. There are three main variables:

- ◆ **System Contact name**

The System Contact is a MIB-II simple string variable defined by almost all SNMP boxes. It usually contains a user name, as well as an email address.

- ◆ **Machine Location (string)**

The Machine Location is a MIB-II variable that almost all boxes support. It is a simple string that defines the location of the box.

- ◆ **Community String**

The community string clear text password is used for basic SNMP security. This also maps to VACM groups, but for initial read/only access, it is limited to just one group.

To set system information properties using the Configuration utility

You use the System Information section of the SNMP Administration screen to set the system information properties.

1. In the **System Contact** box, type the contact name and email address for the person to contact regarding issues with this Link Controller.
2. In the **Machine Location** box, type a machine location, such as First Floor, or Building 1, that describes the physical location of the Link Controller.
3. In the **Community String** box, type a community name. The community name is a clear text password used for basic SNMP security and for grouping machines that you manage.

Configuring traps

To configure traps, you provide three pieces of information:

- ◆ **trapcommunity <community string>**

This sets the community string (password) to use for sending traps. If set, it also sends a trap upon startup: **coldStart(0)**.

- ◆ **trapport <port>**

This sets the port on which traps are sent. There must be one **trapport** line for each **trapsink** host.

- ◆ **authtrapenable <integer>**

Setting this variable to **1** enables traps to be sent for authentication warnings. Setting it to **2** disables it.

To set trap configuration properties using the Configuration utility

You use the Trap Configuration section of the SNMP Administration screen to set trap properties.

1. Check the **Auth Trap Enabled** box to allow traps to be sent for authentication warnings.
2. In the **Community** box, type the community name to which this Link Controller belongs. Traps sent from this box are sent to the management system managing this community.
3. In the **Service** box, type the service name on which the Link Controller sends traps. Traps sent from the Link Controller are sent to the management system on through this port.
4. In the **Sink** box, type the host that should be notified when a trap is sent by the Link Controller SNMP agent.
5. Click the Add (>>) button to add it to the **Current List**. To remove a trap sink from the **Current List**, click the trap sink you want to remove, and click the Remove (<<) button.
6. Click the **Apply** button.

SNMP configuration files

The SNMP options that you specify in the SNMP Administration screen are written to one or more of the following configuration file or files. If you prefer, you can configure SNMP by directly editing the appropriate files with a text editor rather than using the Configuration utility.

- ◆ **hosts.deny**

This file denies all UDP connections to the SNMP agent.

- ◆ **hosts.allow**

This file specifies which hosts are allowed to access the SNMP agent.

- ◆ **snmpd.conf**

This file configures the SNMP agent.

- ◆ **snmptrap.conf**

For the Link Controller, the configuration in **/etc/snmptrap.conf** determines which messages generate traps, and what those traps are. Edit this file only if you want to add traps.

◆ **3dns_snmptrap.conf**

For the 3-DNS Controller, the configuration in **/etc/3dns_snmptrap.conf** determines which messages generate traps and what those traps are. Edit this file only if you want to add traps.

◆ **syslog.conf**

Configure **/etc/syslog.conf** to pipe specified message types through **checktrap.pl**.

/etc/hosts.deny

This file must be present to deny by default all UDP connections to the SNMP agent. The contents of this file are as follows:

ALL : ALL

/etc/hosts.allow

The **/etc/hosts.allow** file is used to specify which hosts are allowed to access the SNMP agent. There are two ways to configure access to the SNMP agent with the **/etc/hosts.allow** file. You can type in an IP address, or list of IP addresses, that are allowed to access the SNMP agent, or you can type in a network address and mask to allow a range of addresses in a subnetwork to access the SNMP agent.

For a specific list of addresses, type in the list of addresses you want to allow to access the SNMP agent. Addresses in the list must be separated by blank space or by commas. The basic syntax is as follows:

daemon: <IP address> <IP address> <IP address>

For example, you can type the following line which sets the SNMP agent to accept connections from the IP addresses specified:

bigsnmpd: 128.95.46.5 128.95.46.6 128.95.46.7

For a range of addresses, the basic syntax is as follows, where **daemon** is the name of the daemon, and **IP/MASK** specifies the network that is allowed access. The **IP** must be a network address:

daemon: IP/MASK

For example, you might use the following line which sets the **bigsnmpd** daemon to allow connections from the **128.95.46.0/255.255.255.0** address:

bigsnmpd: 128.95.46.0/255.255.255.0

The example above allows the 254 possible hosts from the network address **128.95.46.0** to access the SNMP daemon. Additionally, you may use the keyword **ALL** to allow access for all hosts or all daemons.

◆ **Note**

192.168.1/24 CIDR syntax is not allowed.

/etc/snmpd.conf

The **/etc/snmpd.conf** file controls most of the SNMP agent. This file is used to set up and configure certain traps, passwords, and general SNMP variable names. A few of the necessary variables are listed below:

◆ **System Contact Name**

The System Contact is a MIB-II simple string variable defined by almost all SNMP boxes. It usually contains a user name, as well as an email address. This is set by the **syscontact** key.

◆ **Machine Location (string)**

The Machine Location is a MIB-II variable that almost all boxes support. It is a simple string that defines the location of the box. This is set by the **syslocation** key.

◆ **Community String**

The community string clear text password is used for basic SNMP security. This also maps to VACM groups, but for initial read/only access it is limited to only one group.

◆ **Trap Configuration**

Trap configuration is controlled by these entries in the **/etc/snmpd.conf** file:

- **trapsink <host>**

This sets the host to receive trap information. The **<host>** is an IP address.

- **trapport <port>**

This sets the port on which traps are sent. There must be one **trapport** line for each **trapsink** host.

- **trapcommunity <community string>**

This sets the community string (password) to use for sending traps. If set, it also sends a trap upon startup: **coldStart(0)**.

- **authtrapenable <integer>**

Setting this variable to **1** enables traps to be sent for authentication warnings. Setting it to **2** disables it.

- **data_cache_duration <seconds>**

This is the time in seconds during which data is cached. The default value for this setting is one second.

◆ **Note**

---

*A **trapport** line controls all **trapsink** lines that follow it until another **trapport** line appears. Therefore, to change the trap port for a trap sink, the new **trapport** line must be inserted before the trap sink's **trapsink** line, with no other **trapport** lines in between. The same logic follows for **trapcommunity** lines.*

/etc/snmptrap.conf

This configuration file includes OID, trap, and regular expression mappings. The configuration file specifies whether to send a specific trap based on a regular expression. An excerpt of the configuration file is shown in Figure 9.2.

```
# Default traps.
.1.3.6.1.4.1.3375.1.1.110.2.6 (ROOT LOGIN) ROOT LOGIN
.1.3.6.1.4.1.3375.1.1.110.2.5 (denial) REQUEST DENIAL
.1.3.6.1.4.1.3375.1.1.110.2.4 (BIG-IP Loading) SYSTEM RESET
.1.3.6.1.4.1.3375.1.1.110.2.3 (Service detected UP) SERVICE UP
.1.3.6.1.4.1.3375.1.1.110.2.2 (Service detected DOWN) SERVICE DOWN
#.1.3.6.1.4.1.3375.1.1.110.2.1 (error) Unknown Error
#.1.3.6.1.4.1.3375.1.1.110.2.1 (failure) Unknown Failure
```

Figure 9.2 Excerpt from the */etc/snmptrap.conf* file

Some of the OIDs have been permanently mapped to Link Controller specific events. The OIDs that are permanently mapped for the Link Controller include:

- Root login
- Request denial
- System reset
- Service up
- Service down

You may, however, insert your own regular expressions and map them to the **110.1 OID**. The */etc/snmptrap.conf* file contains two examples for mapping your own OIDs:

- Unknown error
- Unknown failure

By default, the lines for these files are commented out. Use these OIDs for miscellaneous events. When lines match your expression, they are sent to your management software with the 110.2.1 OID.

If you change this file, restart the SNMP agent **bigsnmppd** as follows:

```
bigstart restart bigsnmpd
```

For the 3-DNS Controller, the configuration in */etc/3dns_snmptrap.conf* determines which messages generate traps and what those traps are. Edit this file only if you want to add traps.

Syslog

In order to generate traps, you must configure **syslog** to send syslog lines to **checktrap.pl**. If the syslog lines make a match to the specified configuration in the *snmptrap.conf* file, a valid SNMP trap is generated. The following

lines in the `/etc/syslog.conf` file require that the **syslog** examine information logged, scan the **snmptrap.conf** file, and determine if a trap should be generated:

```
local0.* | exec /sbin/checktrap.pl.  
local1.* | exec /sbin/checktrap.pl.  
auth.* | exec /sbin/checktrap.pl.  
local2.* | exec /sbin/checktrap.pl. (for 3-DNS only)
```

◆ **Note**

*If you uncomment these lines, make sure you restart **syslogd**.*

If you change this file, restart the SNMP agent **bigsnmppd** with the following command:

```
bigsstart restart bigsnmppd
```

Configuring snmpd to send responses out of different ports or addresses

You can configure the **snmpd** daemon to respond on different ports or bind the daemon to a specific interface. Use the following syntax to configure **snmpd**:

```
snmpd -p [(udp|tcp):]port[@address][, ...]
```

Use this command to make the agent list on the specified list of sockets instead of the default port, which is port 161. Separate multiple ports by commas. You can specify transports by prepending the port number with the transport name (**udp** or **tcp**) followed by a colon.

To bind to a particular interface, you can specify the address you want it to bind with. For example, you can specify the following command to make the agent listen on UDP port 161 for any address, TCP port 161 for any address, and UDP port 9161 on only the interface associated with the localhost address.

```
snmpd -p 161, tcp:161, 9161@localhost
```

◆ **Note**

*The **-T** flag changes the default transport mapping to use (in the previous example, the default transport mapping is UDP).*



A

---

bigpipe Command Reference

---

bigpipe commands

This chapter lists the various **bigpipe** commands, including syntax requirements and functional descriptions. Table A.1 outlines the conventions used in the command line syntax.

Item in text	Description
\	Continue to the next line without typing a line break.
< >	You enter text for the enclosed item. For example, if the command has <your name> , type in your name.
	Separates alternate options for a command.
[ ]	Syntax inside the brackets is optional.
...	Indicates that you can type a series of items.

Table A.1 *Command line conventions*

The following table provides a concise listing of the individual **bigpipe** commands, along with the page reference where you can find the detailed description.

Command	Description	Page
-?	Displays online help for an individual bigpipe command.	A-3
config	Synchronizes the /config/bigip.conf between the two Link Controller units in a redundant system.	A-4
conn	Shows information about current connections such as the source IP address, virtual server and port, and node.	A-6
default_gateway	Creates a pool of default gateways.	A-7
failover	Sets the Link Controller as active or standby.	A-8
global	Sets global variable definitions.	A-9
-h and help	Displays online help for bigpipe command syntax.	A-16
interface	Sets options on individual interfaces.	A-17
load	Loads the Link Controller configuration and resets.	A-20
maint	Toggles the Link Controller into and out of maintenance mode.	A-21
merge	Loads a saved Link Controller configuration without resetting the current configuration.	A-22

Command	Description	Page
mirror	Copies traffic from any port or set of ports to a single, separate port.	A-23
monitor	Defines a health check monitor.	A-25
-n	Displays addresses and ports numerically rather than by name.	A-32
nat	Defines external network address translations for nodes.	A-33
node	Defines node property settings.	A-35
pool	Defines load balancing pools.	A-37
ratio	Sets load-balancing weights and priority levels used in the Ratio and Priority load balancing modes.	A-41
reset	Clears the Link Controller configuration and counter values.	A-42
save	Writes the current configuration to a file.	A-43
self	Assigns a self IP address for a VLAN or interface.	A-44
service	Defines properties for services.	A-46
snat	Defines and sets options for SNAT (Secure NAT).	A-47
stp	Implements spanning tree protocol (STP).	A-49
summary	Displays summary statistics for the Link Controller.	A-50
trunk	Aggregates links to form a trunk.	A-51
unit	Displays the unit number assigned to a particular Link Controller.	A-52
verbose	Used to modify the verbose log level.	A-53
verify	Parses the command line and checks syntax without executing the specified command.	A-54
version	Displays the bigpipe utility version number.	A-55
virtual	Defines virtual servers, virtual server mappings, and virtual server properties.	A-56
vlan	Defines VLANs, VLAN mappings, and VLAN properties.	A-59
vlangroup	Defines VLAN groups.	A-61

-?

b <command> -?

For certain commands, displays online help, including complete syntax, description, and other related information. For example, to see online help for the **bigpipe service** command, type:

b service -?

config

```
b config save <file>
b config install <file>
b config sync
b config sync all
b config sync running
```

Manages user configuration sets. A user configuration set (UCS) is the set of all configuration files that a user may edit to configure a Link Controller. A UCS file is an archive that contains all the configuration files in a UCS.

The **config** command allows you to save the system configuration to a UCS file, install the configuration from a UCS file, and synchronize the configuration with the other systems in a redundant pair.

Options

The **config save <file>** option saves the currently running configuration to **/config/bigip.conf** and **/config/bigip_base.conf**, and creates the UCS file with the file name specified by **<file>**.

The **config install <file>** option unpacks and installs the UCS file specified by **<file>**, overwriting all configuration files, including the file **/config/bigip.conf**.

The **config sync** option saves the currently running configuration to the file **/config/bigip.conf** and copies the file **/config/bigip.conf** to the other Link Controller in a redundant pair.

The **config sync all** option creates a temporary UCS file and transfers it to the other Link Controller.

The **config sync running** option saves the currently-running configuration to a temporary file and copies it to the other Link Controller.

Saving configuration files to an archive

The **config save <file>** command saves all configuration files to a single archive file, **<file>.ucs**, on the local unit without copying it to the standby unit. By default, **<file>.ucs** is saved to the directory **/user/local/ucs**. An alternate location can be specified by expressing **<file>** as a relative or absolute path. For example:

```
b config save /user/local/config_backup/my_conf
```

This writes the file **my_conf.ucs** to the directory **/user/local/config_backup**.

Installing an archived configuration file

config install <file> reinstalls the archived configuration files saved as **<file>.ucs** to their working locations on the local unit.

If you use command line utilities to set configuration options, be sure to save the current configuration to the relevant files before you use the configuration synchronization feature. (Alternatively, if you want to test the memory version on the standby unit first, use **bigpipe config sync running**.) Use the following **bigpipe** command to save the current configuration:

b save

◆ Note

*A file named **/usr/local/ucs/cs_backup.ucs** is created prior to installing a UCS from a remote machine.*

Synchronizing configuration files

config sync without the **all** option synchronizes only the basic configuration file **/config/bigip.conf**.

config sync all synchronizes the following configuration files:

- The common **BIG/db** keys
- All common files in **/config**
- All common files in **/etc**

config sync running synchronizes the running version of **/config/bigip.conf**, which is the image that resides in memory as the system runs. This file is loaded into memory on the standby unit, it is not saved.

◆ Note

*The **config sync** command applies only to Link Controller and not to 3-DNS.*

config save <file> saves all configuration files to a single archive file, **<file>.ucs**, on the local unit without copying it to the standby unit. By default, **<file>.ucs** is saved to the directory **/user/local/ucs**. An alternate location can be specified by expressing **<file>** as a relative or absolute path. For example:

b config save /user/local/config_backup/my_conf

This writes the file **my_conf.ucs** to the directory **/user/local/config_backup**.

conn

```
b conn [ <client_ip>[:<client_service>] ] dump [mirror]
```

Displays information about current client connections to virtual addresses and virtual servers. This command can also show connections that are active on the given Link Controller, as well as those that are standby connections for the peer Link Controller. By default, the **dump** command only shows items that are active on the given unit.

Displaying all current connections

The following command displays all current client connections:

```
b conn dump
```

The output shows the source IP address, virtual server IP address, and node to which the client is connected.

```
bigip conn dump

fromvirtualnode
100.100.100.30:49152 ->100.100.100.100:23 ->200.200.200.10:23
100.100.101.90:49153 ->100.100.100.100:80 ->200.200.200.10:80
...
```

*Figure A.1 Formatted output of the **conn** command*

This command can also show connections that are active on the given Link Controller, as well as those that are standby connections for the peer Link Controller. By default, the **dump** command only shows items that are active on the given unit.

Displaying connections for a specific virtual server

Use the following syntax to display the current connections for a specific virtual server:

```
b conn <client_ip>[:<client_service>] dump
```

Note that the argument **<client_service>** refers to what is typically a five-digit number displayed in the output of this command.

Displaying standby connections

To view standby items, you must use the **mirror** qualifier, as follows.

```
b conn dump mirror
```

default_gateway

```
b default_gateway use pool <pool_name>
b default_gateway show
b default_gateway delete
```

This command creates, shows, or deletes a pool of default gateways, with nodes in the pool corresponding to different routes. Connections originating from the system with a destination for which there is no other route choose a route from the default gateway pool. Note that the default gateway pool is not a last-hop pool for services running on the system.

There can be only one default gateway pool at any one time.

Defining a default gateway pool removes the need to define a default route. However, if a default route is defined, that route will be used when all the nodes in the default gateway pool are down.

Since the system performs route lookups on nodes as they are defined, the default gateway pool must be stored at the top of the **bigip.conf** file. Also, all nodes in the default gateway pool must reside on the same IP network as the system.

We recommend that all nodes in the default gateway pool have the same MTU.

As an alternative to using the **default_gateway** command, you can use the Setup utility, which allows you to create the default gateway pool at the time that you configure your base network.

Options

The **use pool <pool_name>** option specifies the name of the default gateway pool and must be 1-31 characters in length. Example: **my_pool**.

The **show** option shows the members of the default gateway pool.

The **delete** option deletes the default gateway pool.

failover

```
b failover standby | show | init | failback
```

Switches the Link Controller to be the standby unit in a redundant configuration. This command should be used with care, and is provided only for special situations. The Controller automatically switches between active and standby modes, without operator intervention.

Options

The **standby** option switches the Link Controller to the standby unit.

The **show** option displays the node on which the Link Controller is currently running.

The **init** option initializes the Link Controller's initial state.

Changing failover state

Before you switch the current mode, first determine which mode the Controller is running using the command above. In an active/standby or active-active configuration, run the following command to switch the Controller to be the standby unit:

```
b failover standby
```

Displaying failover status

Show the status of the Link Controller with the following command:

```
b failover show
```

Initializing failover state

You can use the **bigpipe failover init** command to refresh the parameters of the failover mechanism with any new configuration data entered into the BIG/db database.

```
b failover init
```

global

```

b global auto_lasthop enable | disable | show
b global fastest_max_idle_time <seconds>
b global fastflow_active auto | on | off | show
b global fastflow_active auto | on | off | show
b global gateway failsafe arm | disarm | show
b global ipforwarding enable | disable
b global mirror enable | disable | show
b global memory_reboot_percent <percent>
b global open_3dns_ports enable | disable | show
b global open_corba_ports enable | disable | show
b global open_snmp_ports enable | disable | show
b global open_telnet_port enable | disable
b global open_ftp_ports enable | disable
b global open_ssh_port enable | disable
b global open_rsh_ports enable | disable
b global open_failover_ports enable | disable | show
b global persist_map_proxies enable | disable
b global persist timer limit | timeout | show
b global persist across_services enable | disable
b global persist across_virtuals enable | disable
b global self_conn_timeout enable | disable | show
b global sticky table_limit <max_num> | show
b global verbose_log_level <level>
b global webadmin_port <port>
b global l2_aging_time <seconds>

```

auto_lasthop

When this variable is enabled, it automatically designates the lasthop router inside IP address as a lasthop route for replies to inbound traffic. If **auto_lasthop** is disabled, the lasthop router inside IP address must be specified as a **lasthop pool**. The default setting is **enable**.

fastest_max_idle_time

Sets the number of seconds a node can be left idle by the **fastest** load balancing mode. This forces the Link Controller to send fewer connections to a node that is responding slowly, and also allows the Link Controller to periodically recalculate the response time of the slow node.

fastflow_active

You can use this variable to control additional enhancements that speed packet flow for TCP connections when the packets are not fragmented. In most configurations these software enhancements are automatically turned on and do not require any additional configuration.

However, you may want to turn off these enhancements for individual virtual servers that use IPFW rate filters. With the speed enhancements **on**, IPFW only examines the first SYN packet in any given connection. If you want to filter all packets, you should turn the speed enhancements **off**. To do this, you first set the global state of the system **on**, and then you turn the feature **off** for individual virtual servers that use IPFW rate filtering. You can also change the settings for these enhancements from the command line or in the Configuration utility.

There are three global states you can set with **fastflow_active**. The default state is **auto**. The global states are:

- **off**
- **auto**
- **on**

The additional speed enhancements are globally disabled if the **sysctl** variable **fastflow_active** is **off** or if **fastflow_active** is set to **auto** and an IPFW rate filter exists in the configuration.

To provide the benefits of software acceleration for virtual servers that do not use rate filtering and turn off software acceleration for virtual servers that use IPFW rate filtering, you can set the global variable **fastflow_active** to **on** with the following command:

```
b global fastflow_active on
```

After you set the **sysctl** variable, use the following **bigpipe** command to disable software acceleration for virtual servers that use IPFW rate filtering:

```
b virtual <ip>:<port> accelerate disable
```

gateway failsafe

Turns the gateway fail-safe feature on and off. This command is supported only for redundant systems.

The typical use of gateway fail-safe is a setup where active and standby Link Controller units use different routers as gateways to the Internet. Fail-over is triggered if the gateway for the active unit is unreachable.

To arm fail-safe on the gateway, enter the following command:

```
b global gateway failsafe arm
```

To disarm fail-safe on the gateway, enter the following command:

```
b global gateway failsafe disarm
```

To see the current fail-safe status for the gateway, enter the following command:

```
b global gateway failsafe show
```

For more information about configuring gateway fail-safe, see *Health monitors*, on page 2-79.

ip forwarding

Enables IP forwarding for the Link Controller. IP forwarding exposes all of the node IP addresses to the external network, making them routable on that network. The default setting is **disabled**.

mirror

Enables mirroring functions globally for the Link Controller. The mirror feature duplicates the active unit's real-time connection or persistence information state on the standby unit for smooth transition to the inactive unit at fail-over. The default setting is **enabled**.

memory_reboot_percent

The value you type, **80** or higher, is the percentage of memory that is in use before the Link Controller automatically reboots. The default value for this variable is **95**. To disable this feature, set the value to **0**.

open_3dns_ports

This variable is required only when running one or more separate 3-DNS Controllers in the network. It does not apply to running the 3-DNS software module on the Link Controller itself. The variable is disabled on the Link Controller when the 3-DNS Controller is not present in the network configuration.

open_corba_ports

This variable enables and disables the CORBA ports, which allow administrative CORBA connections. The default setting is **disabled**.

open_snmp_ports

This variable enables and disables the SNMP ports, which allow administrative SNMP connections. The default setting is **disabled**.

open_telnet_port

This variable enables or disables ports for Telnet access, and the default setting is **disable**.

The following command sets this variable to open the Telnet port (**23**) to allow administrative Telnet connections. This is useful for Link Controller units that do not support encrypted communications, or for a unit that needs to communicate with the 3-DNS software.

The following command opens the Telnet port:

```
b global open_telnet_port enable
```

The following command closes the Telnet port:

b global open_telnet_port disable

open_ftp_ports

This variable enables or disables ports for FTP access, and the default setting is **disable**.

The following command open the FTP ports (**20** and **21**) to allow administrative FTP connections, which is useful for Link Controller units that do not support encrypted communications.

b global open_ftp_ports enable

The following command closes FTP ports:

b global open_ftp_ports disable

open_ssh_ports

This variable enables or disables ports for SSH access on Link Controller units that support encrypted communication. The default setting is **enable**.

The following command opens the SSH port (**22**) to allow encrypted administrative connections:

b global open_ssh_port enable

The following command closes the SSH port:

b global open_ssh_port disable

open_rsh_ports

This variable enables or disables ports for RSH access, and it is useful for Link Controller units that do not support encrypted communications, or for connecting to 3-DNS Controllers that do not support encrypted communication.

The default setting is **disable**.

The following command opens the RSH ports (**512**, **513**, and **514**) to allow RSH connections:

b global open_rsh_ports enable

The following command closes RSH ports:

b global open_rsh_ports disable

open_failover_ports

This variable enables or disables network failover when a VLAN has port lockdown enabled.

The following command enables network failover:

b global open_failover_ports enable

The following command disables network failover:

b global open_failover_ports disable

persist map_proxies

The default setting for the map proxies for the persistence variable is **enable**. The AOL proxy addresses are hard-coded. This enables you to use client IP address persistence with a simple persist mask, but forces all AOL clients to persist to the same server. All AOL clients will persist to the node that was picked for the first AOL client connection received.

The class B networks, **195.93** and **205.188**, are mapped to **152.163** for persistence. For example, client **195.93.3.4** would map to **152.63.3.4** for persistence records only. This mapping is done prior to applying the persist mask. Use **bigpipe pool persist dump** to verify that the mapping is working.

We recommend that in addition to setting this **sysctl** variable, you set a persist mask of **255.255.0.0** so that all the AOL addresses map to a common address. For example, Table A.2 is an example of how setting this variable and a persist mask of **255.255.0.0** would map a sample set of client addresses.

Sample Client Address	Persist Address
152.44.12.3	195.93.0.0
152.2.99.7	195.93.0.0
170.11.19.22	195.93.0.0
202.67.34.11	195.93.0.0
205.188.11.2	195.93.0.0
208.33.23.4	208.33.0.0 (non AOL address is not mapped)

Table A.2 Address mapping of sample clients

persist timer

The following command forces the persistent connection timer to reset on each packet for persistent sessions. This is the default value.

```
b global persist timer limit
```

The following command resets the timer only when the persistent connection is **initiated**.

```
b global persist timer timeout
```

◆ Note

For SSL persistence, the timer is always reset on each packet.

`persist across_services`

When this variable is enabled, all simple persistence connections from a client IP address that go to the same virtual address also go to the same node (matches the client address and the virtual IP address but not the virtual port).

The default setting for this variable is **disabled**.

`persist across_virtuals`

When this variable is enabled, all simple persistent connections from the same client IP address are sent to the same node (matches the client IP address but not the virtual address or virtual port the client is using). The default setting for this variable is **disabled**.

`self_conn_timeout`

This variable is used as a tracking mechanism for UDP connections. After the number of seconds specified by this variable has expired, the UDP connection terminates. The default value for this variable is **5**.

`sticky table_limit`

This is the maximum number of sticky entries allowed to accumulate on the Link Controller when using destination address affinity (sticky persistence). When the maximum value is reached, the Link Controller stops accumulating sticky entries. The default value for this entry is **2048**.

`verbose_log_level`

This variable sets logging levels for both TCP and UDP traffic. Each log level is identified by a level number used in place of the **<level>** parameter.

The following command turns on port denial logging for both TCP and UDP traffic. This logs TCP and UDP port denials to the virtual server address and the Link Controller address.

```
b global verbose_log_level 15
```

The following command turns logging off altogether:

```
b global verbose_log_level 0
```

Setting log levels only for TCP traffic

The following command turns on only TCP port denial logging, which logs TCP port denials to the Link Controller address.

```
b global verbose_log_level 2
```

The following command turns on virtual TCP port denial logging, which logs TCP port denials to the virtual server address.

```
b global verbose_log_level 8
```

Setting log levels for UDP traffic

The following command turns on only UDP port denial logging, which logs UDP port denials to the Link Controller address.

```
b global verbose_log_level 1
```

The following command turns on only virtual UDP port denial logging, which logs UDP port denials to the virtual server address.

```
b global verbose_log_level 4
```

webadmin_port

Specifies the port number used for administrative web access. The default port for web administration is port **443**.

l2_aging_time

Specifies a time period after which dynamic entries in the L2 forwarding table are flushed out if the MAC address is no longer present on the network. The default value is **300** seconds.

-h and -help

```
b [-h | -help ]
```

Displays the **bigpipe** command syntax or usage text for all current commands.

◆ **Note**

*More detailed man pages are available for some individual **bigpipe** commands. To display detailed online help for the **bigpipe** command, type: **man bigpipe**.*

---

interface

```
b interface show
b interface [<interface_name>] show [verbose]
b interface <interface_name> media show
b interface <interface_name> duplex show
b interface <interface_name> media <media_type>
b interface <interface_name> duplex full | half | auto
b interface [<interface_name>] stats reset
b interface <interface_name> enable | disable
b interface <interface_name> renames <driver_name>
```

Displays the names of installed network interface cards and, for each interface, sets properties such as MAC address, media options, duplex mode, and status, resets interface statistics, enable or disable interfaces, and change driver name mappings.

Options

The **<interface_name>** variable is a name such as **3.1**, where **3** is the physical slot number holding the network interface hardware and **1** is the physical port number on that interface on that hardware.

The **show [verbose]** option displays the current status, settings, and network statistics for the specified interface. The **verbose** argument provides more detailed information. If no interface is specified, this option displays information for all interfaces.

The **media show** option displays information about the media type for the specified interface.

The **duplex show** option displays the duplex mode of the specified interface.

The **media <media_type>** option is a valid media type for the specified interface. Examples include **auto**, **100baseTX**, and **10baseT**. Note that only certain combinations of media type and duplex mode are valid for any particular type of interface.

The **duplex full | half | auto** option sets the duplex mode of the specified interface.

The **stats reset** option resets the statistics for the specified interface.

The **enable | disable** option enables or disables the specified interface.

The **renames <driver_name>** option changes the mapping from the interface's driver name to its physical location name. The **<driver_name>** option is the network interface name in the form of driver and unit number, such as **exp0** and **bs1**. Note that this is the old-style network interface name.

Displaying interface information

To display the status, settings, and statistics for all interfaces on the Link Controller, use the following command.

```
b interface show [verbose]
```

To display the status, settings, and statistics for a specific interface on the Link Controller, use the following command-line syntax.

```
b interface <interface_name> show [verbose]
```

Note that if the **verbose** argument is used, the output provides additional information on status. If the **verbose** argument is not used, the output focuses on statistics.

To display the media type for an interface, use the following command-line syntax,

```
b interface <interface_name> media show
```

To display the duplex mode for an interface, use the following command-line syntax.

```
b interface <interface_name> duplex show
```

Setting the media type

The media type may be set to the specific media type for the interface card or it may be set to **auto** for auto detection. If the media type is set to **auto** and the card does not support auto detection, the default type for that interface will be used, for example **1000BaseTX**.

To set the media type, use the following command-line syntax.

```
b interface <interface_name> media <media_type>
```

Setting the duplex mode

Duplex mode may be set to **full**, **half duplex**, or **auto**. If the media type does not allow duplex mode to be set, this will be indicated by an onscreen message. If media type is set to **auto**, or if setting duplex mode is not supported, the duplex setting will not be saved to the **bigip.conf** file.

To set the duplex mode, use the following command-line syntax.

```
b interface <interface_name> duplex full | half | auto
```

Resetting statistics

You can reset interface statistics for all interfaces or for a specific interface.

To reset statistics for all interfaces, use the following command.

```
b interface stats reset
```


To reset statistics for a specific interface, use the following command-line syntax:

```
b interface <interface_name> stats reset
```

Enabling or disabling an interface

Enabling or disabling an interface allows you to control whether the interface receives and sends packets. If an interface begins to behave strangely, you disable and then enable the interface to effectively reset it.

To enable or disable an interface, use the following command-line syntax.

```
b interface <interface_name> enable | disable
```

Changing driver name mapping

You can change the mapping from an interface's driver name to its physical location name, using the following syntax.

```
b interface <interface name> renames <driver name>
```

load

```
b [verify] load [ <filename> | - ]  
b [-log] load [ <filename> | - ]
```

Resets all of the Link Controller settings and then loads the configuration settings, by default from the **/config/bigip.conf** and **/config/bigip_base.conf** files.

For testing purposes, you can save a test configuration by renaming it to avoid confusion with the boot configuration file. To load a test configuration, use the **load** command with the **<filename>** parameter. For example, if you renamed your configuration file to **/config/bigtest.conf**, the command would be:

```
b load /config/bigtest.conf
```

The command checks the syntax and logic, reporting any errors that would be encountered if the command executed.

You can type **b load -** in place of a file name, to display the configuration on the standard output device.

```
b save -
```

Use the **load** command together with the **verify** command to validate the specified configuration file. For example, to check the syntax of the configuration file **/config/altbigpipe.conf**, use the following command:

```
b verify load /config/altbigpipe.conf
```

The **-log** option will cause any error messages to be written to **/var/log/bigip** in addition to the terminal.

maint

```
b maint
```

Toggles a Link Controller into and out of maintenance mode. When in maintenance mode, a Link Controller accepts no new connections, but it does allow existing connections to complete.

The **maint** command interactively prompts you to enter or exit the maintenance mode.

b maint

If the Link Controller is already in maintenance mode, the **maint** command takes the Link Controller out of maintenance mode. If the Link Controller is in maintenance mode for more than 20 minutes, that Link Controller immediately begins to accept new connection requests.

If the Link Controller has been in maintenance mode for more than 20 minutes, it automatically updates all network ARP caches; this process normally takes a few seconds. However, you can speed up the process by reloading the configuration file, using the following command:

b -f /config/bigip.conf

merge

```
b [-log] merge [<file_name>]
```

Loads the Link Controller configuration from the file specified in the **<file_name>** variable, without resetting the current configuration.

---

mirror

```
b mirror [<mirror_to_interface>] show
b mirror <mirror_to_interface> interfaces add <interface_list>
b mirror <mirror_to_interface> interfaces delete <interface_list>
b mirror <mirror_to_interface> delete
```

For the Link Controller, you can copy traffic from any port or set of ports to a single, separate port. This is called *port mirroring*. You should attach a sniffer device to the target port, called the *mirror-to* port, for debugging and/or monitoring.

Options

The **<mirror_to_interface>** variable specifies the port to which you want one or more ports to be mirrored.

The **show** option displays a specific mirror-to interface. If no interface is specified, this option displays all mirror-to interfaces.

The **interfaces add <interface_list>** variable specifies one or more ports that you want to mirror to the mirror-to port.

The **interfaces delete <interface_list>** variable specifies one or more ports that you want to delete from a port mirror.

The **delete** option deletes the specified mirror-to interface.

Displaying port mirroring

Using the argument, you can display all mirror-to interfaces or a specific mirror-to interface.

To display all mirror-to interfaces, type the following command:

```
b mirror show
```

To display a specific mirror-to interface, use the following command-line syntax:

```
b mirror <mirror_to_interface> show
```

Creating a port mirror

Creating a port mirror consists of specifying a mirror-to port and adding to it one or more ports (that is, a port list) to be mirrored. The **bigpipe** syntax for setting up port mirroring is:

```
b mirror <mirror_to_interface> interfaces add <interface_list>
```

For example, you could type the following command:

```
b mirror 3.24 interfaces add 3.1 3.3 3.10
```

Deleting interfaces from a port mirror

The **bigpipe** syntax for deleting interfaces from a port mirror is as follows:

```
b mirror <mirror_to_interface> interfaces delete <inteface_list>
```

For example, you could type the following command:

```
b mirror 3.24 interfaces delete 3.10
```

Deleting a port mirror

The **bigpipe** syntax for deleting a port mirror is:

```
b mirror <mirror_to_interface> delete
```

For example, you could type the following command:

```
b mirror 3.24 delete
```

monitor

```
b monitor <monitor_name> '{ use <monitor_template> [<attr> <attr_value>]... }'
b monitor show all
b monitor <monitor_name> show
b monitor dump [all]
b monitor <name> delete
b monitor <name> enable | disable
b monitor instance <ip_address>:<service> enable | disable
b monitor instance <ip_address> enable | disable
```

Defines a health monitor. A health monitor is a configuration object that defines how and at what intervals a node is pinged to determine if it is **up** or **down**. Once a monitor is defined, instances of the monitor are created for a node or nodes, one instance per node, using the **bigpipe node** command.

Monitors verify services and connections of node servers. The **icmp** or **tcp_echo** monitors may be used to monitor node addresses. If the node server or node address fails to respond in the specified timeout period, it will be marked as **down**. When a node server or node address is marked as **down**, traffic is no longer directed to it.

Several steps are needed to create a monitor and associate it with a node server or node address. A monitor must be created, based on a monitor template that the Link Controller provides. In some cases, a monitor template is directly usable. Once a monitor is created, the node address or node server is associated with the monitor, creating a monitor instance.

Options

The **<monitor_name>** variable specifies the name you want to use for the monitor you are creating or managing.

The **<monitor_template>** variable specifies the health monitor template you want to use to create your monitor. For a list of templates that you can specify, see *Monitor templates*, on page A-29.

The **<attr>** variable specifies an attribute of the monitor to which you want to assign a value. For a list of monitor attributes, see *Monitor templates*, on page A-29.

The **<attr_value>** variable specifies the value of the attribute specified with the **<attr>** option.

The **show all** option displays all existing monitors.

The **show** option displays the specified monitor.

The **delete** option deletes the specified monitor.

The **enable | disable** option enables or disables the specified monitor.

The **instance** **<ip address>:<service>** option enables or disables a monitor instance for the specified IP address and port.

The **instance** **<ip address>** option enables or disables a monitor instance for the specified IP address.

Creating a monitor

Creating a monitor simply names and sets the options for a monitor, based on a monitor template. The options may be obtained from a predefined set of default options or the option values may be specified on the command line during creation.

Options include destination address, interval time, timeout value, send string, and receive string, etc. Options can be changed later using the **modify** option.

The following is an example of a command to create an **http** monitor:

```
b monitor my_http '{ use http send "GET /my.html"\nrecv "TESTING" }'
```

The command above creates a monitor with the name **my_http**, based on the **http** template. The **send** and **recv** strings are modified from the default values. The **interval**, **timeout**, **destination address**, **username**, and **passwd** configuration options are not specified on the command line because the monitor will use the default values.

Note that single quotes are used when entering monitor commands on the command line, to prevent the command shell from attempting to interpret the double quotes within the monitor definition.

Modifying a monitor

If you want to change the default values of certain options, such as **interval** and **timeout**, you can use syntax as in the following example:

```
b monitor my_http '{ interval <seconds> timeout <seconds> }'
```

Creating a monitor instance

Creating a monitor instance simply associates a monitor or group of monitors with a node address or node server.

Each monitor template contains a destination address option. Almost always, this is the meta character string **"*:*"**, which causes the Link Controller to create the monitor instance using the IP address and port supplied on the command line. For example, the destination address option **dest** in the **tcp** monitor template is shown in Figure A.2.


```
monitor tcp {
    # type tcp
    interval 5
    timeout 16
    dest *.*
    send ""
    recv ""
}
```

Figure A.2 Destination address option in a monitor template

We can create two instances of this monitor by entering the following command:

```
b node 10.10.10.10:80 10.10.10.12:80 monitor use tcp
```

The **dest *** attribute in the **tcp** monitor causes the two monitor instances to be created, substituting the IP address and port combination supplied on the command line into the destination address. In other words, there are two monitor instances created, one that communicates with address **10.10.10.10:80**, and one that communicates with **10.10.10.12:80**. The node **10.10.10.10:80** depends on the monitor instance **10.10.10.10:80**. If the monitor instance cannot get a response from node **10.10.10.10:80**, then the node is marked as **down**. The same is true for node **10.10.10.12:80**.

It is also possible to enter explicit addresses into a monitor. For example, Figure A.3 shows a monitor called **exp_tcp** that specifies an explicit destination address.

```
monitor exp_tcp {
    # type tcp
    use "tcp"
    interval 5
    timeout 16
    dest 10.10.10.24:80
    send ""
    recv ""
}
```

Figure A.3 Explicit destination address in a monitor

In this case, the following command causes one monitor instance to be created, one that communicates with address **10.10.10.24:80**:

```
b node 10.10.10.10:80 10.10.10.12:80 monitor use exp_tcp
```

In this case, the nodes **10.10.10.10:80** and **10.10.10.12:80** depend on the health of node **10.10.10.24:80**. If that node does not respond, both **10.10.10.10:80** and **10.10.10.12:80** are marked as **down**.

The following is another example of specifying a destination address on the command line:

```
b node '*:http' monitor use my_http
```

The command above creates a monitor instance for all node addresses with a service of **http**. Note that it is necessary to enter the single quotes when entering this command on the command line to prevent the shell from interpreting the special character *****.

Modifying a monitor instance

The **enable/disable** attribute can be changed within a monitor instance. For example:

```
b monitor instance 10.20.3.2:http disable
```

This command disables a monitor instance for a node server. The monitor will not attempt to establish a connection with the service until it is later enabled.

Deleting a monitor

To delete a monitor, use the **bigpipe monitor** command with the **delete** option, as in the following example:

```
b monitor my_http delete
```

Deleting a monitor instance

To delete a monitor instance, use the **bigpipe node** command with the **delete** option, as in the following example:

```
b node '*:http' monitor delete
```

Displaying monitor templates

To display a specific monitor template, use the following command-line syntax:

```
b monitor <monitor template> show
```

When you issue the above command, the Link Controller displays the specified template.

To display all monitor templates, use the following command:

```
b monitor show all
```

Displaying monitor instances

Using the **bigpipe node** command, you can display the status of a monitor instance, along with the corresponding node status. For example:

```
b node 192.168.200.50:http monitor show
```

To see this information for all monitor instances, use the following command:

```
b node monitor show
```

Monitor templates

Table A.3 lists the monitor templates and shows the template-specific attribute sets for each.

Name/Type	Template-Specific Attribute Set
icmp	none
tcp_echo	transparent (optional)
tcp	send "" recv "" transparent (optional) reverse (optional)
http	username "" password "" send "GET /index.html" recv "" get (optional) url (optional) transparent (optional) reverse (optional)
https	username "" password "" send "GET /index.html" recv "" get (optional) url (optional) transparent (optional) reverse (optional)
external	run "" args ""
ftp	username "anonymous" password "bigip1@internal" get "/README" url (optional)
nntp	username "" password "" newsgroup "local"
pop3	username "" password ""
smtp	domain "bigip1@internal"

Table A.3 The monitor templates

Name/Type	Template-Specific Attribute Set
snmp_dca	CPU coefficient "" CPU threshold "" memory coefficient "" memory threshold "" disk coefficient "" disk threshold "" userid "" userid coefficient "" userid threshold ""
snmp_dca_base	userid "" userid coefficient "" userid threshold ""
imap	username "" password "" folder "INBOX" message_num (optional)
radius	username "username" password "password" secret "12345678"
ldap	base "o=Org, c=US" filter "sn=Doe"
sql	username "" password "" database ""
https_443	dest *:443

Table A.3 The monitor templates

Table A.4 defines the attributes used in the templates.

Attribute	Definition
interval <seconds>	Ping frequency time interval in seconds.
timeout <seconds>	Ping timeout in seconds.
dest <node_addr>	Ping destination node. <node_address> Usually *:* for simple monitors, *:* for all others, causing the monitor instance to ping the address or address:port for which it is instantiated. Specifying address and/or port forces the destination to that address/port.
send <string>	Send string for ECV. Default send and recv values are empty (""), matching any string.

Table A.4 Monitor attributes

Attribute	Definition
recv <string>	Receive expression for ECV. Default send and recv values are empty (""), matching any string.
get <string>	For the http and https monitors get replaces the recv statement, automatically filling in " GET ". For the ftp monitor get can be used to specify a full path to a file. This will automatically fill in dest .
url	For the http , https , and ftp monitors, url replaces the recv statement, supplying a URL and automatically fill in dest with the URL address.
reverse	A mode that sets the node down if the received content matches the recv string.
transparent	A mode that forces ping through the node to the dest address for transparent nodes, such as firewalls.
run <program>	An external user-added EAV program.
args <program_args>	List of command line arguments for external program. args are quoted strings set apart by spaces.
username <username>	User name for services with password security. For ldap this is a distinguished name (an LDAP-format user name).
password <password>	Password for services with password security.
newsgroup <newsgroup>	Newsgroup, for type nntp EAV checking only
database <database>	Database name, for type sql EAV checking only.
domain <domain_name>	Domain name, for type smtp EAV checking only
secret	Shared secret for radius EAV checking only.
folder	Folder name for imap EAV checking only.
message_num	Optional message number for imap EAV checking only
base	Starting place in the LDAP hierarchy from which to begin the query, for ldap EAV checking only.
filter	LDAP- format key of what is to be searched for, for ldap EAV checking only.

Table A.4 Monitor attributes

-n

b -n

Used with other commands, such as **bigpipe virtual**, to display services and IP addresses numerically rather than by service name and host name, respectively. For example, type the following command to display services numerically:

b -n virtual

Figure A.4 shows an example of output that uses IP address instead of host names.

```
virtual +-----> 11.100.1.1          UNIT 1
|                (cur, max, limit, tot) = (0, 0, 0, 0)
|                (pkts,bits) in = (0, 0), out = (0, 0)
+----+--> SERVICE 80                UP
|                (cur, max, limit, tot) = (0, 0, 0, 0)
|                (pkts,bits) in = (0, 0), out = (0, 0)
MEMBER 11.12.1.100:80                UP
|                (cur, max, limit, tot) = (0, 0, 0, 0)
|                (pkts,bits) in = (0, 0), out = (0, 0)
```

Figure A.4 The output of **bigpipe -n virtual**

nat

```

b nat <orig_addr> to <trans_addr> [unit <unit ID>]
b nat <orig_addr> [...<orig_addr>] delete
b nat [<trans_addr> [...<trans_addr>] ] show | delete
b nat [<orig_addr> [...<orig_addr>] ] show | delete
b nat [<orig_addr>...] stats reset
b nat <orig_addr> vlans <vlan_list> enable | disable
b nat <orig_addr> vlans delete all
b nat <orig_addr> vlans show
b nat <orig_addr> arp [enable | disable | show]

```

Defines a network address translation (NAT), which is an IP address, routable on the external network, that a node can use to initiate connections to hosts on the external network and receive direct connections from clients on the external network. The **nat** command defines a mapping between the IP address of a server behind the Link Controller **<orig_addr>** and an unused routable address on the network in front of the Link Controller **<trans_addr>**.

The primary reason to define a NAT is to allow one of the servers in the server array behind the Link Controller to initiate communication with a computer in front of or external to the Link Controller.

Options

The **<orig addr>** variable is the originating IP address.

The **<trans addr>** variable is the translated IP address.

The **unit <unit ID>** option specifies a unit ID, currently **1** or **2**. The default unit ID is set to **1**.

The **delete** option deletes a NAT from the Link Controller.

The **stats reset** option resets statistics for the specified NAT.

The **vlans <vlan_list>** option lists the existing VLANs on which access to the NAT is enabled or disabled. A NAT is accessible on all VLANs by default.

The **vlans delete all** option deletes the specified NAT for all VLANs.

The **vlans show** option displays the VLANs on which the specified NAT is enabled.

Defining a NAT

Use the following syntax to define a NAT:

```
b nat <orig ip> to <trans ip> [unit <id>] [arp disable] \  
[vlangs <vlan name>... disable]
```

The node behind the Link Controller with the IP address specified by **<orig ip>** has a presence in front of the Link Controller as IP address **<trans ip>**.

For example:

```
b nat 11.0.0.100 to 10.0.140.100
```

Deleting a NAT

Use either of the following commands to permanently delete one or more NAT's from the Link Controller configuration:

```
b nat <orig_addr>... <orig_addr> delete  
b nat <trans_addr>... <trans_addr> delete
```

Additional Restrictions

The **nat** command has the following additional restrictions:

- A virtual server cannot use the IP address defined in the **<trans ip>** parameter.
- A NAT cannot use a Link Controller IP address.
- The IP address defined in the **<origin ip>** parameter must be routable to a specific server behind the Link Controller.
- A NAT cannot use an originating or translated IP address defined for and used by a SNAT or another NAT.
- You must delete a NAT before you can redefine it.

node

```

b node <node_ip>[:<service>]... enable | disable
b node <node_ip>[:<service>]... show
b node <node_ip>[:<service>]... limit <max_conn>
b node [<node_ip>:<service>]... stats reset
b node <node_ip>[:<service>] up | down
b node <node_ip>[:<service>] monitor use <monitor_name> [and <monitor_name>]...
b node [<node_ip>[:<service>]] monitor show | delete
b node <node_ip>[<node_ip>]... virtual | actual

```

Displays information about nodes and allows you to set properties for nodes, and node addresses. Nodes may be identified using wildcard notation. Thus * represents all nodes on the network, *.80 represents all port 80 nodes, 11.11.11.1:* represents all nodes with address 11.11.11.1.

Options

The **<node_ip>[:<service>]** variable is an IP address of the node address.

The **enable | disable** options enable or disable traffic for one or more specified IP addresses.

The **limit <max_conn>** option defines the maximum number of connections allowed for one or more specified nodes.

The **stats reset** option resets statistics for the specified node.

The **monitor use <monitor_name>** option associates one or more specified monitors with the specified node.

The **monitor show | delete** option shows or deletes a monitor instance running on the specified node.

Displaying nodes

You can display information about a specified node. For example, the following command displays information about node 192.168.200.50:20:

```
b node 192.168.200.50:20 show
```

Note that the **show** keyword is optional. The resulting information is displayed as follows:

```

NODE 192.168.200.50      UP          CHECKED
|   (cur, max, limit, tot) = (0, 0, 0, 0)
|   (pkts,bits) in = (0, 0), out = (0, 0)
+-  PORT 20              UP          CHECKED
      (cur, max, limit, tot) = (0, 0, 0, 0)
      (pkts,bits) in = (0, 0), out = (0, 0)

```

Modifying nodes

Use the following syntax to set the maximum number of connections allowed for one or more nodes:

```
b node <ip addr>:<port>... <ip addr>:<port> limit <limit>
```

Note that to remove a connection limit, you also issue the above command, but you set the **<limit>** variable to **zero**.

Use the following syntax to set the maximum number of connections allowed for one or more IP addresses:

```
b node <ip addr>... <ip addr> limit <limit>
```

Note that to remove a connection limit, you also issue the above command, but you set the **<limit>** variable to **zero**.

Use the following syntax to enable or disable traffic for one or more IP addresses:

```
b node <ip addr>... <ip addr> enable
```

```
b node <ip addr>... <ip addr> disable
```

◆ **Note**

*For information on using the **bigpipe node** command to associate a node with a health monitor, see **monitor**, on page A-25.*

pool

```

b pool <pool_name> { lb_method <lb_method_specification> <member_definition> }
b pool <pool_name> { lb_method <lb_method_specification> persist_mode
  <persist_mode_specification> <member_definition>... }
b pool <pool_name> { lb_method <lb_method_specification> min_active_members <min_value>
  <member_definition>... }
b pool <pool_name> { lb_method <lb_method_specification> <member_definition> fallback
  <host> <protocol> <port> <URI path> }
b pool <pool_name> { forward }
b pool <pool_name> add { <member_definition>... }
b pool <pool_name> delete { <member_definition>... }
b pool <pool_name> modify { [lb_method <lb_method_specification>] [persist_mode
  <persist_mode_specification>] <member_definition>... }
b pool <pool_name> { snat disable }
b pool <pool_name> header insert <quoted string>
b pool <pool_name> delete
b pool [<pool_name>] show
b pool <pool_name> lb_method show
b pool <pool_name> persist dump
b pool <pool_name> persist dump mirror
b pool <pool_name> { persist_mode simple | cookie | ssl | sip [sip_timeout <timeout>] |
  sticky | msrdp }
b pool sip dump
b pool <pool_name> sticky clear
b pool <pool_name> stats reset

```

Displays, creates, modifies, or deletes a pool definition. You can use pools to group members together with a common load-balancing mode and persistence mode.

Options

The **<pool name>** variable is a string from 1 to 31 characters, for example, **new_pools**.

The **<member_definition>** variable specifies the IP address of the member node being added to the pool.

The **<cookie name>** variable specifies a cookie name, which must be 1-31 characters in length.

The **lb_method <lb_method_specification>** option specifies the load balancing mode that Link Controller is to use for the specified pool.

The **persist_mode <persist_mode_specification>** option specifies the persistence type that Link Controller is to use for the specified pool.

The **min_active_members** <min_value> option specifies the minimum number of members that must remain available for traffic to be confined to a priority group when using priority-based activation.

The **fallback** option specifies HTTP redirection, using a set of format strings. You can use these strings to indicate unchanged host names, ports, and URI paths. For more information, see *Specifying HTTP redirection*, on page A-39.

The **forward** option specifies that the pool is to be a forwarding pool.

The **snat disable** option specifies that SNAT connections are to be disabled for that pool.

Displaying a pool

Using the **bigpipe pool** command, you can display specific pools or all pools, and display persistence within a pool.

Use the following syntax to display all pools:

```
bigpipe pool show
```

Use the following syntax to display a specific pool, such as **cgi_pool**:

```
bigpipe pool cgi_pool show
```

Use a command such as the following to display persistence within a pool:

```
bigpipe pool cgi_pool persist show
```

Creating a pool

To create a pool, use command-line syntax such as the following:

```
bigpipe pool cgi_pool { lb_method rr member 10.2.3.11:http \  
member 10.2.3.12:http }
```

This command creates a pool with two members **10.2.3.11** and **10.2.3.12**, and both members use the **round robin** load balancing method.

If the **lb_method** option is not set, it defaults to round robin.

To create a pool using simple persistence, use command-line syntax such as the following:

```
bigpipe pool cgi_pool { lb_method rr persist_mode simple \  
simple_timeout 100 simple_mask 255.255.255.0 \  
member 10.20.3.11:http member 10.20.3.12:http }
```

This command creates a pool with two members, **10.20.3.11** and **10.20.3.12**.

Both members use the **round robin** load balance method. Also, a simple persistence timeout of 100 seconds will be used with this pool. Note that an optional persistence mask may be specified with simple persistence.

Modifying a pool

You can modify a pool to change the defined attributes, such as adding or deleting members, changing the load balancing method, or changing the type of persistence being used.

The following example adds a new member to the existing pool **cgi_pool**:

```
bigpipe pool cgi_pool add { member 10.20.3.2:http }
```

The following example deletes a member from the existing pool **cgi_pool**:

```
bigpipe pool cgi_pool delete { member 10.20.3.2:http }
```

Deleting a pool

You can delete a pool altogether. For example, the following command deletes the pool **cgi_pool**:

```
bigpipe pool cgi_pool delete
```

Note that all references to a pool must be removed before a pool can be deleted.

Specifying HTTP redirection

To specify HTTP redirection (also known as fallback), you can use a set of format strings to indicate unchanged host names, ports, and URI paths. These format strings are as follows:

%h Host name, as obtained from the Host: header of the client

%p Port, from the virtual server listening port

%u URI path, as obtained from a GET/POST request

For example, the following command configures a pool to redirect an HTTP request from **http://www.example.com:8080/sample.html** to **https://www.example.com:443/sample.html**:

```
bigpipe pool my_pool fallback https://%h:443/%u
```

To indicate that the host name, port, and URI path remain unchanged, you would use the following command:

```
bigpipe pool my_pool fallback %h:%p/%u
```

Specifying a load balancing mode

The load balancing modes are specified as values of the attribute **lb_mode**. The **lb_mode** values are shown in Table A.5.

Mode Name	lb_mode attribute value
Round Robin	rr or omit lb_mode specification
Ratio	ratio
Ratio Member	ratio_member
Fastest	fastest
Fastest Member	fastest_member
Least Connections	least_conn
Least Connections Member	least_conn_member
Observed	observed
Observed Member	observed_member
Predictive	predictive
Predictive Member	predictive_member
Dynamic Ratio	dynamic_ratio

Table A.5 *Load balancing modes*

For more information about the load balancing modes, refer to *Load balancing method*, on page 2-5.

---

ratio

```
b ratio [<node_ip>] [<node_ip> ...] show
b ratio <node_ip> [<node_ip>...] <weight>
```

For the **ratio** load-balancing mode, this command sets the weight or proportions for one or more node addresses. For the **priority** load balancing mode, the command sets the priority level. Note that multiple node addresses can have the same priority level setting.

Options

The **<node_ip>** variable specifies an IP address of a specific node.

The **<weight>** variable specifies a whole number. The default weight for any node address is **1**.

The **show** option displays the ratio weights for the specified node addresses.

Displaying ratio settings

To display the current ratio settings for all node address that have ratio settings, use the following command:

```
b ratio [show]
```

The following output is displayed:

```
192.168.200.51    ratio = 3
192.168.200.52    ratio = 1
```

To display the ratio settings for specific node addresses, use the following command-line syntax:

```
b ratio <node addr> ... <node addr> [show]
```

Modifying ratio settings

The following command sets the ratio to **3** for the node address specified:

```
b ratio 192.168.103.20 3
```

reset

```
b reset
```

Clears the configuration values and counter values from memory.

 WARNING

Use this command with caution. All network traffic stops when you run this command.

Typically, this command is used on a standby Link Controller prior to loading a new **/config/bigip.conf** file that contains new service enable and timeout values.

For example, you can execute the following commands on a standby Link Controller:

```
b reset
b load <filename>
```

This sequence of commands ensures that only the values set in the **<filename>** specified are in use.

---

save

```
b save [ <filename> | - ]  
b base save [ <filename> | - ]
```

Writes the current Link Controller configuration settings from memory to the configuration files named **/config/bigip.conf** and **/config/bigip_base.conf**. (**/config/bigip.conf** stores high level configuration settings, such as pools, virtual servers, NATs, SNATs, and proxies. **/config/bigip_base.conf** stores low level configuration settings, like, VLANs, non-floating self IP addresses, and interface settings.)

You can type **b save <filename>**, or a hyphen character (-) in place of a file name, to display the configuration on the standard output device.

b [base] save -

If you are testing and integrating Link Controller units into a network, you may want to use multiple test configuration files. Use the following syntax to write the current configuration to a file name that you specify:

b [base] save <filename>

For example, the following command saves the current configuration from memory to an alternate configuration file named **/config/bigip.conf2**.

b save /config/bigip.conf2

self

```
b self <ip_addr> vlan <vlan_name | vlangroup_name> [ netmask <ip_mask> ][ broadcast  
  <broadcast_addr>] [unit <id>]  
b self <ip_addr> floating enable | disable  
b self <ip_addr> delete  
b self <ip_addr> show  
b self show  
b self <ip_addr> snat automap enable | disable
```

Defines a self IP address on a Link Controller. A self IP address is an IP address mapping to a VLAN or VLAN group and their associated interfaces on a Link Controller. A one true self IP address is assigned to each interface on the unit as part of first time boot configuration, and also a floating (shared) self IP address for units in a redundant pair. Additional self IP addresses may be created for health checking, gateway failsafe, routing, or other purposes. These additional self IP addresses are created using the **self** command.

Options

The **<ip_addr>** variable specifies an IP address to assign to the Link Controller.

The **vlan <vlan_name | vlangroup_name>** option specifies the VLAN or VLAN group to which the self IP address is being assigned.

The **netmask <ip mask>** option specifies an IP mask used to set the network of the self IP address.

The **broadcast <broadcast_addr>** option specifies the broadcast address.

The **unit <id>** option specifies an optional unit ID, **1** or **2**. The default value is **1**.

The **floating** option enables or disables a floating self IP address.

The **snat automap** option enables or disables SNAT automapping on the specified self IP address. Once **snat automap** is enabled, the self IP address can be used as the translation address when SNAT automapping is enabled for a VLAN.

Creating self IP addresses

The following are examples of using the **bigpipe self** command to create self IP addresses:

```
b self 10.1.0.1 vlan external netmask 255.255.0.0  
b self 10.2.0.1 vlan internal netmask 255.255.0.0
```

For a redundant configuration, the IP addresses that are shared by the two units are configured as floating IP addresses. For example:

```
b self 10.1.1.1 vlan external netmask 255.255.0.0 floating enable
b self 10.2.1.1 vlan internal netmask 255.255.0.0 floating enable
```

To create self IP addresses that are shared between the two units in an active-active configuration, assign a unit number to each self IP address, as in the following examples:

```
b self 10.1.1.1 vlan external netmask 255.255.0.0 unit 1 floating enable
b self 10.1.1.2 vlan external netmask 255.255.0.0 unit 2 floating enable
b self 10.2.1.1 vlan internal netmask 255.255.0.0 unit 1 floating enable
b self 10.2.1.2 vlan internal netmask 255.255.0.0 unit 2 floating enable
```

service

```
b service <service> [<service>...] limit <limit>
b service <service> [<service>...] tcp enable | disable
b service <service> [<service>...] timeout tcp <timeout>
b service <service> [<service>...] udp enable | disable
b service <service> [<service>...] timeout udp <timeout>
b service [<service>... ] show
b service [<service>... ] stats reset
```

Enables and disables network traffic on services, and also sets connection limits and timeouts. An idle connection is one in which no data has been received or sent for the number of seconds specified by the service timeout command.

The default timeout value for **tcp** services is **1005**, and **60** seconds for **udp** services. For idle connection reaping to be effective, you should set the timeout value to be greater than the configured timeout for the service daemons installed on your nodes.

You can use port numbers or service names (for example, **www**, **http**, or **80**) for the **<service>** parameter. Note that the settings you define with this command control the service for all virtual servers that use it. By default, all services are disabled.

Options

The **<service>** variable specifies any valid port number, between **1** and **65535**, inclusive, or any valid service name in the **/etc/services** file.

The **<limit>** variable specifies the maximum number of simultaneous connections to be allowed to the service for all virtual servers. To turn off a connection limit for a service, specify a value of **0**.

The **<seconds>** variable specifies the number of seconds until a connection to the service times out.

snat

```

b snat map <orig_ip> [...<orig_ip>] to <snat_ip><snat_ip> [unit <unit ID>] [netmask
  <ip>] [arp disable] [vlan <vlan_name_list> disable]
b snat map default to <snat_ip> [unit <unit ID>] [netmask <ip>]
b snat <snat_ip> [...<snat_ip>] delete | show
b snat default delete | show
b snat default dump [verbose]
b snat [<snat_ip> [...<snat_ip>] ] dump [verbose]
b snat globals show
b snat default show
b snat [<snat_ip> [...<snat_ip>] ] show
b snat [<snat_ip> [...<snat_ip>] ] delete
b snat [<snat_ip> [...<snat_ip>] ] arp show
b snat [<orig_ip> [...<orig_ip>] limit <max_conn>
b snat limit <max_conn>
b snat default limit <max conn>
b snat <orig_ip> [...<orig_ip>] mirror enable | disable
b snat default mirror enable | disable
b snat <orig_ip> [...<orig_ip>] timeout tcp | udp <seconds>
b snat default timeout tcp | udp <seconds>
b snat <orig_ip> [...<orig_ip>] stats reset
b snat default stats reset
b snat <orig_ip> [...<orig_ip>]> disable | enable
b snat <snat_ip> [...<snat_ip>] vlans <vlan_list> disable | enable
b snat <snat_ip> [...<snat_ip>] vlans enable all
b snat <snat_ip> [...<snat_ip>] vlans show
b snat map <vlan_name> to auto
b snat <snat_ip> [...<snat_ip>] arp [enable | disable]

```

The **snat** command creates and deletes secure network address translations (SNATs), and displays information about them. A SNAT defines one or more addresses that nodes can use as a source IP address when initiating connections to hosts on the external network. Note that clients cannot use SNAT addresses to connect directly to nodes.

This command also allows you to set properties on a SNAT. A SNAT defines the relationship between an externally visible IP address, or translated address, and a group of internal IP addresses, or originating address, of individual servers at your site.

Options

The **<orig addr>** variable specifies an originating IP address, that is, an address that is behind the Link Controller.

The **<trans addr>** variable specifies a translated IP address, that is, an address that is outside the Link Controller.

The **<ip addr>** variable can specify either an originating or a translated address.

The **<vlan name>** variable specifies the name of an existing VLAN on which access to the SNAT is enabled or disabled. By default, a SNAT is accessible on all VLANs.

The **<id>** variable specifies a unit ID, currently **1** or **2**. The default unit ID value is **1**.

The **<limit>** variable specifies a connection limit.

The **<seconds>** variable specifies the number of seconds for timeout.

The **auto** option enables SNAT automapping.

Defining a SNAT

SNATs map one or more originating addresses to a single translated address. Use the following syntax to define one or many originating addresses to translated address maps:

```
b snat map <orig addr> [<orig addr>... ] to <trans addr>
```

For example, the following command maps a SNAT, which has two clients, to a single translated address:

```
b snat map 192.140.100.10 192.140.100.20 to 192.168.11.22
```

You can set the following properties on a SNAT:

- A connection limit (**limit** option)
- A **tcp** timeout value (**timeout tcp** option)
- A **udp** timeout value (**timeout udp** option)
- Connection mirroring (**mirror** option)
- ARP enable or disable
- A VLAN deny access list

Deleting SNAT

Use the following command-line syntax to permanently delete one or more SNAT's from the Link Controller configuration:

```
b snat <ip addr>... <ip addr> delete
```

stp

```
b stp <stp_name> interfaces add <if_list> | all
b stp <stp_name> hello <interval>
b stp <stp_name> max_age <interval>
b stp <stp_name> forward_delay <interval>
b stp <stp_name> interfaces delete <if _list>
b stp <stp_name> enable | disable
```

The Link Controller provides Spanning Tree Protocol (STP) implementation for loop resolution in configurations where one or more external switches is connected in parallel with the Link Controller. This feature allows you to configure two or more interfaces on the platform as an STP domain. For interfaces in the STP domain, the spanning tree algorithm identifies the most efficient path between the network segments, and establishes the switch associated with that path as the **root**. Links forming redundant paths are shut down, to be re-activated only if the root fails.

The STP domain should contain all ports that are connected in parallel to an external switch where there are nodes on the link capable of generating or receiving traffic. You will want a second domain if there is an additional switch or switches connected in parallel with additional Link Controller interfaces.

Options

The **<stp_name>** variable specifies an arbitrary name for the spanning tree protocol (STP) domain.

The **<interface_name>** variable specifies an interface name, for example, **3.1**.

summary

b summary

Displays a summary of current usage statistics. The output display format for the **summary** command is shown in Figure A.5. You can find detailed descriptions of each of statistic displayed by the **summary** command in *Monitoring the Link Controller*, on page 6-2.

```

BIG-IP total uptime           = 1 (day) 4 (hr) 40 (min) 8 (sec)
BIG-IP total uptime (secs)    = 103208
BIG-IP total # connections    = 0
BIG-IP total # pkts           = 0
BIG-IP total # bits           = 0
BIG-IP total # pkts(inbound)  = 0
BIG-IP total # bits(inbound)  = 0
BIG-IP total # pkts(outbound) = 0
BIG-IP total # bits(outbound) = 0
BIG-IP error no nodes available      = 0
BIG-IP tcp port deny                 = 0
BIG-IP udp port deny                 = 0
BIG-IP virtual tcp port deny         = 0
BIG-IP virtual udp port deny         = 0
BIG-IP max connections deny          = 0
BIG-IP virtual duplicate syn ssl     = 0
BIG-IP virtual duplicate syn wrong dest = 0
BIG-IP virtual duplicate syn node down = 0
BIG-IP virtual maint mode deny       = 0
BIG-IP virtual addr max connections deny = 0
BIG-IP virtual path max connections deny = 0
BIG-IP virtual non syn               = 0
BIG-IP error not in out table         = 0
BIG-IP error not in in table          = 0
BIG-IP error virtual fragment no port = 0
BIG-IP error virtual fragment no conn = 0
BIG-IP error standby shared drop      = 0
BIG-IP dropped inbound                = 0
BIG-IP dropped outbound               = 0
BIG-IP reaped                        = 0
BIG-IP ssl reaped                     = 0
BIG-IP persist reaped                = 0
BIG-IP udp reaped                    = 0
BIG-IP malloc errors                  = 0
BIG-IP bad type                       = 0
BIG-IP mem pool total 96636758 mem pool used 95552 mem percent used 0.10

```

Figure A.5 The *summary* output display

For more information on the out put of the bigpipe summary command, see *Explanation of summary statistics*, on page 6-3.

---

trunk

```
b trunk <controlling_if> define <if_list>
b trunk [<controlling_if>] show [verbose]
b trunk [<controlling_if>] stats reset
```

The **trunk** command aggregates links (individual physical interfaces) to form a trunk. This link aggregation increases the bandwidth of the individual NICs in an additive manner. Thus, four fast Ethernet links, if aggregated, create a single 400 Mb/s link. The other advantage of link aggregation is link failover. If one link in a trunk goes down, traffic is simply redistributed over the remaining links.

A trunk must have a controlling link and acquires all the attributes of that controlling link from Layer 2 and above. Thus, the trunk automatically acquires the VLAN membership of the controlling link but does not acquire its media type and speed. Outbound packets to the controlling link are load balanced across all of the known-good links in the trunk. Inbound packets from any link in the trunk are treated as if they came from the controlling link.

A maximum of eight links may be aggregated. For optimal performance, links should be aggregated in powers of two. Thus, ideally, you will aggregate two, four, or eight links. Gigabit and fast ethernet links cannot be placed in the same trunk.

For more information on interface naming, refer to *Interface naming conventions*, on page 1-2.

Options

The **<controlling link>** variable specifies the name of the interface chosen to be the controlling link for the trunk. Any attributes of the controlling link at layer 2 and above, such as membership in a VLAN, apply to the trunk.

The **<link>** variable specifies an interface name, for example **3.1**.

The **show** option displays information and statistics for the trunk, on a single line.

The **<verbose>** option, used with the **show** option, displays the information and statistics for the trunk in more detailed format.

unit

```
b unit [show]
b unit peer [show]
```

The unit number on a system designates which virtual servers use a particular unit in an active-active redundant configuration. You can use the **bigpipe unit** command to display the unit number assigned to a particular Link Controller. For example, to display the unit number of the unit you are on, type the following command:

```
b unit show
```

To display the unit number of the other unit in a redundant system, type in the following command:

```
b unit peer show
```

◆ **Note**

If you use this command on a redundant system in active/standby mode, the active unit shows as unit 1 and 2, and the standby unit has no unit numbers.

◆ **Tip**

*The **bigpipe unit peer show** command is the best way to determine whether the respective state mirroring mechanisms are connected.*

verbose

```
b verbose virtual_server_udp_port_denial
b verbose virtual_server_tcp_port_denial
b verbose bigip_udp_ort_denial
b verbose bigip_tcp_port_denial
```

Used to modify the verbose log level. This command is an alternative to using the **bigpipe global verbose** command.

Table A.6 defines the command and shows the equivalencies.

b verbose command	b global verbose command
b verbose bigip_udp_port_denial Turns UDP port denial logging on . This logs UDP port denials to the Link Controller address.	b global verbose_log_level=1
b verbose bigip_tcp_port_denial Turns TCP port denial logging on . This logs TCP port denials to the Link Controller address.	b global verbose_log_level=2
b verbose virtual_server_udp_port_denial Turns virtual UDP port denial logging on . This logs UDP port denials to the virtual server address.	b global verbose_log_level=4
b verbose virtual_server_tcp_port_denial Turns virtual TCP port denial logging on . This logs TCP port denials to the virtual server address.	b global verbose_log_level=8
b verbose bigip_udp_port_denial b verbose bigip_tcp_port_denial b verbose bigip_udp_ort_denial b verbose bigip_tcp_port_denial Turns UDP and TCP port denial on for both virtual server and Link Controller addresses.	b global verbose_log_level=15

Table A.6 *bigpipe verbose and global verbose command equivalencies*

verify

```
b [log] verify <command...>
verify load [<filename> | -]
```

Parses the command line and checks syntax without executing the specified command. This distinguishes between valid and invalid commands

Use the **verify** command followed by a command that you want to validate:

```
b verify virtual 10.10.10.100:80 use pool my_pool
```

The command checks the syntax and logic, reporting any errors that would be encountered if the command executed.

Use the **verify** command together with the **load <filename>** command to validate the specified configuration file. For example, to check the syntax of the configuration file **/config/altbigpipe.conf**, use the following command:

```
b verify load /config/altbigip.conf
```

version

b version

Displays the version of the Link Controller operating system and the features enabled.

For example, for a Link Controller, the **bigpipe version** command displays the output shown in Figure A.6.

```
Product Code:
BIG-IP Link Controller

Enabled Features:
Statistics Journaling
IP Classifier
VS Availability Dependencies
Wide IP Persistence
3-DNS Server-Performance Load Balancing
Zone File Management
3-DNS ECV
BIG-IP Link Control
Dynamic Ratio Load Balancing
Gateway Failsafe
SNAT
Pools
Failover
Dynamic Load Balancing
Simple Persistence
ECV Transparent
Network Proximity Table
Internet Weather Map
Metrics and Limits
3-DNS Network-Performance Load Balancing
3-DNS Static Load Balancing
3-DNS EAV
3-DNS Link Control
HTTP Header Insert
HTTP Redirects
Static Load Balancing
NAT
Mirroring
Node HA
Destination Address Affinity
ECV
Health Check
```

Figure A.6 The *version* output display

virtual

```
b virtual <virt_ip>[:<service>] [unit <ID>] [netmask <ip>] [broadcast <ip>] use pool
  <pool_name>
b virtual <virt_ip>:<service> [/<bitmask>][unit <ID>] use pool <pool_name>
b virtual <virt_ip>[:<service>] [unit <ID>] [netmask <ip>] forward
b virtual <virt_ip>:<service> translate port enable | disable | show
b virtual <virt_ip>:<service> svc_down_reset enable | disable | show
b virtual <virt_ip>:<service> translate addr enable | disable | show
b virtual <virt_ip>:<service> lasthop pool <pool_name> | none | show
b virtual <virt_ip>:<service> mirror conn enable | disable | show
b virtual <virt_ip>:<service> conn rebind enable | disable | show
b virtual [<virt_ip>:<service>] stats reset
b virtual <virt_ip>:<service> accelerate enable | disable | show
b virtual <virt_ip>:<service> use pool <pool_name> accelerate disable
b virtual <virt_ip>:<service> vlans <vlan_list> disable | enable
b virtual <virt_ip>:<service> vlans show
b virtual <virt_ip> arp enable | disable | show
b virtual <virt_ip> any_ip enable | disable
b virtual <virt_ip> any_ip timeout <seconds>
b virtual <virt_ip> [:<service>] [...<virt_ip>[:<service>]] show
b virtual <virt_ip> [:<service>] [...<virt_ip>[:<service>]] enable | disable
b virtual <virt_ip>[:<service>] [ ... <virt_ip>[:<service>]] delete
b virtual <virt_ip>[:<service>] [... <virt_ip>[:<service>]] limit <max_conn>
b virtual <vlan_name>[:<service>]
b virtual <vlan_name> use pool <pool_name>
```

Creates, deletes, and displays information about virtual servers. This command also allows you to set properties on a virtual server, such as connection mirroring, connection limits, and timeouts.

A virtual server defines the relationships between an externally visible IP address that clients use to connect to your site, and the internal IP addresses of individual member servers that actually provide services for your site.

Options

The **<virtual addr>** variable specifies the IP address of the virtual server.

The **<virtual port>** variable specifies a port number or service name.

The **<bitmask>** variable specifies a number representing the bits that are the network part of the virtual IP address.

The **<vlan name>** variable specifies the name of an existing VLAN for which you want to enable or disable access. By default, a virtual server is accessible on all VLANs.

The **<id>** variable specifies a unit ID, currently **1** or **2**. The default value for the unit ID is **1**.

The **<ip addr>** variable specifies the IP address, of the form **10.20.30.40**.

The **<pool name>** variable specifies the name of an existing server pool.

The **translate port** option enables, disables, or shows port translation for a virtual server.

The **svc_down_reset** option enables, disables, or shows the ability of the Link Controller to automatically reset connections when a service becomes unavailable.

The **translate addr** option enables, disables, or shows address translation for a virtual server.

The **lasthop pool** option allows you to specify a pool to which to send connections back, instead of using the same router from which the connection was received.

The **mirror conn** option enables, disables, or shows the mirroring of connections in active/standby configurations.

The **conn rebind** option enables, disables, or shows dynamic connection rebinding.

The **stats reset** option resets statistics for a virtual server.

The **accelerate** option enables, disables, or shows FastFlow acceleration, that is, increased speed of packet flow for TCP connections when the packets are not fragmented.

The **arp** option causes the Link Controller to respond to ARP requests for the virtual server address and send a gratuitous ARP request for router table updates.

The **any_ip** option allows you to configure a virtual server to load balance IP traffic other than TCP and UDP traffic, such as IPSEC traffic.

Defining a virtual server using pools and rules

To associate a pool of members with a virtual server, use a command such as this:

```
b virtual 10.20.2.102:http use pool cgi_pool
```

Defining a virtual server with a wildcard port

Use the following syntax to define an individual virtual server and the node or nodes to which the virtual server maps. Note that this syntax allows wildcard ports:

```
b virtual <virt addr> use pool <pool name>
```

You can also create multiple wildcard servers, one per VLAN. To create a wildcard server for a VLAN, use the following syntax:

```
b virtual <vlan_name> use pool <pool_name>
```

Deleting a virtual server

Use the following syntax to permanently delete one or more virtual servers from the Link Controller configuration:

```
b virtual <virt addr>:<port>... <virt addr>:<port> delete
```


vlan

```

b vlan <name> rename <new_name>
b vlan <vlan_name> delete
b vlan <vlan_name> tag <tag_number>
b vlan <vlan_name> interfaces add [tagged] <if_list>
b vlan <vlan_name> interfaces delete <if_list>
b vlan <vlan_name> interfaces delete all
b vlan <vlan_name> interfaces show
b vlan <vlan_name> port_lockdown enable | disable
b vlan <vlan_name> bridging enable | disable
b vlan <vlangroup_name> proxy_forward enable | disable
b vlan <vlan_name> failsafe arm | disarm | show
b vlan <vlan_name> timeout <seconds> | show
b vlan <vlan_name> snat automap
b vlan show
b vlan <vlan_name> show
b vlan <vlan_name> rename <new_vlan_name>
b vlan <if_name> mac_masq <mac_addr> | show
b vlan <if_name> mac_masq 0:0:0:0:0
b vlan <vlan name> l2_agingtime <seconds>
b vlan <vlan name> fdb add <MAC address> interface <if_name>
b vlan <vlan name> fdb delete <MAC address> interface <if_name>
b vlan <vlan name> fdb static show
b vlan <vlan name> fdb dynamic show
b vlan <vlan name> fdb show

```

This command creates, displays and modifies settings for VLANs. VLANs are part of the base configuration.

When creating a VLAN, a tag value (VLAN ID) for the VLAN is automatically chosen unless it is specified on the command line. If a tag is specified on the command line and that tag number is **0**, the **vlan** command creates an empty VLAN.

A VLAN can have both tagged and untagged interfaces. An interface can be added to a single VLAN as an untagged interface. Also, an interface can be added to multiple VLANs as a tagged interface.

The **vlan** command defines VLANs, VLAN mappings, and VLAN properties. By default, each interface on a Link Controller is an untagged member of a VLAN. The lowest-numbered interface is assigned to the **external** VLAN, the interface on the main board is assigned to the **admin** VLAN, and all other interfaces are assigned to the **internal** VLAN.

Options

The **<vlan name>** variable specifies a VLAN name, 1-15 characters in length.

The **tag <tag number>** option specifies a valid VLAN tag number, in the range **0-4095**. Note that if **0** is specified as the tag number, the **vlan** command creates an empty VLAN.

The **interfaces add [tagged]** option specifies that the interfaces specified with the **<if_list>** argument are to be added to the specified VLAN, as either tagged or untagged interfaces.

The **interfaces delete** option deletes all interfaces for the specified VLAN.

The **<if_list>** variable specifies a list of interfaces to be added to a VLAN.

The **interfaces show all** option shows all interfaces for the specified VLAN.

The **port_lockdown** option enables or disables connection to the Link Controller through the specified VLAN.

The **proxy_forward** option enables or disables proxy forwarding, for purposes of L2 forwarding.

The **failsafe** option allows you to arm, disarm, or show the failsafe mechanism for redundant systems.

The **timeout <timeout>** option specifies a timeout value for the failsafe mechanism.

The **snat automap** option enables SNAT automapping for the specified VLAN.

The **rename <new_vlan_name>** option specifies the name to which you want to rename the specified VLAN.

The **<if_name>** variable specifies an interface name.

The **mac_masq <MAC address>** option specifies a MAC address, such as **0:a0:be:ef:1f:3a**, that will be shared by both Link Controller units in a redundant system.

The **l2_agingtime <seconds>** option specifies the value in seconds of L2 forwarding time.

The **fdb** option adds the specified interfaces as entries in the L2 forwarding table.

The **port_lockdown** option enables or disables connection to the Link Controller through the specified VLAN.

vlangroup

```

b vlangroup [<vlan name list>] [show]
b vlangroup [<vlan name list>] list
b vlangroup <vlan name list> delete
b vlangroup <vlan name> tag <number>
b vlangroup [<vlan name list>] tag [show]
b vlangroup [<vlan name list>] interfaces [show]
b vlangroup <vlan name> vlans add <vlan if name list>
b vlangroup <vlan name list> vlans delete <vlan if name list>
b vlangroup <vlan name list> vlans delete all
b vlangroup [<vlan name list>] vlans [show]
b vlangroup <vlan name list> port_lockdown enable | disable
b vlangroup [<vlan name list>] port_lockdown [show]
b vlangroup <vlan name list> proxy_forward enable | disable
b vlangroup [<vlan name list>] proxy_forward [show] vlangroup <vlan name list> failsafe
    arm
b vlangroup <vlan name list> failsafe disarm
b vlangroup [<vlan name list>] failsafe [show]
b vlangroup <vlan name list> timeout <number>
b vlangroup [<vlan name list>] timeout [show] vlangroup <vlan name list> snat automap
    enable (deprecated)
b vlangroup <vlan name list> snat automap disable (deprecated)
b vlangroup <vlan name list> mac_masq <MAC addr>
b vlangroup [<vlan name list>] mac_masq [show]
b vlangroup <vlan name list> fdb add <MAC addr> interface <if name>
b vlangroup <vlan name list> fdb delete <MAC addr> interface <if name>
b vlangroup [<vlan name list>] fdb [show]
b vlangroup [<vlan name list>] fdb show static
b vlangroup [<vlan name list>] fdb show dynamic
b vlangroup <vlan name> rename <vlan name>

```

The **vlangroup** command defines a VLAN group, which is a grouping of two or more VLANs belonging to the same IP network for the purpose of allowing L2 packet forwarding between those VLANs.

The VLANs between which the packets are to be passed must be on the same IP network, and they must be grouped using the **vlangroup** command. For example:

```
b vlangroup network11 { vlans add internal external }
```

A self IP address must be assigned to the VLAN group using the following command:

```
b self <ip_addr> vlan network11
```

L2 forwarding must be enabled for the VLAN group using the VLAN **proxy_forward** attribute. This attribute is enabled by default when the VLAN group is enabled.

Options

For a description of the **bigpipe vlangroup** command options, see *Options*, on page A-60.



Glossary

A record

The **A** record is the ADDRESS resource record that a Link Controller returns to a local DNS server in response to a name resolution request. The **A** record contains a variety of information, including one or more IP addresses that resolve to the requested domain name.

active unit

In a redundant system, an active unit is a system that currently load balances name resolution requests. If the active unit in the redundant system fails, the standby unit assumes control and begins to load balance requests.

alternate method

The alternate method specifies the load balancing mode that the Link Controller uses to pick a virtual server if the preferred method fails. See also *preferred method*.

Any IP Traffic

Any IP Traffic is a feature that allows the Link Controller to load balance protocols other than TCP and UDP.

big3d agent

The **big3d** agent is a monitoring agent that collects metrics information about server performance and network paths between a Link Controller and a specific local DNS server. The Link Controller uses the information collected by the **big3d** agent for dynamic load balancing.

bigpipe

The **bigpipe** utility provides command line access to the Link Controller.

BIG/stat

BIG/stat is a statistical monitoring utility that ships on the Link Controller. This utility provides a snap-shot of statistical information.

BIG/top

BIG/top is a statistical monitoring utility that ships on the Link Controller. This utility provides real-time statistical information.

BIND (Berkeley Internet Name Domain)

BIND is the most common implementation of the Domain Name System (DNS). BIND provides a system for matching domain names to IP addresses. For more information, refer to <http://www.isc.org/products/BIND>.

chain

A chain is a series of filtering criteria used to restrict access to an IP address. The order of the criteria in the chain determines how the filter is applied, from the general criteria first, to the more detailed criteria at the end of the chain.

completion rate

The completion rate is the percentage of packets that a server successfully returns during a given session.

Completion Rate mode

The Completion Rate mode is a dynamic load balancing mode that distributes connections based on which network path drops the fewest packets, or allows the fewest number of packets to time out.

default VLANs

The Link Controller is configured with two default VLANs, one for each interface. One default VLAN is named *internal* and one is named *external*. See also *VLAN*.

default wildcard virtual server

A default wildcard virtual server has an IP address and port number of **0.0.0.0:0**, or ***:*** or **"any":"any"**. This virtual server accepts all traffic that does not match any other virtual server defined in the configuration.

domain name

A domain name is a unique name that is associated with one or more IP addresses. Domain names are used in URLs to identify particular Web pages. For example, in the URL **http://www.f5.com/index.html**, the domain name is **f5.com**.

dynamic load balancing

Dynamic load balancing modes use current performance information from each node to determine which node should receive each new connection. The different dynamic load balancing modes incorporate different performance factors such as current server performance and current connection load.

dynamic load balancing modes

Dynamic load balancing modes base the distribution of name resolution requests to virtual servers on live data, such as current server performance and current connection load.

Dynamic Ratio load balancing mode

Dynamic Ratio mode is like Ratio mode (see *Ratio mode*), except that ratio weights are based on continuous monitoring of the links and are therefore continually changing.

external monitor

An external monitor is a user-supplied health monitor. See also, *health check*, *health monitor*.

external VLAN

The external VLAN is a default VLAN on the Link Controller. In a basic configuration, this VLAN has the administration ports locked down. In a normal configuration, this is typically a VLAN on which external clients request connections to internal servers.

fail-over

Fail-over is the process whereby a standby unit in a redundant system takes over when a software failure or a hardware failure is detected on the active unit.

fail-over cable

The fail-over cable directly connects the two controller units together in a redundant system.

Fastest mode

Fastest mode is a dynamic load balancing mode that bases connection distribution on which server currently exhibits the fastest response time to node pings.

FDDI (Fiber Distributed Data Interface)

FDDI is a multi-mode protocol used for transmitting data on optical-fiber cables at speeds up to 100 Mbps.

floating self IP address

A floating self IP address is an additional self IP address for a VLAN that serves as a shared address by both units of a Link Controller redundant system.

Global Availability mode

Global Availability mode is a static load balancing mode that bases connection distribution on a particular server order, always sending a connection to the first available server in the list. This mode differs from Round Robin mode in that it searches for an available server always starting with the first server in the list, while Round Robin mode searches for an available server starting with the next server in the list (with respect to the server selected for the previous connection request).

health check

A health check is a Link Controller feature that determines whether a node is **up** or **down**. Health checks are implemented through health monitors. See also *health monitor* and *external monitor*.

health monitor

A health monitor checks a node to see if it is **up** and functioning for a given service. If the node fails the check, it is marked **down**. Different monitors exist for checking different services. See also *health check* and *external monitor*.

host

A host is a network server that manages one or more virtual servers that the Link Controller uses for load balancing.

ICMP (Internet Control Message Protocol)

ICMP is an Internet communications protocol used to determine information about routes to destination addresses, such as virtual servers managed by Link Controller systems.

interface

The physical port on a Link Controller is called an interface. See also *link*.

internal VLAN

The internal VLAN is a default VLAN on the Link Controller. In a basic configuration, this VLAN has the administration ports open. In a normal configuration, this is a network interface that handles connections from internal servers.

IPSEC

IPSEC (Internet Security Protocol) is a communications protocol that provides security for the network layer of the Internet without imposing requirements on applications running above it.

iQuery

The iQuery protocol is used to exchange information between Link Controllers. The iQuery protocol is officially registered with IANA for port 4353, and works on UDP and TCP connections.

Kilobytes/Second mode

The Kilobytes/Second mode is a dynamic load balancing mode that distributes connections based on which available server currently processes the fewest kilobytes per second.

last hop

A last hop is the final hop a connection took to get to the Link Controller. You can allow the Link Controller to determine the last hop automatically to send packets back to the device from which they originated. You can also specify the last hop manually by making it a member of a last hop pool.

Least Connections mode

Least Connections mode is a dynamic load balancing mode that bases connection distribution on which server currently manages the fewest open connections.

link

A link is a physical interface on the Link Controller connected to another physical interface in a network.

link aggregation

The link aggregation feature allows you to combine a number of links together to act as one interface.

Link Controller active unit

In a redundant system, the Link Controller active unit is the controller that currently load balances connections. If the active unit in the redundant system fails, the standby unit assumes control and begins to load balance connections.

Link Controller web server

The Link Controller web server runs on a Link Controller and hosts the Configuration utility.

local DNS

A local DNS is a server that makes name resolution requests on behalf of a client. With respect to the Link Controller, local DNS servers are the source of name resolution requests. Local DNS is also referred to as LDNS.

load balancing mode

A load balancing mode is a particular method of determining how to distribute connections across links.

loopback adapter

A loopback adapter is a software interface that is not associated with an actual network card. The nPath routing configuration requires you to configure loopback adapters on servers.

MAC (Media Access Control)

MAC is a protocol that defines the way workstations gain access to transmission media, and is most widely used in reference to LANs. For IEEE LANs, the MAC layer is the lower sublayer of the data link layer protocol.

MAC address

A MAC address is used to represent hardware devices on an Ethernet network.

member

Member is a reference to a node when it is included in a particular pool. Pools typically include multiple member nodes.

metrics information

Metrics information is the data that is typically collected about the paths between Link Controllers and local DNS servers. Metrics information is also collected about the performance and availability of virtual servers. Metrics information is used for load balancing, and it can include statistics such as round trip time, packet rate, and packet loss.

MindTerm SSH

MindTerm SSH is the third-party application on Link Controller systems that uses SSH for secure remote communications. SSH encrypts all network traffic (including passwords) to effectively eliminate eavesdropping, connection hijacking, and other network-level attacks. SSH also provides secure tunneling capabilities and a variety of authentication methods.

minimum active members

The minimum active members is the number of members that must be active in a priority group in order for the Link Controller to send its requests to that group. If the number of active members falls below this number, requests are sent to the next highest priority group (the priority group with the next lowest priority number).

monitor

The Link Controller uses monitors to determine whether nodes are **up** or **down**. There are several different types of monitors and they use various methods to determine the status of a server or service.

monitor destination IP address or IP address:port

The monitor destination IP address or address:port for a user defined monitor is used mainly for setting up a node alias for the monitor to check. All nodes associated with that monitor will be marked down if the alias node (destination IP address:port) is marked down. See also *node alias*.

monitor instance

You create a monitor instance when a health monitor is associated with a node, node address, or port. It is the monitor instance that actually performs the health check, not the monitor.

monitor template

A monitor template is a system-supplied health monitor that is used primarily as a template to create user-defined monitors, but in some cases can be used as is. The Link Controller includes a number of monitor

templates, each specific to a service type, for example, HTTP and FTP. The template has a template type that corresponds to the service type and is usually the name of the template.

name resolution

Name resolution is the process by which a name server matches a domain name request to an IP address, and sends the information to the client requesting the resolution.

name server

A name server is a server that maintains a DNS database, and resolves domain name requests to IP addresses using that database.

named

The **named** daemon manages domain name server software.

NAT (Network Address Translation)

A NAT is an alias IP address that identifies a specific node managed by the Link Controller to the external network.

node

A node is a specific combination of an IP address and port (service) number associated with a server in the array that is managed by the Link Controller.

node address

A node address is the IP address associated with one or more nodes. This IP address can be the real IP address of a network server, or it can be an alias IP address on a network server.

node alias

A node alias is a node address that the Link Controller uses to verify the status of multiple nodes. When the Link Controller uses a node alias to check node status, it pings the node alias. If the Link Controller receives a response to the ping, it marks all nodes associated with the node alias as **up**. If the controller does not receive a response to the ping, the it marks all nodes associated with the node alias as **down**.

node port

A node port is the port number or service name that is hosted by a specific node.

node status

Node status indicates whether a node is **up** and available to receive connections, or **down** and unavailable. The Link Controller uses the node ping and health check features to determine node status.

Observed mode

Observed mode is a dynamic load balancing mode that bases connection distribution on a combination of two factors: the server that currently hosts the fewest connections and also has the fastest response time.

packet rate

The packet rate is the number of data packets per second processed by a server.

Packet Rate mode

The Packet Rate mode is a dynamic load balancing mode that distributes connections based on which available server currently processes the fewest packets per second.

path

A path is a logical network route between a Link Controller and a local DNS server.

path probing

Path probing is the collection of metrics data, such as round trip time and packet rate, for a given path between a requesting LDNS server and a Link Controller.

performance monitor

A performance monitor gathers statistics and checks the state of a target device.

persistence

Persistence is a series of related connections received from the same client, having the same session ID. When persistence is turned **on**, a controller sends all connections having the same session ID to the same node, instead of load balancing the connections.

picks

Picks represent the number of times a particular virtual server is selected to receive a load balanced connection.

pool

A pool is composed of a group of network devices (called members). The Link Controller load balances requests to the nodes within a pool based on the load balancing method and persistence method you choose when you create the pool or edit its properties.

pool ratio

A pool ratio is a ratio weight applied to pools in a wide IP. If the Pool LB mode is set to Ratio, the Link Controller uses each pool for load balancing in proportion to the weight defined for the pool.

port

A port is represented by a number that is associated with a specific service supported by a host. Refer to the Services and Port Index for a list of port numbers and corresponding services.

port-specific wildcard virtual server

A port-specific wildcard virtual server is a wildcard virtual server that uses a port number other than 0. See *wildcard virtual server*.

port mirroring

Port mirroring is a feature that allows you to copy traffic from any port or set of ports to a single, separate port where a sniffing device is attached.

Predictive mode

Predictive mode is a dynamic load balancing mode that bases connection distribution on a combination of two factors: the server that currently hosts the fewest connections, and also has the fastest response time. Predictive mode also ranks server performance over time, and passes connections to servers which exhibit an improvement in performance rather than a decline.

preferred method

The preferred method specifies the first load balancing mode that the Link Controller uses to load balance a resolution request. See also *alternate method*.

QOS equation

The QOS equation is the equation on which the Quality of Service load balancing mode is based. The equation calculates a score for a given path between a link and a local DNS server. The Quality of Service mode distributes connections based on the best path score for an available link. You can apply weights to the factors in the equation, such as round trip time and completion rate.

Quality of Service load balancing mode

The Quality of Service load balancing mode is a dynamic inbound load balancing mode that bases connection distribution on a configurable combination of the packet rate, completion rate, round trip time, hops, virtual server capacity, kilobytes per second, and topology information.

rate class

You create a rate filter from the Configuration utility or command line utility. When you assign a rate class to a rate filter, a rate class determines the volume of traffic allowed through a rate filter. See also *rate filter*.

rate filter

Rate filters consist of a basic filter with a rate class. Rate filters are a type of extended IP filter. They use the same IP filter method, but they apply a rate class, which determines the volume of network traffic allowed through the filter. See also *rate class*.

ratio

A ratio is a parameter that assigns a weight to a virtual server for load balancing purposes.

Ratio mode

The Ratio load balancing mode distributes connections across an array of virtual servers in proportion to the ratio weights assigned to each individual virtual server.

redundant system

Redundant system refers to a pair of controllers that are configured for fail-over. In a redundant system, there are two controller units, one running as the active unit and one running as the standby unit. If the active unit fails, the standby unit takes over and manages connection requests.

remote administrative IP address

A remote administrative IP address is an IP address from which a controller allows shell connections, such as Telnet or SSH.

resource record

A resource record is a record in a DNS database that stores data associated with domain names. A resource record typically includes a domain name, a TTL, a record type, and data specific to that record type. See also *A record*.

RFC 1918 addresses

An RFC 1918 address is an address that is within the range of non-routable addresses described in the IETF RFC 1918.

Round Robin mode

Round Robin mode is a static load balancing mode that bases connection distribution on a set server order. Round Robin mode sends a connection request to the next available server in the order.

round trip time (RTT)

Round trip time is the calculation of the time (in microseconds) that a local DNS server takes to respond to a ping issued by the **big3d** agent running on a link. The Link Controller takes RTT values into account when it uses dynamic load balancing modes.

Round Trip Time mode

Round Trip Time mode is a dynamic load balancing mode that bases connection distribution on which virtual server has the fastest measured round trip time between the link and the local DNS server.

self IP address

Self IP addresses are the IP addresses owned by the Link Controller that you use to access the internal and external VLANs.

service

Service refers to services such as TCP, UDP, HTTP, and FTP.

Setup utility

The Setup utility is a utility that takes you through the initial system configuration process. The Setup utility runs automatically when you turn on a system for the first time.

SNAT (Secure Network Address Translation)

A SNAT is a feature you can configure on the Link Controller. A SNAT defines a routable alias IP address that one or more nodes can use as a source IP address when making connections to hosts on the external network.

SNAT automap

This feature allows the Link Controller to perform a SNAT automatically on any connection that is coming from the system's internal VLAN. It is easier to use than traditional SNATs and solves certain problems associated with traditional SNATs.

SNMP (Simple Network Management Protocol)

SNMP is the Internet standard protocol, defined in STD 15, RFC 1157, developed to manage nodes on an IP network.

source processing

Source processing means that the interface rewrites the source of an incoming packet.

spanning tree protocol (STP)

Spanning tree protocol is a protocol that provides loop resolution in configurations where one or more external switches is connected in parallel with the Link Controller.

SSH

SSH is a protocol for secure remote login and other secure network services over a non-secure network.

SSL gateway

An SSL gateway is a gateway for decrypting HTTPS requests to an HTTP server and encrypting the reply.

standby unit

A standby unit in a redundant system is a unit that is always prepared to become the active unit if the active unit fails.

state mirroring

State mirroring is a feature on the Link Controller that preserves connection and persistence information in a Link Controller redundant system.

static load balancing modes

Static load balancing modes base connection distribution on a pre-defined list of criteria and virtual server availability; they do not take current server performance or current connection load into account.

sticky mask

A sticky mask is a special IP mask that you can configure on the Link Controller. This mask optimizes sticky persistence entries by grouping more of them together.

tagged VLAN

You can define any interface as a member of a tagged VLAN. You can create a list of VLAN tags or names for each tagged interface.

transparent node

A transparent node appears as a router to other network devices, including the Link Controller.

trunk

A trunk is a combination of two or more interfaces and cables configured as one link. See also *link aggregation*.

unavailable

The **unavailable** status is used for links and virtual servers. When a link or virtual server is **unavailable**, the Link Controller does not use it for load balancing.

unknown

The **unknown** status is used for links and virtual servers. When a link or virtual server is new to the Link Controller and does not yet have metrics information, the Link Controller marks its status as **unknown**. The Link Controller can use unknown servers for load balancing, but if the load balancing mode is dynamic, the Link Controller uses default metrics information for the unknown server until it receives live metrics data.

up

The **up** status is used for links and virtual servers. When a link or virtual server is **up**, the link or virtual server is available to respond to process connections.

user-defined monitor

A user-defined monitor is a custom monitor configured by a user, based on a system-supplied monitor template. For some monitor types, you must create a user-defined monitor in order to use them. For all monitor types, you must create a user-defined monitor to change system supplied monitor default values.

virtual address

A virtual address is an IP address associated with one or more virtual servers managed by the Link Controller.

virtual port

A virtual port is the port number or service name associated with one or more virtual servers managed by the Link Controller. A virtual port number should be the same TCP or UDP port number to which client programs expect to connect.

virtual server

Virtual servers are a specific combination of virtual address and virtual port, associated with a content site that is managed by a Link Controller or other type of host server.

VLAN

VLAN stands for virtual local area network. A VLAN is a logical grouping of network devices. You can use a VLAN to logically group devices that are on different network segments.

VLAN name

A VLAN name is the symbolic name used to identify a VLAN. For example, you might configure a VLAN named **marketing**, or a VLAN named **development**. See also *VLAN*.

watchdog timer card

A watchdog timer card is a hardware device that monitors the Link Controller for hardware failure.

wide IP

A wide IP is a collection of one or more domain names that maps to one or more groups of virtual servers managed by Link Controllers. The Link Controller load balances name resolution requests across the virtual servers that are defined in the wide IP that is associated with the requested domain name.

wildcard virtual server

A wildcard virtual server is a virtual server that uses an IP address of **0.0.0.0**, ***** or **"any"**. A wildcard virtual server accepts connection requests for destinations outside of the local network. Wildcard virtual servers are included only in Transparent Node Mode configurations.



Index

/config/aliases file 7-11
 /config/bigip.conf A-43
 /config/gated.conf file 7-7
 /config/routes file 7-8
 /config/sendmail.cf file 7-11
 /etc/hosts file 7-1, 7-2, 7-9
 /etc/hosts.allow file 9-6
 /etc/irs.conf file 7-9
 /etc/resolv.conf file 7-9
 /etc/services file 2-62
 /etc/snmptrap.conf file 9-8
 /etc/syslog.conf file 6-17
 -? command A-3

A

access levels 6-22
 active units 8-2
 active/standby
     setting preferred active unit 8-10
 adding to /config/routes 7-8
 address mappings
     configuring 2-67
 Address Resolution Protocol (ARP) 7-5
 address translation
     disabling 2-36
 Administrator Kit, description Intro-2
 agent types 2-17
 arm fail-safe 8-4
 ARP 8-3
 ARP cache, updating 6-18
 ARP requests
     disabling for SNATs 2-73
 audit trails, for reset events 6-9
 authoritative name servers 7-10

B

base configuration, about 1-1
 BIG/db database keys 6-1
 BIG/stat utility 6-1, 6-23
     command line options 6-23
     customizing 6-23
 big3d agent
     about 3-1
     and dynamic load balancing 5-4
     collecting path data 3-3
     configuring 3-1
     default settings 3-1
     using factories 3-2
 BIG-IP product family Intro-4
 bigip.bonfire_compatibility_mode variable A-9
 bigip.fastest_max_idle_time variable A-9
 bigip.memory_reboot_percent variable A-11
 bigip.open_3dns_lockdown_ports variable A-9, A-11

bigip.open_ftp_ports variable A-12
 bigip.open_rsh_ports variable A-12
 bigip.open_ssh_port variable A-12
 bigip.open_telnet_port variable A-11
 bigip.persist_map_proxies A-13
 bigip.persist_on_any_port_same_virtual variable A-14
 bigip.persist_on_any_virtual variable A-14
 bigip.persist_time_used_as_limit variable A-13
 bigip.verbose_log_level variable A-14
 bigip.virtualnoarp variable A-15
 bigip.webadmin_port variable A-15
 bigpipe commands A-1
     conn A-6
     failover A-7, A-8
     global A-9
     help A-3, A-16
     interface A-17
     maint A-21
     mirror A-23
     mirroring 8-5
     monitor A-25
     -n A-32
     nat A-33
     node A-35
     ratio A-41
     reset A-42
     save A-43
     service A-46
     snat A-47
     stp A-49
     summary A-50
     trunk A-51
     unit A-52
     verbose A-53
     virtual A-56
     vlan A-59
 bigpipe utility Intro-1, 6-1
 bigstat command

    See BIG/stat utility

bigtop utility Intro-1
 bit statistics 6-1
 bit status 6-14
 bitmask 2-75
 broadcast 2-51
 browser, supported versions Intro-1
 byte counters, resetting 6-9

C

caching proxy servers 2-25
 client IP addresses
     in headers 2-33
     preserving 2-33
     tracking connections for 2-22
 client_addr variable 2-33
 client_port variable 2-33

- clients
  - and secure connections 2-32
- coefficient variable 2-19
- command line access 7-4
- config command A-9
- configuration settings A-43
- Configuration utility
  - configuring a pool 2-3
  - described 6-1
  - web-based Intro-1
- configurations
  - clearing memory A-42
- conn command 6-21, A-6
- connection count counter, resetting 6-9
- connection limits
  - for nodes 2-60
  - for ports 2-62, A-46
  - for virtual servers 2-54
- connection requests 2-41
- connection statistics 6-1
- connection status, displaying 6-12
- connection timeout 2-63
- connections 2-58
  - and Maintenance mode 6-18
  - and SNATs 2-70
  - client A-6
  - distributing by priority 2-21
  - forwarding 2-38
  - FTP A-12
  - increasing number of 2-69
  - long-lived 8-5
  - monitoring concurrent 6-1
  - Telnet A-11
  - to external clients 2-65
  - viewing 6-20
- content servers
  - adding to hosts file 7-2
  - default route 7-6
- CPU metrics
  - gathering 2-17, 2-19

D

- data collection agents 2-17
- default gateway pool
  - changing 7-5
  - default 7-5
- default route
  - for content servers 7-6
  - for external gateways 7-5
- default routers 2-78
- default routes 8-3
  - See also default gateway pool 7-5
- default SNATs
  - automapped 2-69
  - defined 2-67

- manual 2-69
- Destination address affinity 2-25
- destination IP addresses
  - and persistence 2-25
- disable keyword 6-19
- disk metrics
  - gathering 2-17, 2-19
- DNS configuration
  - configuring proxy 7-9
  - converting from rotary 7-10
  - converting from Round Robin 7-10
  - converting from round robin 7-10
  - resolving names 7-9
  - zone tables 7-10
- domain names 7-10
- dropped connections, viewing 6-20
- duplex mode 1-3
- dynamic load balancing
  - about 5-2
  - and big3d agents 5-4
- dynamic load balancing modes
  - and Internet Link Evaluator 4-1
  - types 5-4
- Dynamic Ratio
  - about 5-10
  - and Quality of Service mode 5-8
  - configuring 5-10
  - using with Quality of Service mode 5-10
- Dynamic Ratio mode
  - configuring RealSystem Server for 2-10
  - configuring servers for 2-10
  - configuring SNMP for 2-17
  - configuring WMI for 2-13
  - setting 2-20

E

- e-commerce site
  - and persistence 2-21
  - configuring wide IPs 5-13
- elapsed time, viewing 6-20
- email configuration 7-10, 7-11
- email, sending 6-16
- enable keyword 6-19
- encrypted communications 7-3
- external interfaces 2-75, A-33
- external monitor 2-89
- external network. See external interfaces
- external VLANs 7-2

F

- factories, types of
  - hops 3-2
  - probing 3-2
  - SNMP 3-2
- failover command A-7, A-8

- fail-over process 8-4
- fail-safe
  - features of 8-7
  - settings 8-1
- fallback hosts 2-28
- fallback ports 2-29
- fallback protocols 2-29
- fallback URIs 2-29
- format strings
  - example 2-30
  - for redirection 2-29
  - listed 2-30
- forward attribute 2-38
  - example 2-39
- forwarding pool
  - and default gateway pool 2-38
  - defined 2-38
- forwarding virtual servers
  - and SNATs 2-70
  - defined 2-38, 2-47
- FQDNs
  - and DNS resolution 7-9
  - and redirection 2-28
- FTP A-12
  - configuring wide IPs 5-13
- ftp monitor 2-84
- Full Read/Write access level 6-22

G

- GateD
  - configuration file 7-8
  - documentation 7-8
  - dynamic routing 7-7
- gateway command A-10
- gateway fail-safe 8-7
- global command 6-12, A-9
- global statistics, resetting 6-9

H

- h command A-16
- hardware maintenance, performing 6-18
- hard-wired fail-over 8-9
- header insertion
  - and SSL proxies 2-34
  - attributes 2-33
  - syntax 2-33
- health monitor templates
  - external 2-89
  - ftp 2-84
  - http 2-83
  - https 2-84
  - icmp 2-81
  - imap 2-87
  - ldap 2-88
  - nntp 2-86

- pop3 2-85
- radius 2-88
- selecting 2-19
- smtp 2-85
- snmp_dca 2-86
- snmp_dca_base 2-86
- sql 2-87
- tcp 2-82
- tcp_echo 2-82
- health monitors 2-79, 2-102
  - associating with a node 2-96
  - associating with nodes 2-17, 2-20
  - configuring 2-17, 2-89
  - logical grouping in 2-99
  - transparent mode in 2-99
- help command A-3, A-16
- high availability 5-13
- host names
  - redirecting 2-28
- hosts file, adding host names 7-1
- http monitor 2-83
- HTTP requests
  - inserting headers 2-33
- https monitor 2-84

I

- icmp monitor 2-81
- IIS servers
  - for rewriting redirections 2-32
- illegal connection attempt statistics, viewing 6-21
- imap monitor 2-87
- inbound traffic
  - accepting 2-65
- interface access methods 1-7
- interface cards 6-15, 8-3
- interface command A-17
- interface media type 1-3
- interface mode 1-3
- interface naming convention 1-2
- interface settings
  - displaying 1-2
- interface status
  - displaying 1-2
- interfaces
  - and multiple VLANs 1-7
- internal interfaces 2-74
- internal network. See internal interfaces
- internal VLANs 7-2
- Internet Link Evaluator
  - about 4-1
- IP addresses A-33
  - and redirection 2-28
  - defining Intro-1
  - destination 2-41
  - for clients 2-24

- specifying for SNATs 2-67
- IP forwarding 2-62
- IP packet filter statistics, viewing 6-21
- IP packet filters
  - and illegal connection attempts 6-21
- ip_tos variable 2-34
- ISAPI filters 2-32

L

- LDAP authentication 7-14
- LDAP monitor 2-88
- LDNS round robin
  - about 5-13
- less file page utility 6-17
- link aggregation and fail-over 1-19
- Link Controller system log 6-21
- Link Controller web server
  - changing passwords 6-21
  - deleting account 6-22
- link_qos variable 2-34
- load balancing
  - configuring inbound 5-11
  - transparent nodes 2-42
- load balancing modes
  - Completion Rate 5-5
  - Global Availability 5-3
  - Hops 5-5
  - Kilobytes/Second 5-5
  - Least Connections 5-6
  - Packet Rate 5-6
  - Quality of Service 5-7
  - Random 5-3
  - Ratio 5-3
  - Round Robin 5-3
  - Round Trip Times (RTT) 5-6
  - setting 2-8
  - Static Persist 5-4
  - types 2-5
  - VS Capacity 5-11
- load balancing pools
  - about A-37
  - defining 2-3
- load calculation 2-19
- log files, viewing 6-21
- log messages, samples of 6-17
- logging 6-16, A-15, A-53

M

- MAC addresses 1-16
- MAC masquerade 1-16
- maint command 6-18, A-21
- Maintenance mode, activating 6-18
- masks
  - for simple persistence 2-23, 2-24
- media access control. See MAC addresses

- media types 1-3
- member node status, displaying 6-12
- memory metrics
  - gathering 2-17, 2-19
- messages
  - gateway fail-safe 8-8
- Microsoft Internet Explorer Intro-1
- min_active_members value 2-21
- MindTerm SSH Console
  - about 7-2
  - using 7-2
- mirroring
  - described 8-4
  - for SNAT connections 2-73
  - global 8-5
- mirroring commands 8-5
- monitor command A-25
- monitoring
  - command-line utilities Intro-1
  - with Configuration utility Intro-1
- monitors
  - associating with nodes 2-20
- multiple services
  - configuring ports 5-13
- multiple wildcard servers 2-45

N

- n command A-32
- name servers 7-10
- naming conventions
  - for interfaces 1-2
- nat command 2-74, 6-11, A-33
- NAT statistics
  - resetting 6-9, 6-11
  - viewing 6-20
- NAT status
  - displaying 6-14
  - viewing 6-23
- NATs
  - about 2-74
  - defining 2-75
  - disabling 2-36
- netmask 2-51, 2-75
- Netscape Navigator Intro-1
- network address translation (NAT). See nat command
- network traffic 8-3
- network traffic statistics, viewing 6-13, 6-14
- network-based fail-over 8-9
- nntp monitor 2-86
- node address statistics
  - resetting 6-9, 6-10
  - viewing 6-1, 6-20
- node addresses
  - displaying status 6-14
  - enabling and disabling 6-20

- ratio weights 2-9
- removing from service 6-18, 6-20
- node command 2-59, 6-10, A-35
- node configuration 2-74
- node server statistics, resetting 6-9, 6-10
- node statistics
  - monitoring 6-1
  - viewing 6-1, 6-20
- node status
  - displaying 6-14
  - viewing 6-23
- nodes
  - and SNATs 2-68, 2-69
  - as pool members 2-5
  - connection limits 2-60
  - connections A-33
  - enabling and disabling 6-20
  - receiving connections 2-5
  - removing from service 6-18, 6-20
  - viewing 6-20

O

- open FTP ports A-12
- openssl.conf file 7-2

P

- packet counters, resetting 6-9
- packet statistics 6-1
- packet status 6-14
- packets
  - access to VLANs 1-7
  - monitoring 6-1
  - viewing 6-20
- pager notifications, activating 6-16
- Partial Read/Write access level 6-22
- passwords
  - and Link Controller web server 6-22
  - changing 6-21
- performance statistics
  - displaying 6-2
  - summary table 6-3
- persistence
  - and mirroring 8-5
  - and multiple services A-14
  - and proxy maps A-13
  - and timers A-13
  - and virtual addresses A-14
  - conditions for 2-22, 2-26
  - defined 2-22
  - for connections 2-58
  - need for 2-21
  - See also destination address affinity
  - using persist mask 2-24
- persistence timer 2-23
- persistence types

- listed 2-22
- Pinger log 6-21
- pool
  - in a wide IP 5-12
- pool attributes
  - configuring 2-2
  - listed 2-2
- pool command 2-2
- pool elements
  - listed 2-3
- pool members 2-5
- pool naming 2-5
- pools
  - and redirection attribute 2-31
  - and SNAT/NAT connections 2-36, 2-37
  - defined 2-1, 2-2
  - for configuring header insertion 2-33
  - modifying 2-2, 2-17
  - See also load balancing pools
- pop3 monitor 2-85
- port mirror command A-23
- port mirroring A-23
- port numbers
  - redirecting 2-28
- port statistics, resetting 6-11
- ports
  - RSH A-9, A-12
  - translation properties 2-51
  - wildcard 2-42
- Powering down 6-17
- priority member activation
  - about 2-8
  - example 2-21
- priority numbers
  - assigning 2-21
- procedures
  - configuring gateway fail-safe 8-7
  - configuring NATs 2-75
  - configuring network fail-over 8-9
  - configuring persistence for virtual servers 2-23
  - configuring SNAT address mappings 2-67, 2-68, 2-70, 2-71
  - configuring SNAT global properties 2-66
  - defining virtual server mappings 2-43
  - setting IP forwarding 2-78
  - turning off port translation 2-44
- proxy, DNS. See DNS configuration

Q

- QOS coefficients
  - about 5-8
  - and wide IPs 5-10
  - considerations 5-8
- QOS equation
  - modifying 5-9

- Quality of Service mode
  - customizing 5-8
  - understanding QOS coefficients 5-8
  - using Dynamic Ratio 5-8, 5-10
- R
- RADIUS authentication
  - about 7-14
  - challenge-response authentication 7-18
- radius monitor 2-88
- rate filter statistics, viewing 6-21
- ratio command A-41
- Ratio mode 2-21
- ratio weights 2-9, 2-10
  - setting 2-9
- Read Only access level 6-22
- RealSystem Server
  - configuring for dynamic ratio load balancing 2-10
- redirectfilter file 2-32
- redirection
  - and rewriting 2-32
  - defined 2-28
  - example 2-29
- redundant systems
  - about 8-1
  - active unit 8-2
  - advanced features of 8-1
  - configuring gateway fail-safe A-10
  - configuring hop pools 2-56
  - displaying unit number A-52
  - fail-over A-7, A-8
  - fail-safe interfaces 8-4
  - gateway fail-safe. See gateway fail-safe
  - mirroring 2-73
  - See also mirroring
  - See also network-based fail-over
  - sharing MAC addresses 1-17
  - standby unit 8-2
- requests
  - inserting headers 2-33
- reset command A-42
- rewriting methods 2-32
- root password
  - defining Intro-1
- rotary DNS. See DNS configuration
- Round Robin 7-10
- Round Robin mode 2-5, 7-10
- routable IP addresses
  - creating 2-65
- route configurations
  - examples 7-6
  - from content servers 7-6
  - on different logical networks 7-6
  - overview 7-5
  - with GateD 7-7
- routers, host names 7-1
- routing table 2-78
- RSH A-12
- rule elements
  - listed and described 2-57
- rule variables
  - for header insertion 2-33
- S
- save command A-43
- saving A-43
- secure network address translation (SNAT)
  - See SNATs
- secure shell A-12
- security
  - and illegal connection attempts 6-21
  - changing passwords 6-22
- self IP addresses
  - enabling snat automap on 2-71
- sendmail 7-11
- serial terminal
  - configured as console 7-12, 7-13
  - configured as terminal 7-12
  - configuring in addition to console 7-13
  - forcing to be console 7-14
- server types 2-40
- server_addr variable 2-34
- server_port variable 2-34
- servers
  - creating multiple wildcard 2-45
- service checks
  - troubleshooting 2-93
- service command 6-11, A-46
- services
  - accessing 2-62
  - disabling SNAT and NAT connections for 2-37
  - displaying numerically A-32
- services status, viewing 6-23
- services, monitoring 6-12
- Setting the MAC masquerade address 1-2, A-18
- Setup utility
  - purpose of 1-1
- shutdowns
  - Link Controller 6-17
- simple persistence
  - defined 2-22
  - example 2-23
- smtp monitor 2-85
- SNAT 2-65
- SNAT attributes
  - listed 2-65
- snat automap attribute
  - enabling 2-71

- SNAT automapping
    - and external VLAN 1-18
    - configuring 2-69
    - defined 2-70
    - uses of 2-69, 2-70
  - snat command 2-65, 2-74, 6-11, A-47
  - SNAT connections
    - mirroring 8-6
  - SNAT mirroring 2-73
  - SNAT settings, printing 6-15
  - SNAT statistics
    - clearing 2-74
    - resetting 6-11
    - viewing 6-21
  - SNATs
    - about 2-65
    - adding manually 2-68
    - address mappings 2-67
    - and active-active configurations 2-69
    - and outbound traffic 2-69
    - and self IP addresses 2-70
    - and VLANs 2-70
    - automapping 2-69
    - automapping defaults 2-69
    - configuring the default 2-67
    - connection limits 2-66
    - defining 2-65
    - disabling 2-36
    - global properties 2-66
    - TCP idle connection timeout 2-66
    - UDP idle connection timeout 2-66
    - uses for 2-65
  - SNMP
    - /etc/hosts.allow file 9-6
    - binding snmpd 9-9
    - client access 9-4, 9-7
    - configuring 9-4
    - configuring for dynamic ratio load balancing 2-17
    - in the Configuration utility 9-3
    - OIDs 9-8
    - syslog 9-8
    - trap configuration 9-7
  - SNMP template types
    - described 2-17
  - snmp_dca monitor 2-17
    - example 2-18
  - snmp_dca template
    - shown 2-86
  - snmp_dca_base monitor 2-17
    - example 2-18
  - snmp_dca_base template
    - shown 2-86
  - source IP addresses 2-75
  - spanning tree protocol (STP) A-49
  - SQL Enterprise Manager 2-93
  - sql monitor 2-87
  - SQL-based service checks
    - troubleshooting 2-93
  - SQL-based services
    - and service checks 2-87
  - SSH A-12
  - SSH client
    - downloading from the web server 7-3
    - installing on UNIX 7-4
    - remote administration Intro-1
  - SSH client. Intro-1
  - SSL proxy
    - for rewriting redirections 2-32
  - standby mode 8-10
  - standby units 8-2
  - static routes 7-8
  - statistical displays, customizing 6-23
  - statistics
    - monitoring 6-1
    - resetting 6-8
    - resetting global 6-12
    - resetting in Configuration utility 6-12
    - using Internet Link Evaluator 4-1
    - viewing 6-20
  - sticky entries A-14
  - sticky persistence 2-25
  - stp command A-49
  - stpd
    - restarting 1-22
  - strings
    - to specify redirection 2-29
  - summary 6-2
  - summary command 6-2, A-50
  - synchronization 8-2
  - synchronization. See controller synchronization
  - syslog file entries, generating 6-9
  - Syslog utility 6-1, 6-16, 9-8
  - system control variables
    - bigip.persist_map_proxies A-13
    - setting 6-9
  - system log files, viewing 6-21
  - system statistics, monitoring 6-1
- ## T
- tagged interfaces
    - defined 1-7
  - tags
    - embedding in packet headers 1-8
  - tcp monitor 2-82
  - tcp_echo monitor 2-82
  - technical support Intro-4
  - templates
    - selecting 2-19
  - terminal. See serial terminal
  - test accounts
    - creating 2-93

- threshold variable 2-19
  - timeout
    - for simple persistence 2-23, 2-24
  - ToS pool attribute
    - example 2-36
  - traffic
    - accepting inbound 2-65
    - distributing by priority 2-21
    - redirecting 2-28
    - restricting through tagged interfaces 1-7
    - restricting through untagged interfaces 1-7
  - translation
    - disabling 2-36
  - translation IP addresses
    - and SNATs 2-70
    - choosing 2-69
    - mapping nodes to 2-67
    - specifying 2-67
  - translation properties 2-51
  - transparent mode 2-99
  - transparent network devices 2-41
  - transparent nodes 2-42
  - trunk command 1-19, A-51
- U**
- UC Davis agent 2-17, 2-18, 2-20, 2-86
  - unit command A-52
  - units
    - active and standby 8-4
  - untagged interfaces
    - defined 1-7
  - URI path
    - redirecting 2-28
  - usage statistics A-50
  - user accounts
    - creating 6-22
    - deleting 6-22
  - user IDs, adding 6-22
  - user-defined metrics
    - gathering 2-17, 2-19
  - users, creating new 6-22
  - utilities
    - bigpipe commands A-1
    - types of Intro-1
- V**
- verbose command A-53
  - verbose keyword 6-15
  - virtual address
    - viewing statistics 6-1
  - virtual address statistics
    - resetting 6-9
    - viewing 6-20
  - virtual addresses
    - defining a netmask 2-51
    - displaying information 2-49
    - enabling and disabling 6-19
    - monitoring 6-12
    - removing from service 6-18, 6-19
    - statistics 2-49
    - translation properties 2-51
  - virtual command 2-40, 6-10, A-56
  - virtual port statistics
    - from bigpipe utility 6-1
    - resetting 6-9, 6-11
    - viewing 6-13, 6-20
  - virtual ports
    - allowing 6-19
    - enabling and disabling 6-19
    - removing from service 6-18, 6-19
  - virtual server configuration
    - referencing rules 2-56
    - See also rules
  - virtual server mappings 2-26, 2-27
    - defining standard 2-40
    - defining wildcard 2-43
    - included nodes 6-20
  - virtual server statistics
    - monitoring 6-1
    - resetting 6-9
    - viewing 6-1, 6-13, 6-20
  - virtual server status
    - displaying 6-12
    - viewing 6-23
  - virtual server types 2-40
  - virtual servers
    - additional features 2-58
    - and host names 7-1
    - and persistence 2-22, 2-26
    - configuring 2-40
    - configuring hop pools 2-56
    - defining wildcard 2-41
    - displaying unit number A-52
    - enabling 2-49
    - enabling and disabling 6-19
    - forwarding 2-75
    - mirroring 2-55, 8-6
    - monitoring 6-12
    - removing from service 6-18, 6-19
    - viewing 6-20
  - VLAN access methods 1-7
  - vlan command A-59
  - VLAN groups 1-15
  - VLAN IDs 1-8
  - vlangroup command 1-15
  - VLANs
    - and SNATs 2-69
    - managing 1-6

W

web administration A-15

wide IPs

- about 5-11

- and pools 5-12

- and QOS coefficients 5-10

- configuring 5-12

wildcard servers

- assigning to VLANs 2-45

- creating multiple 2-45

Windows 2000 Server agent 2-17, 2-18, 2-86

Windows 2000 SNMP agent 2-20

Windows Terminal Server persistence

- See WTS persistence.

WMI

- configuring for dynamic ratio load balancing 2-13

workstation configuration 7-4

WTS persistence modes 2-26

