

BIG-IP™ Controller Reference Guide

version 3.2

Service and Support Information

Product Version

This manual applies to version 3.2 of the BIG-IP® Controller platform.

Obtaining Technical Support

Web	tech.f5.com
Phone	(206) 505-0888
Fax	(206) 505-0802
Email (support issues)	support@f5.com
Email (suggestions)	feedback@f5.com

Contacting F5 Networks

Web	www.f5.com
Toll-free phone	(888) 88BIG-IP
Corporate phone	(206) 505-0800
Fax	(206) 505-0801
Email	sales@f5.com
Mailing Address	200 1st Avenue West Suite 500 Seattle, Washington 98119

Legal Notices

Copyright

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.

Copyright 1997-2000, F5 Networks, Inc. All rights reserved.

Trademarks

F5, BIG/IP, and 3-DNS are registered trademarks of F5 Networks, Inc. SEE-IT and GLOBAL-SITE are trademarks of F5 Networks, Inc. Other product and company names are registered trademarks or trademarks of their respective holders.

Export Regulation Notice

The BIG-IP® Controller may include cryptographic software. Under the Export Administration Act, the United States government may consider it a criminal offense to export this BIG-IP® Controller 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.

Canadian Regulatory Compliance

This class A digital apparatus complies with Canadian ICES-003.

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.

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/gpl.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>.

F5 Networks Limited Warranty

This warranty will apply to any sale of goods or services or license of software (collectively, "Products") from F5 Networks, Inc. ("F5"). Any additional or different terms including terms in any purchase order or order confirmation will have no effect unless expressly agreed to in writing by F5. Any software provided to a Customer is subject to the terms of the End User License Agreement delivered with the Product.

Limited Warranty

Software. F5 warrants that for a period of 90 days from the date of shipment: (a) the media on which the software is furnished will be free of defects in materials and workmanship under normal use; and (b) the software substantially conforms to its published specifications. Except for the foregoing, the software is provided AS IS.

In no event does F5 warrant that the Software is error free, that the Product will operate with any software or hardware other than that provided by F5 or specified in the documentation, or that the Product will satisfy Purchaser's own specific requirements.

Hardware. F5 warrants that the hardware component of any Product will, for a period of one year from the date of shipment from F5, be free from defects in material and workmanship under normal use.

Remedy. Purchaser's exclusive remedy and the entire liability of F5 under this limited warranty and any other guarantee made by F5 is, at F5's option, to repair or replace any Product or component that fails during the warranty period at no cost to Purchaser. Products returned to F5 must be pre-authorized by F5 with a Return Material Authorization (RMA) number marked on the outside of the package, and sent prepaid, insured, and packaged appropriately for safe shipment. The repaired or replaced item will be shipped to Purchaser, at F5's expense, no later than 7 days after receipt by F5. Title to any returned Products or components will transfer to F5 upon receipt. F5 will replace defective media or documentation or, at its option, undertake reasonable efforts to modify the software to correct any substantial non-conformance with the specifications.

Restrictions. The foregoing limited warranties extend only to the original Purchaser, and do not apply if a Product (a) has been altered, except by F5, (b) has not been installed, operated, repaired, or maintained in accordance with F5's instructions, (c) has been subjected to abnormal physical or electrical stress, misuse, negligence, or accident, or (d) has been operated outside of the environmental specifications for the Product. F5's limited software warranty does not apply to software corrections or upgrades.

Support, Upgrades. F5 provides software telephone support services at no charge for 90 days following the installation of any Product: Monday through Friday, from 6 a.m. to 6 p.m. Pacific time, excluding F5's holidays. Such support will consist of responding to trouble calls as reasonably required to make the Product perform as described in the Specifications. For advisory help requests, which are calls of a more consultative nature than a standard trouble call, F5 will provide up to two hours of telephone service at no charge. Additional service for

advisory help requests may be purchased at F5 Networks' then-current standard service fee. During this initial 90 day period, Customer is entitled, at no charge, to updated versions of covered software such as bug fixes, and incremental enhancements as designated by minor revision increases. In addition, Customer will receive special pricing on upgraded versions of covered Products such as new clients, new modules, and major enhancements designated by major revision increases. Customer may purchase a Maintenance Agreement for enhanced maintenance and support services.

DISCLAIMER; LIMITATION OF REMEDY: EXCEPT FOR THE WARRANTIES SPECIFICALLY DESCRIBED HEREIN, F5 DOES NOT MAKE ANY GUARANTEE OR WARRANTY, EXPRESS OR IMPLIED, WITH RESPECT TO PRODUCTS, SPECIFICATIONS, SUPPORT, SERVICE, OR ANYTHING ELSE. F5 HAS NOT AUTHORIZED ANYONE TO MAKE ANY REPRESENTATION OR WARRANTY OTHER THAN AS PROVIDED ABOVE. F5 DISCLAIMS ANY AND ALL WARRANTIES AND GUARANTEES, EXPRESS, IMPLIED, OR OTHERWISE, ARISING WITH RESPECT TO THE PRODUCTS OR SERVICES DELIVERED HEREUNDER, INCLUDING BUT NOT LIMITED TO THE WARRANTY OF MERCHANTABILITY, THE WARRANTY OF FITNESS FOR A PARTICULAR PURPOSE, AND ANY WARRANTY OF NON-INFRINGEMENT OF THE INTELLECTUAL PROPERTY RIGHTS OF ANY THIRD PARTY. F5 WILL HAVE NO OBLIGATION OR LIABILITY, WHETHER ARISING IN CONTRACT (INCLUDING WARRANTY), TORT (INCLUDING ACTIVE, PASSIVE, OR IMPUTED NEGLIGENCE, STRICT LIABILITY, OR PRODUCT LIABILITY), OR OTHERWISE FOR ANY SPECIAL, INCIDENTAL, CONSEQUENTIAL, OR INDIRECT DAMAGES INCLUDING BUT NOT LIMITED TO LOSS OF USE, LOSS OF DATA, BUSINESS INTERRUPTION, LOSS OF REVENUE, LOSS OF BUSINESS, OR OTHER FINANCIAL LOSS ARISING OUT OF OR IN CONNECTION WITH ANY OF THE PRODUCTS OR OTHER GOODS OR SERVICES FURNISHED TO CUSTOMER BY F5, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

End-user Software License

IMPORTANT! READ BEFORE INSTALLING OR OPERATING THIS PRODUCT.

CAREFULLY READ THE TERMS AND CONDITIONS OF THIS LICENSE BEFORE INSTALLING OR OPERATING THIS PRODUCT: BY INSTALLING, OPERATING, OR KEEPING THIS PRODUCT FOR MORE THAN THIRTY DAYS AFTER DELIVERY, YOU INDICATE YOUR ACCEPTANCE OF THESE TERMS AND CONDITIONS. IF YOU DO NOT AGREE WITH THE TERMS AND CONDITIONS OF THIS AGREEMENT, PROMPTLY CONTACT F5 NETWORKS, INC. ("F5") TO ARRANGE FOR RETURN OF THE PRODUCT FOR A REFUND.

1. Scope. This License applies to the software for the BIG-IP® Controller, whether such software is provided separately or as an integral part of a hardware product. As used herein, the term "Software" will refer to all such software, and the corrections, updates, new releases and new versions of such software. A product that consists of Software only will be referred to as a "Software Product" and a combination Software/Hardware product will be referred to as a "Combination Product." All Software is licensed, not sold, by F5. This License is a legal agreement between F5 and the single entity ("Licensee") that has acquired Software from F5 under applicable terms and conditions.
2. License Grant. Subject to the terms of this License, F5 grants to Licensee a non-exclusive, non-transferable license to use the Software in object code form solely on a single central processing unit owned or leased by Licensee. Other than as specifically described herein, no right or license is granted to Licensee to any of F5's trademarks, copyrights, or other intellectual property rights. Licensee may make one back-up copy of any Software Product, provided the back-up copy contains the same

copyright and proprietary information notices as the original Software Product. Licensee is not authorized to copy the Software contained in a Combination Product. The Software incorporates certain third party software which is used subject to licenses from the respective owners.

3. **Restrictions.** The Software, documentation, and the associated copyrights are owned by F5 or its licensors, and are protected by law and international treaties. Except as provided above, Licensee may not copy or reproduce the Software, and may not copy or translate the written materials without F5's prior, written consent. Licensee may not copy, modify, reverse compile, or reverse engineer the Software, or sell, sub-license, rent, or transfer the Software or any associated documentation to any third party.
4. **Export Control.** F5's standard Software incorporates cryptographic software. Licensee agrees to comply with the Export Administration Act, the Export Control Act, all regulations promulgated under such Acts, and all other laws and governmental regulations relating to the export of technical data, and equipment, and products produced therefrom, which are applicable to Licensee. In countries other than the US, Licensee agrees to comply with the local regulations regarding exporting or using cryptographic software.
5. **Limited Warranty.**
 - a. **Warranty.** F5 warrants that for a period of 90 days from the date of shipment: (i) the media on which the Software is furnished will be free of defects in materials and workmanship under normal use; and (ii) the Software substantially conforms to its published specifications. Except for the foregoing, the Software is provided AS IS. In no event does F5 warrant that the Software is error-free, that it will operate with any software or hardware other than that provided by F5 or specified in the documentation, or that the Software will satisfy Licensee's own specific requirements.
 - b. **Remedy.** Licensee's exclusive remedy and the entire liability of F5 under this limited warranty and any other guarantee made by F5 is, at F5's option, to repair or replace any F5 product that fails during the warranty period at no cost to Licensee. Any products returned to F5 must be pre-authorized by F5 with a Return Material Authorization (RMA) number marked on the outside of the package, and sent prepaid, insured, and packaged appropriately for safe shipment. The repaired or replaced item will be shipped to Licensee, at F5's expense, no later than 7 days after receipt by F5. Title to any returned product or components will transfer to F5 upon receipt. F5 will replace defective media or documentation or, at its option, undertake reasonable efforts to modify the Software to correct any substantial non-conformance with the specifications.
 - c. **Restrictions.** The foregoing limited warranties extend only to the original Licensee, and do not apply if a Software Product or Combination Product (i) has been altered, except by F5, (ii) has not been installed, operated, repaired, or maintained in accordance with F5's instructions, (iii) has been subjected to abnormal physical or electrical stress, misuse, negligence, or accident or (iv) has been operated outside of the environmental specifications for the product. F5's limited software warranty does not apply to software corrections or upgrades.
6. **Disclaimer: Limitation of Remedy.** EXCEPT FOR THE WARRANTIES SPECIFICALLY DESCRIBED HEREIN, F5 DOES NOT MAKE ANY GUARANTEE OR WARRANTY, EXPRESS OR IMPLIED, WITH RESPECT TO THE SOFTWARE, SPECIFICATIONS, SUPPORT, SERVICE OR ANYTHING ELSE. F5 HAS NOT AUTHORIZED ANYONE TO MAKE ANY REPRESENTATION OR WARRANTY OTHER THAN AS PROVIDED ABOVE. F5 DISCLAIMS ANY AND ALL WARRANTIES AND GUARANTEES, EXPRESS, IMPLIED OR OTHERWISE, ARISING WITH RESPECT TO THE SOFTWARE OR SERVICES DELIVERED HEREUNDER, INCLUDING BUT NOT LIMITED TO THE WARRANTY OF MERCHANTABILITY, THE

WARRANTY OF FITNESS FOR A PARTICULAR PURPOSE, AND ANY WARRANTY OF NON-INFRINGEMENT OF THE INTELLECTUAL PROPERTY RIGHTS OF ANY THIRD PARTY. F5 WILL HAVE NO OBLIGATION OR LIABILITY, WHETHER ARISING IN CONTRACT (INCLUDING WARRANTY), TORT (INCLUDING ACTIVE, PASSIVE, OR IMPUTED NEGLIGENCE, STRICT LIABILITY OR PRODUCT LIABILITY), OR OTHERWISE, FOR ANY SPECIAL, INCIDENTAL, CONSEQUENTIAL, OR INDIRECT DAMAGES INCLUDING BUT NOT LIMITED TO LOSS OF USE, LOSS OF DATA, BUSINESS INTERRUPTION, LOSS OF REVENUE, LOSS OF BUSINESS, OR OTHER FINANCIAL LOSS ARISING OUT OF OR IN CONNECTION WITH THE SOFTWARE OR OTHER GOODS OR SERVICES FURNISHED TO LICENSEE BY F5, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

7. Termination. This License is effective until terminated, and will automatically terminate if Licensee fails to comply with any of its provisions. Upon termination of this License, the Licensee will destroy the Software and documentation and all copies or portions thereof.
8. Miscellaneous. This Agreement will be governed by the laws of the State of Washington, USA without regard to its choice of law rules. The provisions of the U.N. Convention for the International Sale of Goods will not apply. Any provisions found to be unenforceable will not affect the enforceability of the other provisions contained herein, but will instead be replaced with a provision as similar in meaning to the original as possible. This Agreement constitutes the entire agreement between the parties with regard to its subject matter. No modification will be binding unless in writing and signed by the parties.



Table of Contents

Chapter 1

Introduction to the BIG-IP Controller Reference Guide

Welcome to the BIG-IP Controller Reference Guide	1-1
--	-----

Chapter 2

BIG/pipe Command Reference

BIG/pipe commands	2-1
-?	2-4
alias	2-5
configsync	2-7
conn	2-8
-d	2-9
-f	2-10
failover	2-11
gateway	2-12
-h and -help	2-13
interface	2-14
ipalias	2-20
-l	2-21
lb	2-22
maint	2-23
mirror	2-24
nat	2-25
-n	2-28
node	2-29
persist	2-33
pool	2-34
port	2-42
proxy	2-44
-r	2-47
ratio	2-48
rule	2-50
-s	2-54
snat	2-55
summary	2-59
timeout_node	2-62

timeout_svc	2-64
tping_node	2-66
tping_svc	2-67
treaper	2-69
udp	2-71
unit	2-73
-v	2-74
version	2-75
vip	2-76
Backward-compatible commands	2-84

Chapter 3

Configuration Files

Configuration files for the BIG-IP Controller	3-1
---	-----

Chapter 4

BIG-IP Controller Configuration Utilities

Introducing the BIG-IP Controller configuration utilities	4-1
config	4-2
config_failover	4-3
reconfig-httpd	4-4
config_telnetd	4-5
config_sshd	4-6
config_rshd	4-7
config_synckey	4-8

Chapter 5

BIG-IP System Control Variables

Setting BIG-IP system control variables	5-1
sysctl	5-2
bigip.bonfire_mode	5-3
bigip.bonfire_compatibility_mode	5-4
bigip.fastest_max_idle_time	5-5
bigip.forwarding_vip_overrides_default_snat	5-6
bigip.halt_reboot_timeout	5-7

bigip.improved_fastest	5-8
bigip.max_sticky_entries	5-9
bigip.memory_reboot_percent	5-10
bigip.open_3dns_lockdown_ports	5-11
bigip.open_telnet_port	5-12
bigip.open_ftp_ports	5-13
bigip.open_ssh_port	5-14
bigip.open_rsh_ports	5-15
bigip.persist_map_proxies	5-16
bigip.persist_time_used_as_limit	5-17
bigip.persist_on_any_vip	5-18
bigip.persist_on_any_port_same_vip	5-19
bigip.tcphps_mss_override	5-20
bigip.verbose_log_level	5-21
bigip.vipnoarp	5-22
bigip.webadmin_port	5-23
net.inet.ip.forwarding	5-24
net.inet.ip.sourcecheck	5-25

Chapter 6

System Utilities

sod	6-1
bigd	6-3
big3d	6-8

Chapter 7

BIG/db Configuration Keys

Supported BIG/db configuration keys	7-1
---	-----

Index

Table of Contents



I

Introduction to the BIG-IP Controller Reference Guide

- Welcome to the BIG-IP Controller Reference Guide

Welcome to the BIG-IP Controller Reference Guide

Welcome to the *BIG-IP® Controller Reference Guide*. This guide contains reference information for the BIG-IP Controller BIG/pipe command line utility, configuration files, configuration utilities, system control variables, and system utilities. This book is part of a series of three guides:

❖ ***BIG-IP Controller Getting Started Guide***

Use this guide for hardware configuration and basic software configuration.

❖ ***BIG-IP Controller Administrator Guide***

Use this guide for advanced software configuration and administration of the BIG-IP Controller.

❖ ***BIG-IP Controller Reference Guide***

Use this guide for reference information including the BIG/pipe command line commands, BIG-IP Controller configuration utilities, and other system utilities.

2

BIG/pipe Command Reference

- BIG/pipe commands

BIG/pipe commands

This chapter lists the various BIG/pipe commands with descriptions. Table 2.1 explains the conventions used in the command line syntax described in this chapter.

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 parts of a command.
[]	Syntax inside the square brackets is optional.
...	Indicates that you can type a series of items.

Table 2.1 Command line conventions used in this manual

Some entries contain additional information about using the command. At the end of the chapter is a list of commands from previous versions of the BIG/pipe utility.

Command	Description	Page
-?	Displays online help for an individual bigpipe command.	2-4
alias	Defines an IP alias to be pinged on behalf of a specific group of nodes.	2-5
configsync	Synchronizes the /etc/bigip.conf between the two BIG-IP Controller units in a redundant system.	2-7
conn	Shows information about current connections such as the source IP address, virtual server and port, and node.	2-8
-d	Verifies command syntax for the specified command without executing a command.	2-9
-f	Resets the BIG-IP Controller and loads a specified configuration file.	2-10

Command	Description	Page
failover	Sets the BIG-IP Controller as active or standby.	2-11
gateway	Turns the gateway fail-safe feature on and off.	2-12
-h and -help	Displays online help for BIG/pipe command syntax.	2-13
interface	Sets options on individual interfaces.	2-14
ipalias	Configure shared addresses on interfaces.	2-20
-l	Loads the BIG-IP Controller configuration without resetting the current configuration.	2-21
lb	Sets the load balancing mode.	2-22
maint	Toggles the BIG-IP Controller into and out of maintenance mode.	2-23
mirror	Sets mirroring of the active BIG-IP Controller to the standby controller.	2-24
-n	Displays ports numerically rather than by service name	2-28
nat	Defines external network address translations for nodes.	2-25
node	Defines node property settings.	2-29
persist	Defines and displays persistence settings for simple TCP and UDP persistence.	2-33
pool	Defines load balancing pools.	2-34
port	Defines properties for virtual ports.	2-42
proxy	Defines the properties of the SSL gateway for the SSL Accelerator.	2-44
-r	Clears the BIG-IP Controller configuration and counter values.	2-47
ratio	Sets load-balancing weights and priority levels used in the Ratio and Priority load balancing modes.	2-48
rule	Defines load balancing rules.	2-50
-s	Writes the current configuration to a file.	2-54
snat	Defines and sets options for SNAT (Secure NAT).	2-55
summary	Displays summary statistics for the BIG-IP Controller.	2-59
timeout_node	Sets the amount of time node addresses have to respond to a ping issued by the BIG-IP Controller.	2-62
timeout_svc	Sets the amount of time services have to respond to a service check issued by the BIG-IP Controller.	2-64
tping_node	Sets the interval at which the BIG-IP Controller pings node addresses to determine node status.	2-66
tping_svc	Sets the interval at which the BIG-IP Controller issues service checks to nodes to determine node status.	2-67

Command	Description	Page
treaper	Sets the timeout for idle TCP connections on ports.	2-69
udp	Enables UDP traffic on ports, and sets the timeout for idle UDP connections.	2-71
unit	Displays the unit number assigned to a particular BIG-IP Controller.	2-73
-v	Displays the BIG/pipe utility version number.	2-74
version	Displays the BIG-IP Controller software version number.	2-75
vip	Defines virtual servers, virtual server mappings, and virtual server properties.	2-76
Backward-compatible commands	Lists the commands from previous versions of the BIG-IP Controller that are compatible with this version.	2-84

-?

bigpipe <command> -?

Description

For certain commands, displays online help, including complete syntax, description, and other related information. For example, to see online help for the **bigpipe port** command, type:

bigpipe port -?

alias

```
bigpipe alias [<node ip> [...<node ip>] ] show
```

```
bigpipe alias <node ip> [...<node ip>] delete
```

```
bigpipe alias <node ip> [...<node ip>] pingnode <pingnode ip>
```

Description

Defines a single node address to represent a group of node addresses which are actually IP aliases on the same physical server. To determine if the nodes associated with the representative node alias are available, the BIG-IP Controller sends a single node ping to the node alias, rather than an individual ping to each node address.

Note that you may also find this feature useful for nodes that are configured for service check, as long as each node uses the same port number.

Defining a node alias

Use the following syntax to define the node alias for one or more node addresses, where **<pingnode ip>** is the node alias (the node address that represents the group):

```
bigpipe alias <node ip> [...<node ip>] pingnode <pingnode ip>
```

◆ Note

The address that serves as the node alias (<pingnode ip>) must be a node address that is already defined in one or more virtual server mappings.

The following command defines a node alias for two node addresses, 192.168.42.2 and 192.168.42.3. The BIG-IP Controller performs node pings on the alias address 192.168.42.1 to determine the availability of 192.168.42.2 and 192.168.42.3.

```
bigpipe alias 192.168.42.2 192.168.42.3 pingnode 192.168.42.1
```

Deleting a node alias

The following command deletes the node alias defined for the specific node:

```
bigpipe alias <node ip> delete
```

Displaying current node aliases

The following command displays all node aliases defined on the BIG-IP Controller:

```
bigpipe alias show
```

The following command displays the node alias defined for a specific node:

```
bigpipe alias <node ip> show
```

configsync

bigpipe configsync [all]

Description

Synchronizes configurations of two BIG-IP Controllers in a redundant system by copying the configuration file(s) from the active system to the standby system.

Using the **configsync** command without the **all** option synchronizes only the boot configuration file **/etc/bigip.conf**.

The **all** option changes the set of configuration files modified when the command is executed. When you synchronize a configuration using **configsync all** command, the following configuration files are copied to the other BIG-IP Controller:

❖ The common BIG/db keys

- ❖ **/etc/bigip.conf**
- ❖ **/etc/bigd.conf**
- ❖ **/etc/hosts.allow**
- ❖ **/etc/hosts.deny**
- ❖ **/etc/ipfw.conf**
- ❖ **/etc/rateclass.conf**
- ❖ **/etc/ipfwrate.conf**
- ❖ **/etc/snmpd.conf**
- ❖ **rc.sysctl**

Be sure to save the current configuration to the **/etc/bigip.conf** file before you use the configuration synchronization feature.

◆ WARNING

If you are synchronizing a standby controller that already has configuration information defined, we recommend that you back up that controller's original configuration file(s).

conn

```
bigpipe conn [ <virt ip>[:<port>] ] dump [mirror]
```

Description

Displays information about current client connections to virtual addresses and virtual servers.

The following command displays all current client connections:

```
bigpipe conn dump
```

The output shows the source IP, virtual server and port, and node connected to.

```
bigip conn dump

from                vip                node
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 2.1 Formatted output of the **conn** command*

This command can also show connections that are active on the given controller as well as those that are standby connections for the peer BIG-IP Controller. By default, the **dump** command only shows items that are active on the given unit. To see standby items, you must use the **mirror** qualifier.

```
bigpipe conn dump mirror
```

-d

bigpipe -d [-]

bigpipe -d -f <filename>

Description

Parses the command line and checks syntax without executing the specified command.

This distinguishes between valid and invalid commands, and is particularly useful with the **-f** option, to validate the configuration file.

Use the **-d** command followed by a command that you want to validate:

```
bigpipe -d vip 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 **-d** command together with the **-f <filename>** command to validate the specified configuration file. For example, to check the syntax of the configuration file **/etc/altbigpipe.conf**, use the following command:

```
bigpipe -d -f /etc/altbigip.conf
```

-f

bigpipe -f <filename>

Description

Resets all of the BIG-IP Controller settings and then loads the configuration settings from the specified file, typically the **/etc/bigip.conf** file, or another file you specify.

bigpipe -f /etc/bigip.conf

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 **-f** command with the **<filename>** parameter. For example, if you renamed your configuration file to **/etc/bigtest.conf**, the test command would be:

bigpipe -f /etc/bigtest.conf

failover

bigpipe failover standby | show | init | failback

Description

This group of commands affects the fail-over status of the BIG-IP Controller.

In an active/standby or active-active configuration, run the following command to place a controller in standby mode:

bigpipe failover standby

Show the status of the controller with the following command:

bigpipe failover show

The **failback** command is only applicable if you are running a redundant system in active-active mode.

In an active-active configuration, run the following command after you issue the **bigpipe failover standby** command. This allows the inactive controller to resume handling connections:

bigpipe failover failback

You can use the **bigpipe failover init** command to refresh the parameters of the fail-over daemon (**/sbin/sod**) with any new configuration data entered in the BIG/db database.

bigpipe failover init

gateway

bigpipe gateway failsafe arm | disarm | show

Description

Turns the gateway fail-safe feature on and off. This command is supported only for redundant systems. To configure gateway pingers, you must first set the IP address of the router, ping interval, and timeout period in BIG/db. For information about configuring gateway fail-safe, see the ***BIG-IP Controller Administrator Guide**, Working with Advanced Redundant System Features*.

The typical use of gateway fail-safe is where active and standby BIG-IP Controllers use different routers as gateways to the internet. Fail-over is triggered if the gateway for the active controller is unreachable. Note that this is not a condition that is reliably detected by the interface fail-safe feature, but is reliably detected by gateway fail-safe.

To arm fail-safe on the gateway:

bigpipe gateway failsafe arm

To disarm fail-safe on the gateway, enter the following command:

bigpipe gateway failsafe disarm

To see the current fail-safe status for the gateway, enter the following command:

bigpipe gateway failsafe show

-h and -help

bigpipe [-h | -help]

Description

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

```

bigpipe interface <ifname> show
bigpipe interface <ifname> source enable | disable
bigpipe interface <ifname> dest enable | disable
bigpipe interface <ifname> adminport open | lockdown
bigpipe interface <ifname> failsafe arm | disarm | show
bigpipe interface <ifname> timeout <seconds> | show
bigpipe interface <ifname> mac_masq <mac_addr> | show
bigpipe interface <ifname> vlans enable | disable | show

```

Description

Displays names of installed network interface cards and allows you to set properties for each network interface card.

◆ Note

Interface fail-safe is not designed for gateway or node failure detection, as it cannot detect router or node failures in instances where other sources of Ethernet traffic are active on the interface.

Designating an internal or external interface

With BIG-IP Controller version 3.0, you can define interfaces using three new parameters: **source**, **dest**, and **adminport**. You can mix and match these options to streamline the performance of the BIG-IP Controllers in the network. The attributes that determine the way an interface handles connections are described in Table 2.2.

Interface type	Attributes
Internal	Process source addresses Administrative ports open
External	Process destination addresses Administrative ports locked down

Table 2.2 *Attributes of internal and external interfaces*

Use the following syntax to designate an interface as an internal or external interface.

```
bigpipe interface <ifname> source enable | disable
bigpipe interface <ifname> dest enable | disable
bigpipe interface <ifname> adminport open | lockdown
```

The **<ifname>** parameter takes a valid interface name such as:

- ❖ **exp0**
This is the first Intel NIC
- ❖ **fpa1**
This is the second FDDI NIC
- ❖ **de2**
This is the third DEC/SMC NIC
- ❖ **sk0**
This is the first SysKonnnect Gigabit Ethernet NIC

◆ **Note**

Dual port Ethernet NICs show up as two distinct interfaces

The following example syntax configures the interface **exp0** as an internal interface on the BIG-IP Controller:

```
bigpipe interface exp0 source enable
bigpipe interface exp0 dest disable
bigpipe interface exp0 adminport open
```

The following example syntax configures the interface **exp1** as an external interface on the BIG-IP Controller:

```
bigpipe interface exp1 source disable
bigpipe interface exp1 dest enable
bigpipe interface exp1 adminport lockdown
```

◆ WARNING

Use caution when redefining interfaces. When you reconfigure interfaces, make sure that you have set up the interfaces you need for operation. It is possible to accidentally take the controller out of network service by redefining interfaces.

Displaying status for interfaces

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

```
bigpipe interface show
```

Figure 2.2 is an example of the output you see when you issue this command on an active/standby controller in active mode.

```
exp0      11.11.11.2, dest enable, source disable, disarmed, timeout 30
  shared alias 11.11.11.3 netmask 255.0.0.0 broadcast 11.255.255.255 unit 1
exp1      11.12.11.2, dest disable, source enable, disarmed, timeout 30
  shared alias 11.12.11.3 netmask 255.0.0.0 broadcast 11.255.255.255 unit 1
```

Figure 2.2 The *bigpipe interface show* command output

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

```
bigpipe interface <ifname> show
```

Arming and disarming the fail-safe mode

Use the following command to activate the BIG-IP Controller interface fail-safe mode.

```
bigpipe interface <ifname> failsafe arm
```

When armed, the active controller automatically fails over to the standby controller whenever the active controller detects that there is no activity on the specified interface, and subsequently detects no activity on the interface in response to ARP requests. The default fail-safe mode is set to **disarm**.

WARNING

You should arm the fail-safe mode only after you configure the BIG-IP Controller, and both the active and standby units are ready to be placed into a production environment.

Note that you must specify a default route before using the **bigpipe interface failsafe** command. You specify the default route in the **/etc/hosts** and **/etc/netstart** files.

Use the following command to deactivate the BIG-IP Controller interface fail-safe mode.

```
bigpipe interface <ifname> failsafe disarm
```

Setting the fail-safe timeout

Use the following syntax to set the amount of time, in seconds, that an interface will be monitored for activity in response to a BIG-IP Controller ARP request, in order to be designated operational.

```
bigpipe interface <ifname> timeout <seconds>
```

If no activity is detected on the interface within the specified time, the BIG-IP Controller assumes that the interface is down. Note that the default setting is 30 seconds.

Warning messages and ARP requests are generated after half of the specified time-out period. In the case of an armed BIG-IP Controller in a BIG-IP redundant system, traffic is switched from the active unit to the standby unit at the end of the time-out period. Note that the fail-safe timeout is used only if the fail-safe option is armed on the interface.

Viewing the timeout setting

Use the following syntax to view the fail-over timeout setting for a specific interface:

```
bigpipe interface <ifname> timeout show
```

Displaying the current fail-safe status

Use the following syntax to display the current status and settings for the BIG-IP Controller fail-safe mode:

```
bigpipe interface failsafe show
```

Setting the MAC masquerade address

Sharing the MAC masquerade address makes it possible to use BIG-IP Controllers in a network topology using secure hubs. You can view the media access control (MAC) address on a given controller using the following command:

```
/sbin/ifconfig -a
```

Use the following syntax to set the MAC masquerade address that will be shared by both BIG-IP Controllers in the redundant system.

```
bigpipe interface <ifname> mac_masq <MAC addr>
```

WARNING

*You must specify a default route before using the **mac_masq** command. You specify the default route in the **/etc/hosts** and **/etc/netstart** files.*

Find the MAC address on both the active and standby units and choose one that is similar but unique. A safe technique for choosing the shared MAC address follows:

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

```
Active: exp0 = 0:0:0:ac:4c:a2
```

```
Standby: exp0 = 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 BIG-IP Controllers in the redundant system.

The shared MAC address is used only when the BIG-IP 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 BIG-IP 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

You can use the same technique to configure a shared MAC address for each interface.

Enabling VLAN communication for an interface

To use IEEE 802.1q VLAN Trunk mode, you must first set up VLAN tags in **/etc/netstart** and the shared IP in BIG/db. For detailed information about setting up VLAN tags, see the ***BIG-IP Controller Administrator Guide, Using Advanced Network Configurations***.

Use the following syntax to enable, disable, or show the VLAN status of the specified internal interface:

```
bigpipe interface <ifname> vlans enable | disable | show
```

ipalias

```
ipalias <ifname> <if address> netmask <ip mask> [ broadcast <ip  
address> ] [ unit <id> ] [ tag <vlan tag> ]
```

Description

Configure shared IP addresses on installed network interface cards. The configuration you create with this command is stored in the BIG/db. If you use VLAN tags in your configuration, you can use this command to set the VLAN tag for the shared IP alias.

You must issue this command for each interface that you want configure with the same IP alias. For example, if you want to configure the IP alias 192.168.100.100 for the interfaces **exp0** and **exp1**, type the following commands:

```
bigpipe ipalias exp0 192.168.100.100 netmask 255.255.0.0  
bigpipe ipalias exp1 192.168.100.100 netmask 255.255.0.0
```

-l

bigpipe -l <file_name>

Description

Use the **-l** command to load the BIG-IP Controller configuration from **<file_name>** without resetting the current configuration.

lb

```
bigpipe lb show
bigpipe lb round_robin | rr
bigpipe lb ratio
bigpipe lb priority
bigpipe lb fastest
bigpipe lb least_conn
bigpipe lb predictive
bigpipe lb observed
```

Description

Sets the global load balancing mode for all node list virtual servers.

◆ Note

Pools are configured with their own load balancing method.

Setting the load balancing mode

Use the following syntax to set the load balancing mode:

```
bigpipe lb <mode name>
```

The mode names allowed are displayed in the syntax section above.

The command below sets the load balancing mode to Least Connections, which routes new connections to the node which currently maintains the least number of connections.

```
bigpipe lb least_conn
```

Viewing the currently selected load balancing mode

The following command displays the currently selected load balancing mode.

```
bigpipe lb show
```

`maint`

`bigpipe maint`

Description

Toggles a BIG-IP Controller into and out of Maintenance mode. When in Maintenance mode, a BIG-IP 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.

bigpipe maint

If the BIG-IP Controller is already in maintenance mode, the **maint** command takes the BIG-IP Controller out of maintenance mode. If the BIG-IP Controller is in maintenance mode for more than 20 minutes, the BIG-IP Controller immediately begins to accept new connection requests.

If the BIG-IP 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 the process up by reloading the configuration file, using the following command:

bigpipe -f /etc/bigip.conf

mirror

bigpipe mirror enable | disable | show

Description

Enables and disables mirroring between active and standby BIG-IP Controllers. Mirroring ensures that persistence and connection information on the active controller is duplicated on the standby controllers. This command enables and disables mirroring for all virtual servers.

To enable mirroring on a redundant system:

bigpipe mirror enable

To disable mirroring on a redundant system:

bigpipe mirror disable

To show the current status of mirroring on a redundant system:

bigpipe mirror show

nat

```
bigpipe nat <orig_addr> to <trans_addr>[/<bitmask>] [<ifname>]
    [unit <unit ID>]
bigpipe nat <orig_addr> to <trans_addr> netmask <netmask> \
    [broadcast <broadcast_ip>] [<ifname>] [unit <unit ID>]
bigpipe nat <orig_addr> [...<orig_addr>] delete
bigpipe nat <trans_addr> [...<trans_addr>] delete
bigpipe nat [<trans_addr> [...<trans_addr>] ] show
bigpipe nat [<orig_addr> [...<orig_addr>] ] show
bigpipe nat [<orig_addr>] stats reset
```

Description

Defines 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 BIG-IP Controller **<orig_addr>** and an unused routable address on the network in front of the BIG-IP Controller **<trans_addr>**.

Defining a NAT

A NAT definition maps the IP address of a node **<orig_addr>** to a routable address on the external interface **<trans_addr>**, and can include an optional interface and netmask specification. Use the following syntax to define a NAT:

```
bigpipe nat <orig_addr> to <trans_addr>[/<bitmask>] [<ifname>]
    [unit <unit ID>]
```

The **<ifname>** parameter is the internal interface of the BIG-IP Controller through which packets must pass to get to the destination internal address. The BIG-IP Controller can determine the interface to configure for the NAT in most cases. The **<ifname>** parameter is useful, for example, where there is more than one

internal interface. You can use the **unit** <unit ID> parameter to specify the controller to which this NAT applies in an active-active redundant system.

The following example shows a NAT definition:

```
bigpipe nat 10.10.10.10 to 10.12.10.10/24 expl
```

Deleting NATs

Use the following syntax to delete one or more NATs from the system:

```
bigpipe nat <orig_addr> [...<orig_addr>] delete
```

Displaying status of NATs

Use the following command to display the status of all NATs included in the configuration:

```
bigpipe nat show
```

Use the following syntax to display the status of one or more selected NATs (see the following figure, 2.3):

```
bigpipe nat <orig_addr> [...<orig_addr>] show
```

```
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 2.3 Output when you display the status of a NAT.

Resetting statistics for a NAT

Use the following command to reset the statistics for an individual NAT:

```
bigpipe nat [<orig_addr>] stats reset
```

Use the following command to reset the statistics for all NATs:


```
bigpipe nat stats reset
```

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 BIG-IP Controller.
- ❖ You must delete a NAT before you can redefine it.
- ❖ The interface for a NAT may only be configured when the NAT is first defined.

`-n`

`bigpipe -n`

Description

Use the `-n` option in combination with other commands, such as **bigpipe vip**, to display ports numerically rather than by service name. For example, type the following command to display ports numerically:

`bigpipe -n vip`

Notice the ports are listed numerically rather than by service name. See Figure 2.4.

```
VIP +-----> 11.100.1.1          UNIT 1
|              (cur, max, limit, tot) = (0, 0, 0, 0)
|              (pckts,bits) in = (0, 0), out = (0, 0)
+---+---> PORT 80                UP
|              (cur, max, limit, tot) = (0, 0, 0, 0)
|              (pckts,bits) in = (0, 0), out = (0, 0)
MEMBER 11.12.1.100:80            UP
              (cur, max, limit, tot) = (0, 0, 0, 0)
              (pckts,bits) in = (0, 0), out = (0, 0)
```

Figure 2.4 The output of `bigpipe -n vip`

node

```
bigpipe node <node ip>[:<port>][...<node ip>[:<port>]] \
    enable | disable
bigpipe node [<node ip>[:<port>][...<node ip>[:<port>]] ] show
bigpipe node <node ip>[:<port>][...<node ip>[:<port>]] \
    limit <max conn>
bigpipe node <node ip>[:<port>] up | down
bigpipe node [<node ip>:<port>] stats reset
```

Description

Displays information about nodes and allows you to set properties for nodes, and node addresses.

Enabling and disabling nodes and node addresses

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

```
bigpipe node 192.168.21.1 enable
```

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

```
bigpipe node 192.168.21.1 disable
```

To enable one or more node addresses, use the **node** command with a node address and port, and the **enable** option:

```
bigpipe node 192.168.21.1:80 enable
```

To disable one or more node addresses, use the **node** command with **disable** option:

```
bigpipe node 192.168.21.1:80 disable
```

Marking nodes and node ports up or down

To mark a node address down, use 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):

```
bigpipe node 192.168.21.1 down
```

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

```
bigpipe node 192.168.21.1 up
```

To mark a particular port down, use 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):

```
bigpipe node 192.168.21.1:80 down
```

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

```
bigpipe node 192.168.21.1:80 up
```

Setting connection limits for nodes

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

```
bigpipe node <node ip>[:<port>][...<node ip>[:<port>]] \
  limit <max conn>
```

Note that to remove a connection limit, you also issue the preceding command, but set the **<max conn>** variable to **0** (zero). For example:

```
bigpipe node 192.168.21.1:80 limit 0
```

Setting connection limits for node addresses

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

```
bigpipe node 192.168.21.1 192.168.21.1
          192.168.21.1 limit 100
```

To remove a connection limit, you also issue this command, but set the **<max conn>** variable to **0** (zero).

Displaying status of all nodes

bigpipe node show

When you issue the **node show** command, the BIG-IP 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. 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 2.5 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)
|   (pckts,bits) in = (0, 0), out = (0, 0)
+-  PORT 20              UP
    (cur, max, limit, tot) = (0, 0, 0, 0)
    (pckts,bits) in = (0, 0), out = (0, 0)
```

Figure 2.5 Node status and statistics

Displaying the status of individual nodes and node addresses

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

```
bigpipe 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:

```
bigpipe node 192.168.21.1:80 show
```

Resetting statistics for a node

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

```
bigpipe node [<node ip>:<port>] stats reset
```

`persist`

```
bigpipe persist <port> [...<port>] <seconds>
```

```
bigpipe persist dump
```

Description

Enables or disables simple persistence on one or more virtual ports. Persistence tracks the source IP addresses of all incoming requests, and the nodes and ports that hosted the request. It forces new connections from the source address to use the same node as used by the prior connection from that source IP address. A configurable time limit determines how long the BIG-IP Controller retains persistent connection information. By default, persistence is disabled on all ports. Persistence is affected by certain system control variables.

Setting a persistence timeout

Use the following syntax to set the number of seconds for which the BIG-IP Controller maintains persistent connection information on a specific virtual port:

```
bigpipe persist <port> <seconds>
```

Set <seconds> to **0** to turn persistence **off** for a specific virtual port.

Displaying persistent connections

Use the following syntax to display information about current persistent connections:

```
bigpipe persist [<port>] [...port] dump
```

pool

```
bigpipe pool <pool name> { lb_mode <lb_mode_specification>
    [persist_mode <persist_mode_specification>] <member
    definition>... }
bigpipe pool <pool name> add { <member definition>... }
bigpipe pool <pool name> delete { <member definition>... }
bigpipe pool <pool name> modify { [lb_mode <lb_mode_specification>]
    [persist_mode <persist_mode_specification>] <member
    definition>... }
bigpipe pool <pool name> delete
bigpipe pool [<pool name>] show
bigpipe pool <pool name> lb_mode show
bigpipe pool <pool name> persist show
```

Description

Use the pool command to create, delete, modify, or display the pool definitions on the BIG-IP Controller. Use pools to group members together with a common load balancing mode and persistence mode. For additional information about configuring pools, see the ***BIG-IP Controller Administrator Guide, Working with Intelligent Traffic Control***.

Creating a pool

To create a pool use the following syntax:

```
bigpipe pool <pool_name> {lb_mode <lb_mode_specification>
    [persist_mode <persist_mode_specification>]
    <member_definition>... member <member_definition>}
```

Each of these elements is described in Table 2.4, on page 2-40.

◆ Note

*For detailed information about setting up persistence with pools, see the **BIG-IP Controller Administrator Guide, Working with Advanced Persistence Options**.*

To activate Insert HTTP cookie persistence from the command line

If you specify Insert mode, the information about the server to which the client connects is inserted in the header of the HTTP response from the server as a cookie. The cookie is named **BIGipServer <pool_name>**, and it includes the address and port of the server handling the connection. The expiration date for the cookie is set based on the timeout configured on the BIG-IP Controller.

To activate Insert mode from the command line, use the following syntax:

```
bigpipe pool <pool_name> { <lb_mode_specification> persist_mode
    cookie cookie_mode insert cookie_expiration <timeout> <member
    definition> }
```

The `<timeout>` value for the cookie is written using the following format:

<days>d hh:mm:ss

To activate Rewrite mode cookie persistence from the command line

If you specify Rewrite mode, the BIG-IP Controller intercepts a Set-Cookie, named **BIGipCookie**, sent from the server to the client and overwrites the name and value of the cookie. The new cookie is named **BIGipServer** `<pool_name>` and it includes the address and port of the server handling the connection.

Rewrite mode requires you to set up the cookie created by the server. In order for Rewrite mode to work, there needs to be a blank cookie coming from the web server for the BIG-IP Controller to rewrite. With Apache variants, the cookie can be added to every web page header by adding an entry in the **httpd.conf** file:

Header add Set-Cookie

BIGipCookie=0000000000000000000000000000000000
00000000...

(The cookie should contain a total of 120 zeros.)

◆ WARNING

For backward compatibility the blank cookie can contain only 75 zeros. However, cookies of this size do not allow you to use rules and persistence together.

To activate Rewrite mode from the command line, use the following syntax:

```
bigpipe pool <pool_name> { <lb_mode_specification> persist_mode
  cookie cookie_mode rewrite cookie_expiration <timeout> <member
  definition> }
```

The <timeout> value for the cookie is written using the following format:

```
<days>d hh:mm:ss
```

To activate Passive mode cookie persistence from the command line

If you specify Passive mode, the BIG-IP Controller does not insert or search for blank Set-Cookies in the response from the server. It does not try to set up the cookie. In this mode, it is assumed that the server provides the cookie formatted with the correct node information and timeout.

In order for Passive mode to work, there needs to be a cookie coming from the web server with the appropriate node information in the cookie. With Apache variants, the cookie can be added to every web page header by adding an entry in the **httpd.conf** file:

```
Header add Set-Cookie: "BIGipServer my_pool=184658624.20480.000;
  expires=Sat, 19-Aug-2000 19:35:45 GMT; path=/"
```

In this example, **my_pool** is the name of the pool that contains the server node, **184658624** is the encoded node address and **20480** is the encoded port.

The equation for an address (a.b.c.d) is:

$$d*256^3 + c*256^2 + b*256 + a$$

The way to encode the port is to take the two bytes that store the port and reverse them. So, port 80 becomes $80 * 256 + 0 = 20480$. Port 1433 (instead of $5 * 256 + 153$) becomes $153 * 256 + 5 = 39173$.

After you set up the cookie created by the web server, you must activate Passive mode on the BIG-IP Controller. To activate HTTP cookie persistence from the command line, use the following syntax:

```
bigpipe pool <pool_name> { <lb_mode_specification> persist_mode
    cookie cookie_mode passive <member definition> }
```

◆ Note

The <timeout> value is not used in Passive mode.

To configure the hash cookie persistence option from the command line

If you specify hash mode, the hash mode consistently maps a cookie value to a specific node. When the client returns to the site, the BIG-IP Controller uses the cookie information to return the client to a given node. With this mode, the web server must generate the cookie. The BIG-IP Controller does not create the cookie automatically like it does with insert mode.

Use the following syntax to configure the hash cookie persistence option:

```
bigpipe pool <pool_name> { <lb_mode_specification> persist_mode
    cookie cookie_mode hash cookie_hash_name <cookie_name>
    cookie_hash_offset <cookie_value_offset> cookie_hash_length
    <cookie_value_length> <member definition> }
```

The `<cookie_name>`, `<cookie_value_offset>`, and `<cookie_value_length>` values are described in Table 2.3:

Hash mode values	Description
<code><cookie_name></code>	This is the name of an HTTP cookie being set by a Web site.
<code><cookie_value_offset></code>	This is the number of bytes in the cookie to skip before calculating the hash value.
<code><cookie_value_length></code>	This is the number of bytes to use when calculating the hash value.

Table 2.3 The cookie hash mode values

To activate sticky persistence from the command line

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

```
bigpipe pool <pool_name> modify { persist_mode sticky <enable |
    disable> sticky_mask <ip address> }
```

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

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

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

```
bigpipe pool <pool_name> sticky clear
```

To activate SSL persistence from the command line

Use the following syntax to activate SSL persistence from the command line:

```
bigpipe pool <pool_name> modify { persist_mode ssl ssl_timeout
    <timeout> simple_mask <ip_mask> }
```

For example, if you want to set SSL persistence on the pool `my_pool`, type the following command:

```
bigpipe pool my_pool modify { persist_mode ssl ssl_timeout 3600
    simple_mask 255.255.255.0 }
```

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

The complete syntax for the command is:

```
bigpipe 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 **classc_pool**:

```
bigpipe pool classc_pool modify { persist_mode simple
  simple_timeout 1200 simple_mask 255.255.255.0 }
```

You can turn off a persist mask on 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:

```
bigpipe pool classc_pool modify { simple_mask none }
```

Display persistence information for a pool

To show the persistence configuration for the pool:

```
bigpipe pool <pool_name> persist show
```

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

```
bigpipe pool classc_pool persist show
```

Options

Use the following elements to construct pools:

Pool Element	Description
Pool name	A string from 1 to 31 characters, for example: new_pool
Member definition	member <ip address>:<port> [ratio <value>] [priority <value>]
lb_mode_specificaton	lb_mode [rr ratio priority fastest least_conn predictive observed ratio_member priority_member least_conn_member]
persist_mode_specification	persist_mode [cookie simple ssl sticky]

Table 2.4 *The elements you can use to construct a pool.*

Deleting a pool

To delete a pool use the following syntax:

```
bigpipe pool <pool_name> delete
```

All references to a pool must be removed before a pool can be deleted.

Modifying pools

You can use the command line to add or delete members from a pool. You can also modify the load balancing mode for a pool from the command line. To add a new member to a pool use the following syntax:

```
bigpipe pool <pool_name> add { 1.2.3.2:telnet }
```

To delete a member from a pool use the following syntax:

```
bigpipe pool <pool_name> delete { 1.2.3.2:telnet }
```

Display pools

Use the following syntax to display all pools:

```
bigpipe pool show
```

Use the following syntax to display a specific pool:

```
bigpipe pool <pool_name> show
```

port

```
bigpipe port <port> [...<port>] limit <max conn>
```

```
bigpipe port <port> [...<port>] enable | disable | show
```

Description

Enables and disables network traffic on virtual ports, and also sets connection limits on ports. You can use standard port numbers, service or port names (for example, **www**, **http**, or **80**) for the **<port>** parameter. Note that the port settings you define with this command control the port service for all virtual servers that use the port. By default, all ports are disabled.

A port is any valid port number, between 0 and 65535, inclusive, or any valid service name in the **/etc/services** file.

Allowing and denying virtual ports

You can enable or disable traffic to specific virtual ports. The default setting for all virtual ports is disabled. Use the following syntax to allow one or more virtual ports:

```
bigpipe port <port> [...<port>] enable
```

To deny access to one or more virtual ports:

```
bigpipe port <port> [...<port>] disable
```

Setting connection limits on ports

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

```
bigpipe port <port> [...<port>] limit <max conn>
```

To turn off a connection limit for one or more ports, use the preceding command, setting the **<max conn>** parameter to **0** (zero):

```
bigpipe port <port> [...<port>] limit 0
```


Displaying the status of all virtual ports

Use the following syntax to display the status of virtual ports included in the configuration:

```
bigpipe port show
```

Displaying the status for specific virtual ports

Use the following syntax to display the status of one or more virtual ports:

```
bigpipe port <port> [...<port>] show
```

Figure 2.6 shows a sample of formatted output of the port command.

```
bigpipe port telnet show  
  
PORT 23      telnet      enable  
(cur, max, limit, tot, reaped) = (37,73,100,691,29)  
      (pckts,bits) in = (2541, 2515600), out = (2331, 2731687)
```

Figure 2.6 Formatted output of **port** command showing the Telnet port statistics

proxy

```
bigpipe proxy <ip>:<port> [/bitmask] [<ifname>] [<unit id>] target
    <server | vip> <ip>:<port> ssl enable key <key> cert <cert>
bigpipe proxy <ip>:<port> [<ifname>] [<unit id>] netmask <ip>
    [broadcast <ip>] target <server | vip> <ip>:<port> ssl enable
    key <key> cert <cert>
bigpipe proxy <ip>:<port> enable
bigpipe proxy <ip>:<port> disable
bigpipe proxy <ip>:<port> delete
bigpipe proxy <ip>:<port> show
bigpipe proxy <ip>:<port> lasthop pool <pool_name>
```

Description

Use the proxy command to create, delete, modify, or display the SSL gateway definitions on the BIG-IP Controller. For detailed information about setting up the SSL Accelerator feature, see the *BIG-IP Administrator Guide, Configuring an SSL Accelerator*.

Creating an SSL gateway from the command line

Use the following command syntax to create an SSL gateway. Use this syntax if you want to configure a gateway by specifying a bitmask instead of a netmask and broadcast address:

```
bigpipe proxy <ip>:<port> [/bitmask] [<ifname>] [<unit id>] target
    <server | vip> <ip>:<port> ssl enable key <key> cert <cert>
```

Use this syntax if you want to configure a gateway by specifying a netmask and broadcast address instead of a bitmask:

```
bigpipe proxy <ip>:<port> [<ifname>] [<unit id>] netmask <ip>
    [broadcast <ip>] target <server | vip> <ip>:<port> ssl enable
    key <key> cert <cert>
```

For example, you can create an SSL gateway from the command line that looks like this:

```
bigpipe proxy 10.1.1.1:443 exp0 unit 1 { netmask 255.255.255.0
  broadcast 10.1.1.255 target vip 20.1.1.1:80 ssl enable key
  my.server.net.key cert my.server.net.cert }
```

Note that when the configuration is written out in the **bigip.conf** file, the line **ssl enable** is automatically added. When the SSL gateway is written in the **/etc/bigip.conf** file, it looks like this:

```
proxy 10.1.1.1:443 exp0 unit 1 {
  netmask 255.255.255.0
  broadcast 10.1.1.255
  target vip 20.1.1.1:80
  ssl enable
  key my.server.net.key
  cert my.server.net.cert
}
```

Figure 2.7 An example SSL gateway configuration

Enabling, disabling, or deleting an SSL gateway from the command line

You can enable, disable, or delete an SSL gateway with the following syntax:

```
bigpipe proxy <ip>:<port> enable
bigpipe proxy <ip>:<port> disable
bigpipe proxy <ip>:<port> delete
```

For example, if you want to enable the SSL gateway 209.100.19.22:443, type the following command:

```
bigpipe proxy 209.100.19.22:443 enable
```

For example, if you want to disable the SSL gateway 209.100.19.22:443, type the following command:

```
bigpipe proxy 209.100.19.22:443 disable
```

For example, if you want to delete the SSL gateway 209.100.19.22:443, type the following command:

```
bigpipe proxy 209.100.19.22:443 delete
```

Displaying configuration information for an SSL gateway from the command line

Use the following syntax to view the configuration for the specified SSL gateway:

```
bigpipe proxy <ip>:<port> show
```

For example, if you want to view configuration information for the SSL gateway 209.100.19.22:80, type the following command:

```
bigpipe proxy 209.100.19.22:80 show
```

```
SSL PROXY +---> 11.12.1.200:443 -- Originating Address -- Enabled    Unit 1
|           | Key File Name balvenie.scotch.net.key
|           | Cert File Name balvenie.scotch.net.cert
|           | LastHop Pool Name
+====> 11.12.1.100:80 -- Destination Address -- Server

SSL PROXY +---> 11.12.1.120:443 -- Originating Address -- Enabled    Unit 1
|           | Key File Name balvenie.scotch.net.key
|           | Cert File Name balvenie.scotch.net.cert
|           | LastHop Pool Name
+====> 11.12.1.111:80 -- Destination Address -- Vip
```

*Figure 2.8 Output from the **bigpipe proxy show** command*

Adding a last hop pool to an SSL gateway from the command line

Use the following syntax to reference a last hop pool from an SSL gateway:

```
bigpipe proxy <ip>:<port> lasthop pool <pool_name>
```

For example, if you want to assign the last hop pool named **ssllasthop_pool** to the SSL gateway **11.12.1.200:443**, type the following command:

```
bigpipe proxy 11.12.1.200:443 lasthop pool
    ssllasthop_pool
```

-r

bigpipe -r

Description

Use the following syntax to clear the configuration and counter values from memory:

bigpipe -r

◆ WARNING

This command should be used with caution. All network traffic stops when you run this command.

Typically, this command is used on a standby BIG-IP Controller prior to loading a new **/etc/bigip.conf** file that contains new **tping** and **treaper** values.

For example, you can execute the following commands on a standby BIG-IP Controller:

bigpipe -r

bigpipe -f <filename>

This sequence of commands ensures that only the values set in the **<filename>** specified are in use.

ratio

```
bigpipe ratio [<node ip>] [<node ip> ...] show  
bigpipe ratio <node ip> [<node ip>...] <weight>
```

Description

This command provides two functions related to load balancing:

- ❖ For the Ratio load balancing mode, the 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.

Setting ratio weight for one or more node addresses

The default ratio setting for any node address is **1**. If you use the Ratio or Priority load balancing modes, 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 modes have the same affect as the Round Robin load balancing mode.

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

```
bigpipe ratio <node ip> [...<node ip>] <weight>
```

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

```
bigpipe ratio 192.168.103.20 3
```

Displaying the ratio weights for node addresses

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

```
bigpipe ratio show
```

The command displays the following output:

```
192.168.200.51    ratio = 3
```

```
192.168.200.52    ratio = 1
```

Displaying ratio weight for specific node addresses

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

```
bigpipe ratio <node ip> [...<node ip>] show
```

◆ Note

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

rule

```
bigpipe rule <rule name> ' { <if statement> | <use statement> } '
bigpipe rule <rule name> delete
bigpipe rule [<rule name>] show
```

Description

Use the **rule** command to create, delete, or display the rules on the BIG-IP Controller. Rules allow a virtual server to access any number of pools on the BIG-IP Controller. For more detailed information about using rules, see the ***BIG-IP Administrator Guide**, Working with Intelligent Traffic Control*.

◆ Note

Before you define a rule, you must define the pool or pools that you want the rule to reference.

Create a rule

You can add rules by manually typing them into an existing `/etc/bigip.conf` file. However, you can use the **bigpipe rule** command to create, delete, or display rules. To create a rule with **bigpipe**, type the complete rule on the command line without line breaks. For example, you can type in this rule:

```
bigpipe rule cgi_rule ' {if (http_uri ends_with
    "cgi") {use ( cgi_pool )} else {use (
    default_pool )}} '

```

If the **http_uri** string ends with "cgi" then the members from **cgi_pool** are used for load balancing. If the **http_uri** string does not end with "cgi", then the members of **default_pool** are used for load balancing.

Associating a rule with Virtual Server

You can associate a rule with a virtual server by using the following syntax:


```
bigpipe vip <virt ip>:<port> use rule <rule_name>
```

For example, if you want to associate the rule **cgi_rule** to the virtual server **10.20.2.101:http**, type in the following command:

```
bigpipe vip 10.20.2.101:http use rule cgi_rule
```

Deleting a rule

You can delete a rule using the following syntax:

```
bigpipe rule <rule_name> delete
```

Display rules

Use the following syntax to display all rules:

```
bigpipe rule show
```

Use the following syntax to display a specific rule:

```
bigpipe rule <rule_name> show
```

Definitions

You can create a rule by combining a number of different elements. A simple rule could contain the following elements:

```
rule <rule_name> { if ( <variable>
    <binary_operator> "<literal>" ) { use (
    <pool_name> ) } else { use (
    <another_pool_name> ) } }
```

For example, a rule named **cgi_rule** that sends CGI connections to a load balancing pool named **cgi_pool**, or HTTP connections to a pool named **http_pool** looks like this:

```
bigpipe rule cgi_rule ' {if (http_uri ends_with
    "cgi") {use ( cgi_pool )} else {use ( http_pool
    )}} '
```

Use the elements in Table 2.5 to create rules.

Element	Description
A rule definition is	<code>rule { <statement> }</code>
A statement is	<code><use_statement></code> <code><if_statement></code> <code>discard</code>
A use statement	<code>use (<pool_name>)</code>
An if statement	<code>if (<expression>) { <statement> }</code> <code>[{ else <statement> }]</code>
An expression	<code><literal></code> <code><variable></code> <code>(<expression>)</code> <code>exist <variable></code> <code>not <expression></code> <code><expression> <binary_operator> <expression></code>
literal	<code><regex_literal></code> <code><string_literal></code> <code><address_literal></code>
A regular expression literal	Is a string of 1 to 63 characters enclosed in quotes that may contain regular expressions
A string literal	Is a string of 1 to 63 characters enclosed in quotes
An address literal	<code><dot_notation_longword> [netmask <dot_notation_longword>]</code>

Table 2.5 *The elements you can use to construct rules.*

Element	Description
Dot notation longword	<0-255>.<0-255>.<0-255>.<0-255>
variable	<pre> http_method http_header http_version http_uri http_host http_cookie <cookie_name> client_addr ip_protocol </pre>
binary operator	<pre> or and contains matches equals starts_with ends_with matches_regex </pre>

Table 2.5 *The elements you can use to construct rules.*

-S

```
bigpipe -s [ <filename> | - ]
```

Description

Writes the current BIG-IP Controller configuration settings from memory to the default boot configuration file named **/etc/bigip.conf**.

You can just type **bigpipe -s**, or a hyphen character (-) in place of a file name, to display the configuration on the standard output device.

```
bigpipe -s -
```

Or you can simply type the following command:

```
bigpipe -s
```

If you are testing and integrating BIG-IP Controllers 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:

```
bigpipe -s <filename>
```

For example, the following command saves the current configuration from memory to an alternate configuration file named **/etc/bigip.conf2**.

```
bigpipe -s /etc/bigip.conf2
```

snat

```

bigpipe snat map <node ip> [...<node ip>] to \
    <SNAT ip> [netmask <ip>] [<ifname>] [unit <unit ID>]
bigpipe snat map default to <SNAT ip> [<ifname>] \
    [unit <unit ID>] [netmask <ip>]
bigpipe snat <SNAT ip> [...<SNAT ip>] delete
bigip snat default delete
bigpipe snat default dump [verbose]
bigpipe snat [<node ip> [...<node ip>] ] dump [verbose]
bigpipe snat globals show
bigpipe snat default show
bigpipe snat [<node ip> [...<node ip>] ] show
bigpipe snat limit <max conn>
bigpipe snat default limit <max conn>
bigpipe snat <node ip> [...<node ip>] limit \
    <max conn>
bigpipe snat <node ip> [...<node ip>] mirror \
    enable | disable
bigpipe snat default mirror enable | disable
bigpipe snat <node ip> [...<node ip>] timeout tcp | udp \
    <seconds>
bigpipe snat [default] timeout tcp | udp <seconds>
bigpipe snat <SNAT ip> [...<SNAT ip>] stats reset
bigpipe snat default stats reset

```

Description

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.

Defining the default SNAT

Use the following syntax to 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. You can use the **unit** `<unit ID>` parameter to specify a unit in an active-active redundant configuration.

```
bigpipe snat map default to <SNAT ip> [<ifname>] [unit <unit ID>]
[netmask <ip>]
```

Creating individual SNAT addresses

Use the following command syntax to create a SNAT mapping:

```
bigpipe snat map <node ip> [...<node ip>] to \
<SNAT ip> [<ifname>] [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.

Deleting SNAT Addresses

The following syntax deletes a specific SNAT:

```
bigpipe snat <SNAT ip> | default delete
```

Showing SNAT mappings

The following **bigpipe** command shows mappings:

```
bigpipe snat [<SNAT ip>] [...<SNAT ip>] show
bigpipe snat default show
```

The following command shows the current SNAT connections:

```
bigpipe snat [<SNAT ip>] [...<SNAT ip>] dump [ verbose ]
bigpipe snat default dump [ verbose ]
```

The optional **verbose** keyword provides more detailed output.

The following command prints the global SNAT settings:

```
bigpipe snat globals show
```

Limiting connections

Use the following commands to set the maximum number of concurrent connections allowed for one or more SNAT addresses. Zero indicates no limit.

```
bigpipe snat <SNAT ip> limit <max conn>
```

The default SNAT address connection limit is set with the following command:

```
bigpipe snat default limit <max conn>
```

Set global concurrent connection limit:

```
bigpipe snat limit <max conn>
```

Enabling mirroring for redundant systems

The following example sets SNAT mirroring for all SNAT connections originating at 192.168.225.100:

```
bigpipe snat 192.168.225.100 mirror enable
```

Setting idle connection timeouts

Use the following command to set the timeout for idle TCP connections:

```
bigpipe snat timeout tcp <seconds>
```

Use the following command to set the timeout for idle UDP connections. Note that you must have a timeout set for UDP connections; zero is not allowed:

```
bigpipe snat timeout udp <seconds>
```

Use the following command to set the timeout for idle TCP connections originating at this node address. Set **<seconds>** to 0 (zero) to disable TCP timeout for these nodes.

```
bigpipe snat <node ip> [...<node ip>] timeout tcp <seconds>
```

Use the following command to set the timeout for idle TCP connections originating at the default node address. Set **<seconds>** to 0 (zero) to disable TCP timeout for these nodes.

```
bigpipe snat default timeout tcp <seconds>
```

Use the following syntax to set the timeout for idle UDP connections originating at this node address. Note that you must have a timeout set for UDP connections; zero is not allowed:

```
bigpipe snat <node ip> [ ...<node ip> ] timeout udp <seconds>
```

Use the following syntax to set the timeout for idle UDP connections originating at the default SNAT address. Note that you must have a timeout set for UDP connections; zero is not allowed:

```
bigpipe snat default timeout udp <seconds>
```

Clearing statistics

You can reset statistics by node or by SNAT address. Use the following syntax to clear all statistics for one or more nodes:

```
bigpipe snat <node ip> [ ...<node ip> ] stats reset
```

Use the following syntax to clear all statistics for one or more SNAT addresses:

```
bigpipe snat <SNAT ip> [ ...<SNAT ip> ] stats reset
```

Use the following command to reset the statistics to zero for the default:

```
bigpipe snat default stats reset
```


summary

bigpipe summary

Description

Displays a summary of current usage statistics.

The output display format for the **summary** command is shown in Figure 2.9.

```

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 vip tcp port deny              = 0
  BIG-IP vip udp port deny              = 0
  BIG-IP max connections deny          = 0
  BIG-IP vip duplicate syn ssl          = 0
  BIG-IP vip duplicate syn wrong dest   = 0
  BIG-IP vip duplicate syn node down    = 0
  BIG-IP vip maint mode deny           = 0
  BIG-IP virtual addr max connections deny = 0
  BIG-IP virtual path max connections deny = 0
  BIG-IP vip non syn                   = 0
  BIG-IP error not in out table        = 0
  BIG-IP error not in in table         = 0
  BIG-IP error vip fragment no port    = 0
  BIG-IP error vip 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 2.9 Summary output display

For detailed descriptions of each of statistic displayed by the **summary** command, refer to the ***BIG-IP Controller Administrator Guide, Monitoring and Administration.***

timeout_node

```
bigpipe timeout_node show
bigpipe timeout_node <seconds>
bigpipe timeout_node 0
```

Description

Sets the amount of time that a server has to respond to a BIG-IP Controller ping in order for the server to be marked **up**. If a server fails to respond within the specified time, the BIG-IP Controller assumes that the server is down, and the BIG-IP Controller no longer sends packets to the services hosted by the server. If the server responds to the next ping, or to subsequent pings, the BIG-IP Controller then marks the server **up**, and resumes sending packets to those services.

The default is 15 seconds.

◆ Note

*If the **timeout_node** interval is shorter than the **timeout_svc** setting, a node can be marked **down** before the services on the node are marked **down**.*

Displaying the current timeout value

Use the following command to display the current timeout setting for node ping:

```
bigpipe timeout_node show
```

Setting a timeout value for node ping

Use the following syntax to set the timeout setting for node ping:

```
bigpipe timeout_node <seconds>
```

The sample command below sets the timeout to 33 seconds.

```
bigpipe timeout_node 33
```

Disabling node ping

To disable node ping, you simply set the node ping timeout value to 0 (zero):

```
bigpipe timeout_node 0
```

◆ WARNING

*Node ping is the only form of verification that the BIG-IP Controller uses to determine status of node addresses. If you turn node ping off while one or more node addresses are currently **down**, the node addresses remain marked **down** until you turn node ping back on and allow the BIG-IP Controller to verify the node addresses again.*

timeout_svc

```
bigpipe timeout_svc [<port>] show
bigpipe timeout_svc <port> <seconds>
bigpipe timeout_svc <port> 0
```

Description

Sets the amount of time that a specific node has to respond to a service check issued by the BIG-IP Controller. There are three types of service checks, each of which is affected by this setting:

- ❖ Simple service check where the BIG-IP Controller attempts to establish a connection to the service hosted by the node
- ❖ Extended content verification where the BIG-IP Controller requests specific content from the node
- ❖ Extended application verification where the BIG-IP Controller executes an external service check program that verifies whether or not specific content is available on the node

If a node fails to respond to any type of service check within the specified time, the BIG-IP Controller assumes that the service is down and no longer sends client requests to the service. If the node responds to the next service check, or to subsequent service checks, the BIG-IP Controller marks the service **up**, and resumes sending requests to the service.

◆ WARNING

*The BIG-IP Controller does not attempt to detect the status of a node if node ping is turned off (**bigd -n**) and the **timeout_svc** and **tping_svc** values are set to 0 for a particular node.*

The **timeout_svc** default for each port is set to **0**, which disables service checks on the port.

Note that the BIG-IP Controller monitors only those services that have a **timeout_svc** and **tping_svc** value greater than 0.

Setting the service check timeout

Use the following syntax to set the service check timeout for a specific node port. Note that this setting applies to all nodes that use the port.

```
bigpipe timeout_svc <port> <seconds>
```

For example, the following command sets the service check timeout on port 80 to 120 seconds:

```
bigpipe timeout_svc 80 120
```

Disabling the service check

To disable service check on a specific port, use the above command, but set the **<seconds>** parameter to zero:

```
bigpipe timeout_svc <port> 0
```

Displaying service check timeouts

Use the following command to display the current service check timeout settings for all ports:

```
bigpipe timeout_svc show
```

The system displays the following output:

```
port 80 timeout after 120 seconds
```

The system only displays ports that have a timeout set to a value other than 0.

Use the following syntax to display the current service check timeout setting for a specific port:

```
bigpipe timeout_svc <port> [show]
```

tping_node

```
bigpipe tping_node show
bigpipe tping_node <seconds>
```

Description

Sets the interval (in seconds) at which a BIG-IP Controller issues a ping to each server managed by the BIG-IP Controller. If a specific server responds to the ping within a set time, the server is marked **up** and the BIG-IP Controller sends connections to the services hosted by that server. If a server fails to respond to a ping within the specified time, the BIG-IP Controller assumes that the server is no longer available, and it marks the node **down**.

Note that the **timeout_node** setting determines the number of seconds that a server has to respond to the ping issued by the BIG-IP Controller.

The default setting for **tping_node** is 5 seconds.

Setting a node ping interval

Use the following syntax to set the number of seconds which a server has to respond to a ping issued by the BIG-IP Controller:

```
bigpipe tping_node <seconds>
```

Disabling node ping

To turn node ping off, set the interval to **0** seconds:

```
bigpipe tping_node 0
```

Displaying the current node ping setting

Use the following command to display the current node ping setting:

```
bigpipe tping_node show
```


tping_svc

```
bigpipe tping_svc show
```

```
bigpipe tping_svc <port> <seconds>
```

```
bigpipe tping_svc <port> 0
```

Description

Sets the interval (in seconds) at which BIG-IP Controller issues a service check to one or more specific nodes included in the configuration. There are three types of service check, each of which is affected by this setting:

- ❖ Simple Service check where the BIG-IP Controller attempts to establish a connection to the service hosted by the node
- ❖ Extended content verification where the BIG-IP Controller requests specific content from the node
- ❖ Extended application verification where the BIG-IP Controller executes an external service check program that verifies whether or not specific content is available on the node

If a node fails to respond to a service check within the time specified by the **timeout_svc** setting, the BIG-IP Controller marks the service **down**, and no longer routes client requests to it.

◆ WARNING

*The BIG-IP Controller does not attempt to detect the status of a node if node ping is turned off (**bigd -n**) and the **timeout_svc** and **tping_svc** values are set to 0 for a node.*

Setting global service check intervals for a node port

Use the following syntax to set a service check interval for a specific node port.

```
bigpipe tping_svc <port> <seconds>
```

Use the following syntax to turn service check off for a specific node port.

```
bigpipe tping_svc <port> 0
```

Displaying the current service check interval

Use the following syntax to display the intervals at which the BIG-IP Controller issues service checks to all nodes configured for service check:

```
bigpipe tping_svc show
```

treaper

```
bigpipe treaper show
bigpipe treaper <port> <seconds>
bigpipe treaper <port> 0
```

Description

Sets the expiration time for idle TCP connections on a specific port. An idle connection is one in which no data has been received or sent for the number of seconds specified by the **treaper** command. The **treaper** default value is 1005 seconds. For **treaper** to be effective, you should set its value to be greater than the configured timeout for the service daemons installed on your nodes.

The **treaper** command clears the connection tables, avoiding memory problems due to the accumulation of dead, but not terminated, connections.

Setting the idle TCP connection timeout for a virtual port

Use the following syntax to set an inactive connection timeout for one or more virtual ports:

```
treaper <port> <seconds>
```

To turn the inactive connection timeout off, use the same command but set the number of seconds to zero:

```
treaper <port> 0
```

◆ Note

Typical settings include 120 seconds for 25/SMTP, 120 seconds for 80/www, 300-600 seconds for 20/ftp-data and 21/ftp-data.

Displaying the current inactive connection timeout

Use the following syntax to display the current number of seconds that connections are allowed to remain idle before being dropped:

```
bigpipe treaper show
```

udp

```
bigpipe udp [<port> [...<port>] ] show
bigpipe udp <port> [...<port>] <seconds>
bigpipe udp <port> 0
```

Description

The **udp** command enables UDP traffic on virtual ports and also sets a timeout for idle UDP connections. UDP traffic is enabled only when the timeout is set to a value greater than 0 (zero). You can disable UDP traffic on a port by setting the idle connection timeout to 0 (zero). By default, UDP is disabled on all ports.

Setting the idle connection timeout for UDP traffic

Use the following syntax to set the UDP timeout on one or more virtual ports, where the **<seconds>** parameter is the number of seconds before an idle connection is dropped:

```
bigpipe udp <port> <seconds>
```

For example, the following command sets the UDP timeout to 300 seconds for port 53:

```
bigpipe udp 53 300
```

To turn UDP timeout off for a virtual port, use the above command, setting the **<seconds>** parameter to zero:

```
bigpipe udp <port> 0
```

Displaying UDP settings

Use the following command to display the UDP timeout setting for all ports that allow UDP:

```
bigpipe udp show
```

Use the following syntax to display the timeout setting for a specific virtual port that allows UDP:

```
bigpipe udp <port> show
```

The system displays the output:

```
port 53 idle udp connections expire after 300
seconds <$startrange> BIG/pipe commands: udp;
```

unit

unit [show]

unit peer [show]

Description

The unit number on a BIG-IP Controller designates which virtual servers use a particular controller in an active-active redundant configuration. You can use the **bigpipe unit** command to display the unit number assigned to a particular BIG-IP Controller. For example, to display the unit number of the unit you are on, type the following command:

```
bigpipe unit show
```

To display the unit number of the other controller in a redundant system, type in the following command:

```
bigpipe unit peer show
```

◆ Note

If you use this command on a redundant system in active/standby mode, the active controller shows as unit 1 and 2, the standby controller has no unit numbers.

◆ Note

*The **bigpipe unit peer show** command is the best way to determine whether the respective state mirroring daemons are connected.*

`-V`

`bigpipe -v`

Description

Displays version number of the BIG/pipe command utility.

For example, **bigpipe -v** displays the following output:

```
bigpipe: 3.0
```


version

bigpipe version

Description

Displays the version number of the BIG-IP Controller's operating system.

The **bigpipe version** command outputs the following version information:

```
BIG-IP: version 3.0
```

vip

```

bigpipe vip <virt ip>[:<port>] [<ifname>] [unit <ID>] \
    [netmask <ip>] [broadcast <ip>] use pool <pool_name>
bigpipe vip <virt ip>:<port>[/<bitmask>] [<ifname>] [unit <ID>] \
    use pool <pool_name>
bigpipe vip <virt ip>[:<port>] [<ifname>] [unit <ID>] \
    [netmask <ip>] [broadcast <ip>] use rule <rule_name>
bigpipe vip <virt ip>:<port>[/<bitmask>] [<ifname>] [unit <ID>] \
    use rule <rule_name>
vip [<virt ip>[:<port>]] [...<virt ip>[:<port>]] show
vip <virt ip>[:<port>] [<ifname>] [ ... <virt ip>[:<port>]] \
    enable | disable | delete
vip <virt ip>[:<port>] [... <virt ip>[:<port>]] limit \
    <max conn>
vip <virt ip>:<port> translate port enable | disable | show
vip <virt ip>:<port> translate addr enable | disable | show
vip <virt ip>:<port> lasthop pool <pool_name> | none | show
vip <virt ip>:<port> mirror conn enable | disable | show
vip [<virt ip>:<port>] stats reset

```

Description

Creates, deletes, and displays information about virtual servers. This command also sets connection mirroring, connection limits, and timeouts on a virtual server.

Defining a virtual server

Virtual servers are port-specific, and if you are configuring a site that supports more than one service, you need to configure one virtual server for each service offered by the site. Use the following syntax to define the pools or rules to which a virtual server maps. The **unit <ID>** parameter specifies which unit handles the virtual server in an active-active redundant configuration. You can

associate pools or rules with a virtual server. The following sections describe the syntax for associating a pool or a rule with a virtual server.

Configuring a virtual server to use a load balancing pool

Use the following syntax to create a virtual server that references a load balancing pool. Note that you must create a pool before you can create a virtual server that references the pool. For information about creating a pool, see *Creating a pool*, on page 2-34.

```
bigpipe vip <virt ip>:<port> [ifname] [unit <ID>] use pool
    <pool_name>
```

For example, if you want to create a virtual server that references the pool **my_pool**, the command might look like this:

```
bigpipe vip 11.12.1.53:80 use pool my_pool
```

Configuring a virtual server to use a load balancing rule

Use the following syntax to create a virtual server that references a load balancing rule. Note that you must create a rule before you can create the virtual server that references the rule. For information about creating a rule, see *Associating a rule with a virtual server*, on page 2-50.

```
bigpipe vip <virt ip>:<port> [ifname] [unit <ID>] use rule
    <rule_name>
```

For example, if you want to create a virtual server that references the rule **my_rule**, the command might look like this:

```
bigpipe vip 11.12.1.53:80 use pool my_rule
```

Displaying information about virtual servers

Use the following syntax to display information about all virtual servers included in the configuration:

```
bigpipe vip show
```

Use the following syntax to display information about one or more virtual servers included in the configuration:

```
bigpipe vip <virt ip>:<port> [...<virt ip>:<port>] show
```

The command displays information such as the nodes associated with each virtual server, the nodes' status, and the current, total, and maximum number of connections managed by the virtual server since the BIG-IP Controller was last rebooted.

Defining an interface for a virtual server

If you have multiple external (destination processing) interfaces, you can specify one of them when you define a virtual server. If you specify an interface name, the BIG-IP Controller responds to ARP requests for the virtual address on that interface. If you do not specify an interface name, the BIG-IP Controller responds to ARP requests for the virtual server on the default interface. If you do not want the BIG-IP Controller to respond to ARP requests on any interface, use the option **none** in place of the an **<ifname>** parameter.

All virtual servers that share a virtual address must use the same external interface. Changing the interface for a virtual server changes the interface for all virtual servers having the same virtual address.

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 BIG-IP Controller, and it is based on the virtual address and the current netmask. You can override the default netmask and broadcast for any virtual address.

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.

Note that if you want to use a custom netmask and broadcast, you define both when you define the virtual server:

```
bigpipe vip <virt ip>[:<port>] [<ifname>] [netmask <ip>] \
  [broadcast <ip>] use pool <pool_name>
```

◆ Note

The BIG-IP Controller calculates the broadcast based on the IP address and the netmask. A user-defined broadcast address is not necessary.

Again, even when you define a custom netmask and broadcast in a specific 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:

```
bigpipe vip 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:

```
bigpipe vip 206.168.225.1:80/24 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 a connection limit

The default setting is to have no limit to the number of concurrent connections allowed on a virtual server. You can set a concurrent connection limit on one or more virtual servers using the following command:

```
bigpipe vip <virt ip>[:<port>] [...<virt ip>[:<port>] ] limit \
  <max conn>
```

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

```
bigpipe vip www.SiteOne.com:http www.SiteTwo.com:ssl limit 5000
```

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

```
bigpipe vip <virt ip>[:<port>] [...<virt ip>[:<port>] ] limit 0
```

Setting translation properties for virtual addresses and ports

Turning port translation off for a virtual server is useful if you want to use the virtual server to load balance connections to any service. Use the following syntax to enable or disable port translation for a virtual server.

```
bigpipe vip <virt ip>:<port> translate port enable | disable | show
```

You can also configure the translation properties for a virtual server address. This option is useful when the BIG-IP Controller is load balancing devices which 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. Use the following syntax to enable or disable address translation for a virtual server.

```
bigpipe vip <virt ip>:<port> translate addr enable | disable | show
```

Setting up last hop pools for virtual servers

In cases where you have more than one router sending connections to a BIG-IP redundant system, you may want to route connections back through the same router from which they were received. To configure a last hop pool, you must first create a pool that contains the routers for the BIG-IP redundant system. After you create a router pool, use the following syntax to configure a last hop pool for a virtual server.

```
bigpipe vip <virt ip>:<port> lasthop pool <pool_name> | none | show
```

Mirroring connection information

Mirroring provides seamless recovery for current connections and when a BIG-IP Controller fails. When you use the mirroring feature, the peer controller maintains the same current connection and persistence information as its partner controller. Transactions such as FTP file transfers continue as though uninterrupted.

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

```
bigpipe vip <virt ip>:<port> mirror conn enable | disable
```

To print the current mirroring setting for a virtual server:

```
bigpipe vip <virt ip>:<port> mirror conn show
```

If you do not specify **conn**, the BIG-IP Controller displays all mirrored connection 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:

```
bigpipe vip v1:21 mirror conn enable
```

```
bigpipe vip v1:20 mirror conn enable
```

Removing and returning a virtual server to service

You can remove an existing virtual server from network service, or return the virtual server to service, using the **disable** and **enable** keywords. 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 **down**. Use the following syntax to remove a virtual server from network service:

```
bigpipe vip <virt ip>:<port> [...<virt ip>:<port>]
      disable
```

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

```
bigpipe vip <virt ip>:<port> enable
```

Removing and returning a virtual address to service

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.

```
bigpipe vip <virt ip> disable
```

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

```
bigpipe vip <virt ip> enable
```

Displaying information about virtual addresses

You can also 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:

```
bigpipe vip <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 BIG-IP Controller was last rebooted, or since the BIG-IP Controller became the active unit (redundant configurations only).

Deleting a virtual server

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

```
bigpipe vip <virt ip>:<port> [... <virt ip>:<port>] delete
```


Resetting statistics for a virtual server

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

```
bigpipe vip [<vip ip:port>] stats reset
```

Backward-compatible commands

The following BIG/pipe commands have been included for users of previous versions.

```

dt [<ip>[:<port> ] ]
port <port> [<port>... ] [allow | deny] [ limit <limit> ]
vip <virt ip>:<port> persistmask [ <IP address mask> ]
vip <virt ip>:<port> persistmask [ none | show ]
vip <virt ip>[:<port>] [<ifname>] netmask <ip> \
    [ broadcast <ip> ] define <node ip>[:<port>] \
    [ <node ip>[:<port>... ] [ special ssl <value> <value> ]
nat <node ip> to <NAT ip> [<ifname>] netmask <ip> \
    [ broadcast <ip> ]
fo [ master | slave ]
vip <virt ip>[:<port>] [/<bitmask>] [<ifname>|none ] \
    [unit <unit ID>] define <node ip>[:<port>] \
    [...<node ip>[:<port>] ] [special ssl <seconds> <seconds>]
vip <virt ip>[:<port>] netmask <ip> [broadcast <ip>] \
    [<ifname> | none ] [unit <unit ID>] define <node ip>[:<port>] \
    [...<node ip>[:<port>] ] [special ssl <seconds> <seconds>] \
    [special cookie insert | rewrite | passive <days>d <hh:mm:ss>]]
vip <virt ip>[:<port>] netmask <ip> [broadcast <ip>] \
    [<ifname> | none ] [unit <unit ID>] define <node ip>[:<port>] \
    [...<node ip>[:<port>] ] [special cookie hash <name> <offset>
    <length>]
vip <virt ip>:<port> mirror persist enable | disable | show
vip <virt ip>:<port> persist show | dump | <value>
vip <virt ip>:<port> persist mask <ip> | none | show
vip 0.0.0.0:<port> sticky [ enable | disable | show | clear | dump ]
vip 0.0.0.0:<port> sticky mask [ <ip> | none | show ]
vip sticky dump
vip sticky clear

```

3

Configuration Files

Configuration files for the BIG-IP Controller

File	Description
/etc/bigip.conf	Stores virtual server and node definitions and settings, including node ping settings, the load balancing mode, and NAT and SNAT settings.
/etc/bigconf.conf	Stores the user preferences for the Configuration utility.
/etc/bigd.conf	Stores service check settings.
/etc/hosts.allow	Stores the IP addresses of workstations that are allowed to make administrative shell connections to the BIG-IP Controller.
/etc/netstart	Stores basic system start up settings.
/etc/ipfw.conf	Stores IP filter settings.
/etc/rateclass.conf	Stores rate class definitions.
/etc/ipfwrate.conf	Stores IP filter settings for filters that also use rate classes.
/etc/snmpd.conf	Stores SNMP configuration settings.
/etc/bigip.license	Stores authorization information for the BIG-IP Controller.
/etc/syslog.conf	Stores the configuration files for syslogd under the BIG-IP Controller.
/etc/rc.sysctl	Stores the default UNIX and the BIG-IP Controller <i>sysctl</i> variables.
/etc/hosts	Stores the hosts table for the BIG-IP Controller.
/etc/rc.local	Stores the local daemons, filters, local boot settings for the BIG-IP Controller.
/etc/irs.conf	Controls information retrieval functions in the C library.
/etc/login.conf	UNIX system file, modified for the BIG-IP Controller.
/etc/rc	UNIX system startup script, modified for the BIG-IP Controller.
/etc/sshd_config	This is the configuration file for the secure shell server. It contains all the access information for people trying to get into the system via ssh.
/etc/wideip.conf	This is a 3DNS Controller configuration file. For more information, please refer to the documentation for that product.
/VENDOR	This file contains information describing F5 Networks. It includes the company name, a common name, contact information, and the text of the licensing agreement for the software.

File	Description
/VERSION	Contains name of the product, the number, and the access rights (BIG-IP 3.2 HA, for example).
/usr/contrib/bin/ssh-askpass	This is the external program used by the ssh configuration utility to ask the user for his password from an X-windows system. It allows SSH to connect to a remote site, or generate a PPKey pair, in a secure manner.
/var/f5/httpd/conf/cert.conf	The information for the public key/private key certification infrastructure for the webserver.
/var/f5/httpd/conf/httpd.conf	The main configuration file for the webserver.
/var/f5/httpd/ssl/lib/sslkey.cnf	This file holds the configuration information for how the SSL library interacts with browsers, and how key information is generated.
/var/f5/httpd/ssl/lib/sslkey.conf	This file holds the configuration information for how the SSL library interacts with browsers, and how key information is generated.
/var/f5/httpd/basicauth/users	The webserver password file. Contains the user names and passwords of the people permitted to access whatever is provided by the webserver.
/var/f5/bigdb/user.db	This is the location of the BIG/db database. This database holds various configuration information.

4

BIG-IP Controller Configuration Utilities

- Introducing the BIG-IP Controller configuration utilities

Introducing the BIG-IP Controller configuration utilities

The BIG-IP Controller includes a number of configuration utilities. These utilities allow you to reconfigure portions of your installation after initial configuration.

The following configuration utilities are available on the BIG-IP Controller:

❖ **config**

This utility is also known as the First-Time Boot utility. This utility runs all the other utilities required to configure the BIG-IP Controller.

❖ **config_failover**

Use this utility to prepare a new redundant system for the sync commands that synchronize redundant controllers.

❖ **reconfig-httpd**

Use this utility to reconfigure the web server on the BIG-IP Controller.

❖ **config_telnetd**

Use this utility to configure the Telnet and FTP daemons.

❖ **config_sshd**

Use this utility to configure the SSH daemon.

❖ **config_rshd**

Use this utility to configure the RSH daemon.

❖ **config_synckey**

Use this utility to synchronize the security keys between LB and International redundant BIG-IP Controllers.

config

Description

This utility starts automatically the first time you boot up a BIG-IP Controller. The **config** utility, referred to as the First-Time Boot utility, is a wizard that walks you through a brief series of required configuration tasks. These tasks include defining a root password and configuring IP addresses for the interfaces. You can run the First-Time Boot utility to reconfigure a controller.

The First-Time Boot utility is organized into three phases: configure, confirm, and commit. Each phase walks you through a series of screens, presenting the information in the following order:

- ❖ Root password
- ❖ Host name
- ❖ Default route (typically a router's IP address)
- ❖ Time zone
- ❖ DNS proxying
- ❖ Interface settings for the each network interface
- ❖ Configuration for BIG-IP Controller redundant systems (fail-over IP address)
- ❖ IP address for remote administration
- ❖ Remote administration access for vendor support
- ❖ Settings for the BIG-IP web server
- ❖ Web server administration access for vendor support

First, you configure all of the required information, then you have the opportunity to confirm each individual setting or correct it if necessary, and then your confirmed settings are committed and saved to the system. Note that the screens you see are tailored to the specific hardware and software configuration that you have. If you have a stand-alone system, for example, the First-Time Boot utility skips the redundant system screens.

To run the First-Time Boot utility, type in the following command:

config

config_failover

Description

You must run the **config_failover** script on the second controller in a redundant system in order to share keys with the peer BIG-IP Controller.

The script prompts you for the root password of the other controller in the redundant system. After confirming your input, the **config_failover** script attempts to access the peer system and configure both systems to communicate with one another. This provides the secure communication channel the controllers use to exchange configuration data when you run the **bigpipe configsync** option, or use the **Config Sync** button in the Configuration utility.

Type the following command on the command line to run the **config_failover** script:

```
config_failover
```

reconfig-httpd

Description

Use the **reconfig-httpd** configuration utility to reconfigure the HTTPD server on a BIG-IP Controller.

This script prompts you for an IP address from which administrators can access the BIG-IP Controller through the BIG-IP web server. You can use wildcard characters (*) to include all addresses from a specific part of the network. This utility also prompts you to create a support account for access by technical support.

If the service port for the BIG-IP web server (httpd) is closed, this script automatically opens the service port to permit access to the BIG-IP web server.

To run the BIG-IP web server configuration utility, type in the following command:

```
reconfig-httpd
```

config_telnetd

Description

Use the **config_telnetd** configuration script to configure the Telnet and FTP servers on a BIG-IP Controller. The script prompts you to configure each service independently. This allows you to enable Telnet but not FTP, for example.

The script prompts you for a configuration address for each service from which administrators may access the BIG-IP Controller. You can use wildcard characters (*) to include all addresses from a specific part of the network. This utility also prompts you to create a support account for access by technical support.

If **inetd** is not currently configured, this script configures **inetd** for the requested services. If the ports for Telnet or FTP are closed, this script opens the ports to permit Telnet or FTP connections to the BIG-IP Controller.

To run the Telnet/FTP configuration utility, type in the following command:

```
config_telnetd
```

◆ Note

*Running **config_telnetd** again replaces the current configuration.*

config_sshd

Description

Use this utility to configure the secure shell (SSH) server on a BIG-IP Controller. This utility prompts you for an IP address from which administrators may access the BIG-IP Controller with SSH. You can use wildcard characters (*) to include all addresses from a specific part of the network. This utility also prompts you to create a support account for access by technical support.

If the service port for ssh is closed, this script opens the service port to permit SSH connections to the BIG-IP Controller.

To run the secure shell configuration utility, type in the following command:

```
config_sshd
```

◆ Note

*Re-running **config_sshd** again replaces the current configuration.*

config_rshd

Description

Use the **config_rshd** configuration utility to configure the remote shell (rshd) server on a BIG-IP Controller. This utility prompts you for an IP address from which administrators may access the BIG-IP Controller. You can use wildcard characters (*) to include all addresses from a specific part of the network. This utility also prompts you to create a support account for access by technical support.

If **inetd** is not currently configured, this script configures **inetd** for the remote shell server (rshd). If the service port for **rsh** is closed, this utility opens the service port to permit rsh connections to the BIG-IP Controller.

To run the rsh configuration utility, type in the following command:

```
config_rshd
```

◆ Note

*Running **config_rshd** again replaces the current configuration.*

config_synckey

Description

Use the **config_synckey** utility to store the BIG-IP web server administrator password in an HTTP-compatible format for a BIG-IP Controller. This password format is used by the **bigpipe** and Configuration **configsnc** options on the BIG-IP Controller.

To run the key synchronization utility, type in the following command:

```
config_synckey
```

◆ Note

The web-based password is stored in the BIG/db database with an extremely simple form of encryption. This is the same form of encryption used by web browsers and clients to exchange the authentication key back and forth, so it is no less secure than actually logging in to a website.

5

BIG-IP System Control Variables

- Setting BIG-IP system control variables

Setting BIG-IP system control variables

The BIG-IP Controller hardware and software boot up with a configuration specified, in part, by the system control variables stored in the `/etc/rc.sysctl` file. Most of these variables are standard BSD UNIX system control variables, while some are used exclusively by the BIG-IP Controller. In most cases, a variable is just toggled **off (0)** or **on (1)**, but some variables may also store specific values, such as a port number.

You can use three methods to set system control variables affecting the BIG-IP Controller:

- ❖ **The Configuration utility**

Navigate to a system control variable and edit it in the browser with the Configuration utility.

- ❖ **sysctl command**

Write system control variable values directly to `/etc/rc.sysctl` using this command line utility.

- ❖ **vi or pico**

Use a text editor, such as **vi** or **pico**, to edit `/etc/rc.sysctl` directly.

sysctl

`sysctl -a`

`sysctl <variable name>`

`sysctl -w <variable name>=<value>`

Displaying current system control variable settings

To display the settings of all system control variables, use the following syntax:

`sysctl -a`

To display the current setting for an individual variable, use the following command syntax:

`sysctl <variable name>`

Setting a system control variable

Use the following syntax to write a value for a system control variable in `/etc/rc.sysctl`:

`sysctl -w <variable name>=<value>`

For example, the following command sets **vipnoarp** mode to **on** at boot:

`sysctl -w bigip.vipnoarp=1`

To turn vipnoarp mode **off** at boot, you would write the setting to `/etc/rc.sysctl` using the following command:

`sysctl -w bigip.vipnoarp=0`

`bigip.bonfire_mode`

Description

bigip.bonfire_mode=1 Sets the BIG-IP Controller to operate in Transparent Node mode, where it can perform load balancing on routers and router-like devices, such as transparent firewalls.

bigip.bonfire_mode=0 (Default) Transparent Node mode is **off**.

◆ Note

*With this version of the BIG-IP Controller, Transparent Node mode is no longer necessary. You do not need to set this variable. This variable only exists for backward compatibility. You can define a virtual server with address translation turned **on** or **off** at any time. For more information about address translation, see the BIG-IP Administrator Guide.*

bigip.bonfire_compatibility_mode

Description

bigip.bonfire_compatibility_mode=1 Turns off port translation on the BIG-IP Controller. This is useful if a node port is only being used to specify a service check port.

bigip.bonfire_compatibility_mode=0 (Default) Port translation is on.

◆ Note

*With this version of the BIG-IP Controller, Transparent Node mode is no longer necessary. You do not need to set this variable. This variable only exists for backward compatibility. You can define a virtual server with port translation turned **on** or **off** at any time. For more information about port translation, see the **BIG-IP Administrator Guide**.*

`bigip.fastest_max_idle_time`

Description

bigip.fastest_max_idle_time=<seconds> Sets the number of seconds a node can be left idle by the **fastest** load balancing mode. This forces the BIG-IP Controller to send fewer connections to a node that is responding slowly. This allows the BIG-IP Controller to periodically recalculate the response time of the slow node.

`bigip.forwarding_vip_overrides_default_snat`

bigip.forwarding_vip_overrides_default_snat=1 Setting this variable to **1** turns off SNAT for forwarding virtual servers. The result is that the default SNAT ignores a new connection with a destination that matches a forwarding virtual server. This causes outbound connections to use either a forwarding virtual server or the default SNAT depending on the destination of the packet that initiates the connection. The default setting for this variable is **1**.

bigip.forwarding_vip_overrides_default_snat=0 Setting this variable to **0** leaves SNAT on for forwarding virtual servers. The result is that both a SNAT and a virtual server connection to be created by an outbound packet initiating a connection to the destination specified by a forwarding virtual server.

`bigip.halt_reboot_timeout`

Description

bigip.halt_reboot_timeout=2 This value is the number of seconds the BIG-IP Controller can stop during boot up before the watchdog card hard reboots the controller. The default value for this setting is 2 seconds.

`bigip.improved_fastest`

bigip.improved_fastest=1 To use the improved Fastest, Observed, and Predictive load balancing modes, this variable must be set to **1**. The default value for this variable is **1**.

bigip.improved_fastest=0 If you do not want to use the improved fastest modes, set this variable to **0**. You might want to do this in a case where all connections to a node address are sequential and the application response is variable, slow, and unrelated to transport protocol response.

`bigip.max_sticky_entries`**Description**

bigip.max_sticky_entries=2048 This is the maximum number of sticky entries allowed to accumulate on the BIG-IP Controller when using destination address affinity (sticky persistence). When the maximum value is reached, the BIG-IP Controller stops accumulating sticky entries. The default value for this entry is 2048.

`bigip.memory_reboot_percent`

Description

bigip.memory_reboot_percent=0 The default value for this variable is **0**, which is disabled. The minimum value for this variable is 80, or 80 percent of the total physical memory on the controller. The value you type, 80 or higher, is the percentage of memory that is in use before the BIG-IP Controller automatically reboots.

`bigip.open_3dns_lockdown_ports`**Description**

bigip.open_3dns_lockdown_ports=0 (default) This variable is only required when running a 3DNS Controller. This variable is set to **0** on the BIG-IP Controller when the 3DNS Controller is not present in the network configuration. (See the ***3DNS Administrator Guide*** for more information.)

`bigip.open_telnet_port`

Description

bigip.open_telnet_port=1 Opens the Telnet port (23) to allow administrative Telnet connections (useful for an international BIG-IP Controller, or for a US controller that needs to communicate with international 3DNS Controllers).

bigip.open_telnet_port=0 Opens the Telnet port to allow administrative Telnet connections (useful for international BIG-IP Controllers).

`bigip.open_ftp_ports`

Description

bigip.open_ftp_ports=1 Opens the FTP ports (20 and 21) to allow administrative FTP connections (useful for international BIG-IP Controllers).

bigip.open_ftp_ports=0 The default setting for this variable is **0**. The FTP port does not allow administrative FTP connections.

`bigip.open_ssh_port`

Description

bigip.open_ssh_port=1 Opens the SSH port (22) to allow administrative connections (useful only for US BIG-IP Controllers).

bigip.open_ssh_port=0 The default setting for this variable is **0**. The SSH port does not allow administrative connections.

`bigip.open_rsh_ports`

Description

bigip.open_rsh_ports=1 Opens the RSH ports (512, 513, and 514) to allow RSH connections (useful for international BIG-IP Controllers, or on US controllers that need to communicate with international 3DNS Controllers).

bigip.open_rsh_ports=0 The default setting for this variable is **0**. The RSH port does not allow RSH connections.

`bigip.persist_map_proxies`

bigip.persist_map_proxies=1 The default setting for the map proxies for the persistence variable is **on**. The AOL proxy addresses are hard-coded in this release. 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 vip persist dump** to verify the mapping is working.

We recommend in addition to setting this **sysctl** variable, that you set a persist mask of 255.255.0.0 so that all the AOL addresses map to a common address. For example, Table 5.1 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 5.1 Address mapping of sample clients

bigip.persist_map_proxies=0 Set this variable to **0** to turn this variable **off**.

`bigip.persist_time_used_as_limit`

Description

bigip.persist_time_used_as_limit=0 (Default) Forces the persistent connection timer to reset on each packet for persistent sessions.

bigip.persist_time_used_as_limit=1 Resets timer only when the persistent connection is initiated.

◆ Note

For SSL persistence, the timer is always reset on each packet.

`bigip.persist_on_any_vip`

Description

bigip.persist_on_any_vip=1 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).

bigip.persist_on_any_vip=0 The default setting for this variable is **off**.

`bigip.persist_on_any_port_same_vip`

Description

bigip.persist_on_any_port_same_vip=1 All simple persistent 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).

bigip.persist_on_any_port_same_vip=0 The default setting for this variable is **off**.

`bigip.tcphps_mss_override`

Description

bigip.tcphps_mss_override=<1460 Allows you to decrease the default maximum segment size (MSS) from 1460 to a smaller value. This is the value announced to clients by the TCP server proxy on the BIG-IP Controller in the SYN/ACK packet.

bigip.tcphps_mss_override=0 (Default) The BIG-IP Controller requests the MSS from the node when negotiating connections on the node's behalf.

`bigip.verbose_log_level`

Description

bigip.verbose_log_level=0 Turns port denial logging **off**. No messages are logged.

bigip.verbose_log_level=1 Turns UDP port denial logging **on**. This logs UDP port denials to the BIG-IP Controller address.

bigip.verbose_log_level=2 Turns TCP port denial logging **on**. This logs TCP port denials to the BIG-IP Controller address.

bigip.verbose_log_level=4 Turns virtual UDP port denial logging **on**. This logs UDP port denials to the virtual server address.

bigip.verbose_log_level=8 Turns virtual TCP port denial logging **on**. This logs TCP port denials to the virtual server address.

bigip.verbose_log_level=15 Turns TCP and UDP port denial logging **on**. This logs TCP and UDP port denials to the virtual server address and the BIG-IP Controller address. Setting this variable to **15** turns on logging levels **1, 2, 4, and 8**.

`bigip.vipnoarp`

Description

bigip.vipnoarp=1 Prevents the BIG-IP Controller from issuing ARP requests when rebooted. This is useful for configurations that contain 1,000 or more virtual servers. This setting also prevents you from configuring virtual servers as IP addresses on the BIG-IP Controller external interface.

bigip.vipnoarp=0 The default setting for this variable is **0**. The BIG-IP Controller issues ARP requests on reboot.

`bigip.webadmin_port`

Description

bigip.webadmin_port=443 Specifies the port number used for administrative web access. The default port for web administration is port 443.

`net.inet.ip.forwarding`

Description

net.inet.ip.forwarding=1 Exposes node IP addresses on the internal network, allowing clients to connect directly to nodes, and also allows nodes to initiate connections with computers external to the BIG-IP Controller. Typically, this setting is used only on systems that cannot use NATs (for example, a network that uses CORBA or the NT Domain).

net.inet.ip.forwarding=0 (Default) IP forwarding is **off**.

`net.inet.ip.sourcecheck`

Description

net.inet.ip.sourcecheck=1 This setting enables the BIG-IP Controller to check the source IP address of incoming packets before it checks the packet for other information (for example, the virtual server).

Source checking tries to allocate a route back to the source of the packet, and if the route cannot be found, or if the route of the interface is on an interface different from the interface from which the packet was received, the packet is discarded. Each time a packet is discarded, the **bad source interface** counter is incremented.

net.inet.ip.sourcecheck=0 The default setting for this variable is IP source checking is **0** (off).

6

System Utilities

- `sod`
- `bigd`
- `big3d`

sod

sod [-help] [-tty00] [-tty01]

SOD Option	Description
-help	Prints help text.
-tty00	Use tty0 for fail-over monitoring.
-tty01	Use tty1 for fail-over monitoring.

Description

The switch-over daemon (**sod**) controls the BIG-IP Controller fail-over functions. It has a command line interface for some functions.

Command line usage

The **sod** daemon is used as a command line utility for some of its functions.

To display the online help for **sod**:

```
sod -help
```

Daemon start up options

The **sod** daemon is configured in **/etc/rc.local**. You can configure the **sod** daemon in two ways:

- ❖ Serial port(s) used for hardware fail-over cable connections
- ❖ Forced fail-over role (active or standby) at boot

◆ Note

*Every time you change your **sod** daemon configuration, you need to reboot the BIG-IP Controller.*

◆ Note

*In the examples in this section, **sod** starts the **bigd** daemon after **sod** startup completes, as has been the default configuration since BIG-IP version 1.8.2. This startup order is optimal, avoiding the possibility of creating certain suboptimal conditions at boot. For more information, see the F5 Networks Technical Note titled "Startup Sequence for Large Numbers of Nodes."*

Fail-over cable port configuration in sod startup

The **sod** daemon startup line in **/etc/rc.local** accepts two optional parameters: **-tty00** and **-tty01**. These parameters specify which of the two 9-pin serial ports (one of them may be a 25-pin serial port on older BIG-IP Controller models) is used as the fail-over cable connection. The default is **-tty01**. Use one (or none) of the **-ttyxx** options to configure a single fail-over cable. Use both options to configure two cables (redundant fail-over cables), as in the following example:

```
echo " sod (and bigd)."; /sbin/sod -tty00 -tty01 -- \
bigd ${bigdflags} 2> /dev/null
```

References to these fail-over cable connection ports in the **sod** startup line in **/etc/rc.local** are always made using the UNIX device name, while the hardware and BIOS settings for the ports use COM and serial port designations respectively..

BIOS	COM	UNIX
Port 2 (2f8 irq 3)	COM2	/dev/tty01
Port 1 (3f8 irq 4)	COM1	/dev/tty00

Table 6.1 9-pin serial port designations in BIOS, hardware, and UNIX operating system.

◆ Note

The 9-pin serial port labeled "Terminal" is COM2.

bigd

`bigd [-d filename] [-n] [-s] [-v] [-V]`

Description

This daemon monitors services and nodes for the BIG-IP Controller. The **bigd** daemon provides service check functions for simple (node ping), extended content verification, and extended application verification service checks. Usage is supported for cases where the check port for a node is not the same as the node port. Table 6.2 contains the options available for **bigd**.

Option	Description
-d filename	Check syntax of the specified configuration file, and then exit. This option cannot be used in conjunction with any other option.
-n	Do not ping nodes.
-s	TCP node ping (default is ICMP)
-v	Print version number.
-V	Print verbose output to message logs.

*Table 6.2 The **bigd** options*

Files

- /etc/bigip.conf
- /etc/rc.local
- /etc/bigd.conf
- /var/log/bigd
- /var/log/messages

Configuring bigd

The configuration files in Table 6.3 contain configuration information for **bigd**.

File	Description
/etc/rc.local	Starts bigd with the options specified.
/etc/bigip.conf	Contains configuration information for timeout_svc and tping_svc .
/etc/bigd.conf	Contains configuration information for ECV and EAV service checks.

Table 6.3 bigd configuration files

Starting bigd

The standard way to start **bigd** is by configuring the **sod** startup line in **/etc/rc.local**:

```
echo " sod (and bigd)."; /sbin/sod -- bigd ${bigdflags} /
2> /dev/null
```

This syntax starts **bigd** after the boot configuration in **/etc/bigip.conf** has been loaded and started. This is the optimal sequence for startup if you use ping aliases. If **bigd** is started before **sod** when ping aliases are defined, node pinging starts before ping aliases have been loaded.

You can also start and restart **bigd** on the command line with options:

bigd

This is the best way to restart **bigd** if you make changes to the **/etc/bigd.conf** file. This method stops any existing **bigd** processes and restarts the daemon using the configuration in **/etc/rc.local** and **/etc/bigd.conf**.

Setting the node ping parameters used by bigd

Node ping uses the **timeout_node** and **tping_node** parameters (set in **/etc/bigip.conf**) to set the length of time between pings and the length of time to wait for a ping response before timeout.

Setting service check parameters used by bigd

Simple and extended service checks use the **timeout_svc** and **tping_svc** parameters (set in **/etc/bigip.conf**) to set the length of time between checks and the length of time to wait for a check response before timeout.

Extended service checks also use data from the **/etc/bigd.conf** file. There are seven ways to use Extended Content Verification and Extended Application Verification to check status. The different checks are listed in Table 6.4:

Keyword in bigd.conf	Usage
ssl	ECV
active	ECV, ECV on nodes w/wildcard ports
reverse	ECV fails check if string is found
external	EAV
gateway	router ping
transparent	transparent node mode
simple	wildcard ports simple check

Table 6.4 Keywords in **/etc/bigd.conf**

Service checking for wildcard servers and ports

The **simple** keyword is necessary to perform simple service checks on nodes with wildcard ports. Use the following syntax to set a check on a node where the check port is not the node port:

```
simple [<node addr>:]<node port> <check port>
```

For example, if a wildcard server is defined with a non-wildcard port:

```
bigpipe vip 0.0.0.0:0 define n1:0
```

To configure the check on it, use the **simple** keyword to designate the wildcard **<server:><port>** and **<check port>**:

```
simple n1:0 80
```

Use the following variation on the **active** keyword syntax to configure ECV on nodes with wildcard ports:

```
active <node addr>:0 <check port> [<send string> [<regexp>]]
```

This syntax is only allowed when the node port is 0. When the node port is 0, this is the only syntax that is allowed.

To support EAV on nodes with wildcard ports, an additional variation on the **external** command in the **/etc/bigd.conf** file is added:

```
external <node addr>:0 <check port> [<program name> [<arguments>]]
```

This syntax is only allowed when the node port is 0. When the node port is 0, this is the only syntax that is allowed.

When this syntax is used, the calling convention for the external pinger is changed to:

```
<program name> <node addr> <check port> <arguments>
```

Service checking through transparent nodes

The **/etc/bigd.conf** file supports ECV for transparent nodes. This is done by checking a destination through the particular transparent node you want to check.

The following syntax is supported in the **/etc/bigd.conf** file for ECV through a transparent node:

```
transparent <node_ip>:<port> <site_ip>:<port> [<send_string>
[<recv_expr>]]
```

The **bigdnode** program uses this syntax to make the appropriate socket option settings for the ECV check.

The following example shows how to set up an ECV check through a transparent node. The following virtual servers are defined for this example:

```
bigpipe vip 0.0.0.0:80 define p1:80 p2:80
```

```
bigpipe vip 0.0.0.0:0  define fw1:0 fw2:0
```

Configure the `/etc/bigd.conf` as shown:

```
transparent p1:80 www.yahoo.com:80 'GET /' 'Yahoo'
transparent p2:80 www.yahoo.com:80 'GET /' 'Yahoo'
transparent fw1:0 www.yahoo.com:80 'GET /' 'Yahoo'
transparent fw2:0 www.yahoo.com:80 'GET /' 'Yahoo'
```

◆ Tip

*Note that wildcard ports in virtual server definitions no longer require a defined service check port with the node if you do not want port translation. Instead, **0** is used to indicate that port translation should not take place.*

In this example, node **p1:80** is tested by getting the web page **http://www.yahoo.com/**. The web request is routed through **p1**. The transparent node **fw2:0** is tested by getting the same web page (still on port 80), routed through **fw1**.

big3d

The **big3d** daemon answers 3DNS Controller system queries. 3DNS uses **big3d** to collect information about the network path between the BIG-IP Controller and the client requesting a connection. The **big3d** utilities calculate performance data, and return the data to the requesting 3DNS Controller. The 3DNS Controller uses the path information for its own dynamic load balancing.

You can start or stop the **big3d** process without affecting any other processes on the BIG-IP Controller.

If you no longer want to run the **big3d** process on the BIG-IP Controller, stop the process and remove the corresponding start line from **/etc/rc.local**. The only reason you might want to do this is if your installation once used 3DNS but no longer uses it.

WARNING

*When the **big3d** agent on the BIG-IP Controller is stopped, the 3DNS Controller can no longer provide dynamic load balancing for the virtual servers that run on the BIG-IP Controller. This may affect pool definitions on the 3DNS Controller.*

big3d hardware and software compatibility

The version of the **big3d** daemon, the BIG-IP Controller, and the 3DNS Controller that sends requests to it must be compatible. Any time you upgrade the BIG-IP Controller or the 3DNS Controller, check to make sure the versions of **big3d** are compatible.

Installing big3d

Run the **big3d** install script on the 3DNS Controller to install the correct version of **big3d** on the BIG-IP Controller, and add the auto start info to the BIG-IP Controller **/etc/rc.local** file. This sets up the proper fail-over configuration, so that if the BIG-IP Controller is rebooted or fails over, **big3d** starts automatically on the standby BIG-IP Controller.

Services and port configurations

Communication between the 3DNS Controller and **big3d** daemon on the BIG-IP Controller depends on the proper management of specific ports.

Outbound iQuery requests

The port used by the iQuery protocol to pass queries and results between the 3DNS Controller and **big3d** is now registered with the IANA as port 4353.

In previous versions of the BIG-IP Controller, outbound iQuery traffic service used port 245. Current releases of BIG-IP and 3DNS Controller software enable both of these ports by default, and the **big3d** daemon on the BIG-IP Controller detects iQuery requests on either port.

Firewall ports

The firewall ports 245 and/or 4353 must allow traffic between the BIG-IP Controller and the 3DNS Controller.

WARNING

Firewalls between the 3DNS and BIG-IP Controllers must allow traffic on one or both of these ports. If the firewall rejects iQuery traffic, then 3DNS cannot provide dynamic load balancing for the virtual servers that run on the BIG-IP Controller, which may affect pool definitions on the 3DNS.

7

BIG/db Configuration Keys

- Supported BIG/db configuration keys

Supported BIG/db configuration keys

Currently, the following configuration options are supported by BIG/db:

- ❖ Fail-over
- ❖ State mirroring
- ❖ Gateway pingers
- ❖ Configuration synchronization
- ❖ Interface related settings

The BIG/db keys for each of these features are described in the following series of tables.

Using the Failover and Cluster keys

The switch-over daemon (**sod**) uses all **.Failover** keys (Table 7.1). The **sod** and the mirroring daemon (**sfd**) use all **.Cluster** keys. If you change one of these values, you must update the BIG-IP Controller configuration with the **bigpipe failover init** command. This command forces **sod** to reread the BIG/db database and use the new values. To run this command, type the following command:

```
bigpipe failover init
```

For more information about **sod**, see *sod*, on page 6-1.

Fail-Over Key Name	Description
Common.Bigip.Failover.AwaitPeerAliveDelay	Delay in seconds before testing whether peer is active. The default value is 2 .
Common.Bigip.Failover.AwaitPeerDeadDelay	Delay in seconds before testing whether the peer has failed. The default value is 1 .
Local.Bigip.Failover.UnitId	This key is required. Each controller must have a unique unit ID of 1 or 2 in the event that network communication is not possible with its peer.
Common.Bigip.Failover.ActiveMode	Use active-active mode if set to 1 . By default this is off and active-standby mode is used.

Table 7.1 The BIG/db keys that store fail-over information

Fail-Over Key Name	Description
Common.Bigip.Failover.ManFailBack	If using active-active mode, the fail-over mechanism waits until receiving a command before surrendering resources to a rebooted machine.
Common.Bigip.Failover.NoSyncTime	By default, one BIG-IP Controller synchronizes its time with the other. Set this key to 1 to turn off the time synchronization feature.
Common.Bigip.Failover.DbgFile	File into which fail-over (/sbin/sod) logs debug information.
Common.Bigip.Failover.PrintPeerState	The default value for this key is 0 . Fail-over daemon (/sbin/sod) writes the state of its connection to its peer, hardwire and/or network. This info is written to the fail-over daemon's debug log file.
Common.Sys.Failover.Network	Use the network (via the sfd) as a backup to, or instead of, the serial line for fail-over if this value is 1 . By default this feature is off.
Common.Bigip.Cluster.ActiveKeepAliveSec	The default value for this key is 0 . An active unit sends a heartbeat message to its peer with this frequency. Default is 1 second.
Common.Bigip.Cluster.StandbyTimeoutSec	Consider the peer BIG-IP failed if no message is received within the timeout period. Used by network fail-over. Default is 3 seconds.

Table 7.1 The BIG/db keys that store fail-over information

Using StateMirror keys

The mirroring daemon (**sfd**) uses all **StateMirror** keys (Table 7.2). If you change one of these values you must reboot the BIG-IP Controller to load the new configuration and restart the **sfd**.

State Mirroring Key Name	Description
Common.Bigip.StateMirror.DbgFile	File into which debug information is written, required if debug level is specified (below).
Common.Bigip.StateMirror.DbgLevel	The debug level is composed of the following options: 1 - log reads and writes 2 - log connection attempts and results 4 - log state changes 8 - open log files in append mode, the default is to truncate the files when opening. The debug level is zero by default.
Common.Bigip.StateMirror.NoGC	By default, state mirroring causes mirrored data structures to be deleted when it receives a new connection. This key is brought up to date by the controller's peer. This can cause a delay if the system is absolutely loaded. Turning off the GC is provided as an option.
Common.Bigip.StateMirror.ActiveFile	Enables writing of data from the active unit's kernel into the ActiveFile file. This data file can be read with the sfsread program.
Common.Bigip.StateMirror.StandbyFile	Enables writing of standby data into the StandBy file. This data file can be read with the sfsread program.
Common.Bigip.StateMirror.PeerListenPort	Port on which the BIG-IP Controller listens for connections from the active unit. Default is 1028 .
Local.Bigip.StateMirror.Ipaddr	IP address of this BIG-IP Controller.
Local.Bigip.StateMirror.PeerIpaddr	IP address of this BIG-IP Controller's peer controller. A value is required.

Table 7.2 The BIG/db keys that store state mirroring information

Gateway Pinger Key Name	Description
Local.Bigip.GatewayPinger.Ipaddr	IP address or host name of the gateway router for the BIG-IP Controller
Local.Bigip.GatewayPinger.Pinginterval	Ping interval of this BIG-IP Controller gateway pinger
Local.Bigip.GatewayPinger.Timeout	Timeout of the BIG-IP Controller gateway pinger

Table 7.3 *The BIG/db keys that store gateway pinger information*

Using configuration synchronization and Interface keys

The First-Time Boot utility sets the configuration synchronization and **Interface** keys (Table 7.4 and Table 7.5). You should not change these keys manually with **bigdba**.

Configuration Synchronization Key Name	Description
Common.Bigip.ConfigSyncKey	Used for synchronizing the configuration of two BIG-IP Controllers.
Local.Bigip.FTB.DefaultRoute	First-Time Boot utility Fields. These fields are populated by the First Time Boot utility.
Local.Bigip.FTB.FailoverIp	
Local.Bigip.FTB.FTP	These represent a small fraction of the actual fields produced, as the rest are dynamic and dependent upon First-Time Boot utility user input such as host names and interface identifiers.
Local.Bigip.FTB.FTP.allow	
Local.Bigip.FTB.FTP.support	
Local.Bigip.FTB.HostName = bigip1	
Local.Bigip.FTB.HostNumber = 1	
Local.Bigip.FTB.Interfaces	
Local.Bigip.FTB.IsRedundant	
Local.Bigip.FTB.License	
Local.Bigip.FTB.Locality = domestic	
Local.Bigip.FTB.Product = BIG-IP®	
Local.Bigip.FTB.RootSet	
Local.Bigip.FTB.RSH	
Local.Bigip.FTB.RSH.allow	
Local.Bigip.FTB.RSH.support	
Local.Bigip.FTB.SupportEmail	
Local.Bigip.FTB.SupportFax	
Local.Bigip.FTB.SupportPhone	
Local.Bigip.FTB.Telnet	
Local.Bigip.FTB.Telnet.allow	
Local.Bigip.FTB.Telnet.support = yes	
Local.Bigip.FTB.TimeZone = US/Pacific	
Local.Bigip.FTB.Vendor	
Local.Bigip.FTB.VendorCommonName	
Local.Bigip.FTB.VendorSupportIp	
Local.Bigip.FTB.Version = "BIG/ip 3.2"	
Local.Bigip.FTB.WebServer.Cert.City = Seattle	
Local.Bigip.FTB.WebServer.Cert.Company = "Company"	
Local.Bigip.FTB.WebServer.Cert.Country = US	
Local.Bigip.FTB.WebServer.Cert.Division = undef	
Local.Bigip.FTB.WebServer.Cert.State = WA	
Local.Bigip.FTB.WebServer.Prefix = http	
Local.Bigip.FTB.WebServer.ReplaceKeys = no	

Table 7.4 The BIG/db keys that store configuration synchronization information.

Interface Key Name	Description
Local.Bigip.Interface.*.Name Common.Bigip.Interface.*.Name	The interface name is used as the key for querying or modifying its attributes. The key is common by default, but can be local to accommodate redundant BIG-IP Controllers with different interface names. However, the interface names must all be local or all be common.
Common.Bigip.Interface.*.AddrList.*.Address Common.Bigip.Interface.*.AddrList.*.Netmask Common.Bigip.Interface.*.AddrList.*.Broadcast Common.Bigip.Interface.*.AddrList.*.Type	Attributes of an interface's addresses. Currently, the BIG/db contains only shared IP addresses. Type is Shared or True . Shared is associated with an IP alias, True is associated with an IP address.
Common.Bigip.Interface.*.AddrList.*.UnitId Common.Bigip.Interface.*.AddrList.*.VlanTag	The unit number with which the shared IP alias is associated.

Table 7.5 *The BIG/db keys that store interface information*



Index

- /etc/bigconf.conf file 3-1
- /etc/bigd.conf file 2-7, 3-1, 6-4
- /etc/bigip.conf file 2-7, 2-50, 2-54, 3-1, 6-4
- /etc/bigip.license file 3-1
- /etc/hosts file 3-1
- /etc/hosts.allow file 2-7, 3-1
- /etc/hosts.deny file 2-7
- /etc/ipfw.conf file 2-7, 3-1
- /etc/ipfwrate.conf file 2-7, 3-1
- /etc/irs.conf file 3-1
- /etc/login.conf file 3-1
- /etc/netstart file 3-1
- /etc/rateclass.conf file 2-7, 3-1
- /etc/rc file 3-1
- /etc/rc.local file 3-1, 6-4
- /etc/rc.sysctl file 3-1, 5-1–5-21
- /etc/services file 2-42
- /etc/snmpd.conf file 2-7, 3-1
- /etc/sshd_config file 3-1
- /etc/syslog.conf file 3-1
- /etc/wideip.conf file 3-1
- /usr/contrib/bin/ssh-askpass file 3-2
- /var/f5/bigdb/user.db file 3-2
- /var/f5/httpd/basicauth/users file 3-2
- /var/f5/httpd/conf/cert.conf file 3-2
- /var/f5/httpd/conf/httpd.conf file 3-2
- /var/f5/httpd/ssl/lib/sslkey.cnf file 3-2
- /var/f5/httpd/ssl/lib/sslkey.conf file 3-2
- /VENDOR file 3-1
- /VERSION file 3-2
- ? command 2-4

A

- alias command 2-5–2-6
- ARP requests 2-78, 5-22

B

- backward compatibility 5-3, 5-4
- backward-compatible commands 2-84
- BIG/db keys
 - configuration synchronization 7-5
 - fail-over 7-1
 - gateway ping 7-4
 - interface 7-6
 - state mirroring 7-3
- BIG/ip controllers
 - configuration utilities 4-1–4-8
 - HA 4-3
- BIG/pipe commands 2-1–2-84
- big3d utility 6-8
- bigd utility 6-3
- bigip.bonfire_compatibility_mode variable 5-4
- bigip.bonfire_mode variable 5-3
- bigip.fastest_max_idle_time variable 5-5
- bigip.forwarding_vip_overrides_default_snat 5-6
- bigip.halt_reboot_timeout variable 5-7
- bigip.improved_fastest 5-8
- bigip.max_sticky_entries variable 5-9
- bigip.memory_reboot_percent variable 5-10
- bigip.open_3dns_lockdown_ports variable 5-11
- bigip.open_ftp_ports variable 5-13
- bigip.open_rsh_ports variable 5-15
- bigip.open_ssh_port variable 5-14
- bigip.open_telnet_port variable 5-12

Index

bigip.persist_on_any_port_same_vip
variable 5-19
bigip.persist_on_any_vip variable 5-18
bigip.persist_time_used_as_limit
variable 5-17
bigip.tcpmps_mss_override variable 5-20
bigip.verbose_log_level variable 5-21
bigip.vipnoarp variable 5-22
bigip.webadmin_port variable 5-23
bitmask 2-25
broadcast 2-78

C

config utility 4-2
config_failover utility 4-3
config_rshd utility 4-7
config_sshd utility 4-6
config_synckey utility 4-8
config_telnetd utility 4-5
configsyc command 2-7
Configuration 4-3
configuration files 3-1–3-2
configuration keys
 BIG/db 7-1–7-6
configuration settings 2-10, 2-54
configuration synchronization key names 7-5
configuration utilities 4-1–4-8
configurations
 clearing memory 2-47
 synchronizing 2-7
 testing files 2-9
 validating 2-9
conn command 2-8
connection limits
 for node addresses 2-30
 for nodes 2-30

 for ports 2-42
 for SNAT addresses 2-57
 for virtual servers 2-79
connection timeout 2-71
connections
 client 2-8
 FTP 5-13
 persistence 2-33
 Telnet 5-12
 virtual server 2-78

D

-d command 2-9

E

EAV. See service checks
ECV. See service checks
extended application verification. See service checks
extended content verification. See service checks
external interfaces 2-14, 2-25, 2-78
external network. See external interfaces

F

-f command 2-10
F5 Configuration utility 4-3, 5-1
failover command 2-11
First-Time Boot utility 4-2

G

gateway command 2-12
gateway pinger key names 7-4

H

-h command 2-13
 -help command 2-13
 help command 2-4
 httpd server 4-4

I

interface command 2-14–2-19
 interface key names 7-6
 internal interfaces 2-14
 IP addresses 2-20, 2-25
 IP forwarding 5-24
 ipalias command 2-20

K

key synchronization 4-8

L

-l command 2-21
 lb command 2-22
 load balancing modes
 Priority 2-48
 Ratio 2-48
 setting 2-22
 load balancing pools 2-34, 2-77
 load balancing rules 2-77
 logging 5-21

M

MAC addresses 2-18
 maint command 2-23
 media access control. See MAC addresses
 mirroring. See persistence 2-81

N

nat command 2-25–2-27
 net.inet.ip.forwarding variable 5-24
 net.inet.ip.sourcecheck variable 5-9, 5-25
 netmask 2-25, 2-78
 network address translation (NAT). See nat command
 network interface cards. See NIC
 NIC 2-14–2-19
 node addresses
 connection limits 2-30
 ratio weights 2-48
 node aliases 2-5
 node command 2-29–2-32
 node ping 2-5, 2-62–2-63, 2-66, 6-5
 nodes
 connection limits 2-30
 connections 2-25
 service checks 2-64–2-67
 virtual server mappings 2-76

P

persist command 2-33
 persistence 5-17, 5-18, 5-19
 for virtual ports 2-33
 See also HTTP cookie persistence
 ping. See node ping
 pingnode. See node ping
 pool command 2-34–2-40
 port command 2-42–2-43
 ports 6-9
 defined 2-42
 displaying numerically 2-28
 idle connection timeout 2-69
 persistence 2-33
 RSH 5-15
 SSH 5-14

Index

- translation properties 2-80
- udp 2-71
- Priority mode 2-48

R

- r command 2-47
- ratio command 2-48–2-49
- Ratio mode 2-48
- ratio weights 2-48, 2-49
- rc.sysctl file 2-7
- reconfig-httpd utility 4-4
- redundant systems
 - configuring gateway fail-safe 2-12
 - configuring hop pools 2-80
 - displaying unit number 2-73
 - failover 2-11
 - mirroring 2-57
 - sharing MAC addresses 2-19
 - synchronizing 2-7
- remote shell. See RSH
- RSH 5-15
- rule command 2-50–2-52
- rules
 - defined 2-50
 - listing of 2-52
 - load balancing 2-77
 - load balancing pools 2-50

S

- secure network address translation (SNAT). See snat command
- secure shell 4-6, 5-14
- service checks 2-64–2-65, 2-67
 - EAV 2-64, 2-67, 6-5–6-7
 - ECV 2-64, 2-67, 6-5–6-7
 - simple 2-64, 2-67, 6-5–6-6
 - timeouts 2-65
 - tping_svc 2-67

- services
 - FTP configuration 4-5
 - telnet configuration 4-5
- snat command 2-55–2-58
- sod utility 6-1
- source checking 5-25
- SSH 5-14
- state mirroring key names 7-3
- sticky entries 5-9
- summary command 2-59–2-61
- switch-over daemon 6-1
- synchronization 4-8
- sysctl command 5-2
- system control variables 5-1–5-21
- system utilities
 - big3d 6-8–6-9
 - bigd 6-3–6-7
 - sod 6-1–6-2

T

- text editor 5-1
- timeout_node command 2-62–2-63
- timeout_svc command 2-64–2-65
- tping_node command 2-66
- tping_svc command 2-67–2-68
- translation properties 2-80
- transparent node mode 5-3, 5-4
- treaper command 2-69–2-70

U

- udp command 2-71–2-72
- unit command 2-73
- usage statistics 2-59–2-61
- utilities
 - BIG/pipe commands 2-1–2-84

- configuration 4-1–4-8
- F5 Configuration 4-3
- First-Time Boot 4-2
- system 6-1–6-9

V

- v command 2-74
- vip command 2-76–2-83
- virtual addresses
 - defining a netmask 2-78
 - displaying information 2-82
 - statistics 2-82
 - translation properties 2-80
- virtual ports. See ports
- virtual server mappings 2-76
- virtual servers 2-76–2-83
 - configuring hop pools 2-80
 - configuring load balancing pools 2-77
 - displaying unit number 2-73
 - enabling 2-81
 - ports 2-76
 - rules 2-50, 2-77
- VLAN communication 2-19

W

- web administration 5-23