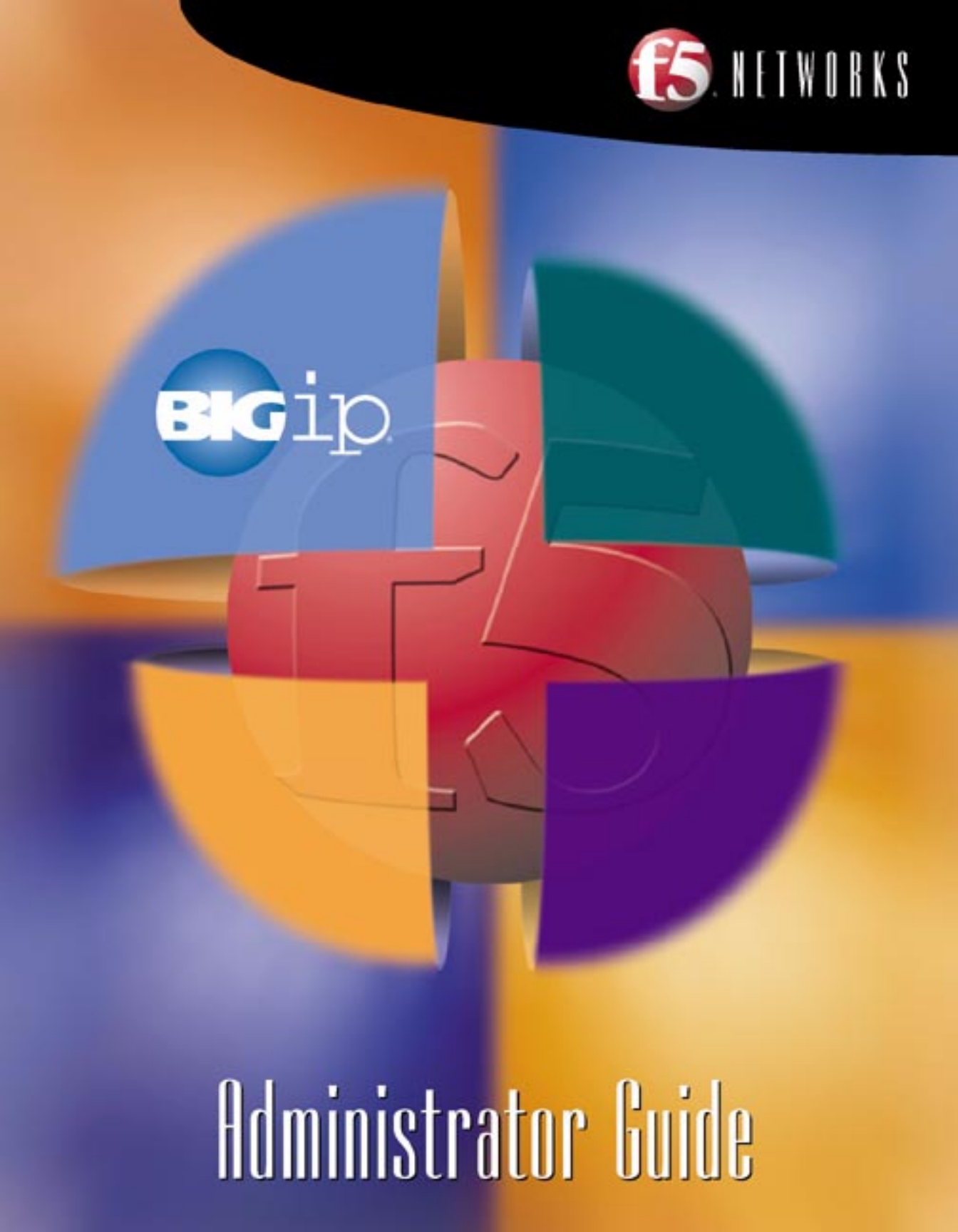
f5 NETWORKS



Administrator Guide

BIG/ip™ Controller Administrator Guide

version 3.0

Service and Support Information

Product Version

This manual applies to version 3.0 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© 2000 by
F5 Networks, Inc.
Seattle, Washington
All rights reserved. Printed in U.S.A.

Trademarks

F5, BIG/ip, and 3DNS 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, if originally sold within the United States, 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.

Canada 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/).

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 (for example, BIG/ip V1.5 to BIG/ipV1.6). 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 (for example, BIG/ip V1.x to BIG/ip V2.0). 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 Administrator Guide

Welcome to the BIG/ip Controller Administrator Guide 1-1
BIG/ip Controller specifications 1-1
Internet protocol and network management support 1-2
Security features 1-3
Configuration scalability 1-4
Configuration and monitoring tools 1-4
Load balancing options 1-5
IP packet filtering, rate classes, and rate filters 1-6
Configurable persistence for e-commerce and dynamic content sites . . 1-7
BIG/ip Controller platform options 1-8
Finding help and technical support resources 1-8
What's new in version 3.0 1-9
Active-active redundant controllers 1-10
Intelligent traffic control (ITC) 1-10
Versatile interfaces 1-11
Per-connection routing 1-12
Secure forwarding virtual servers 1-13
Transparent virtual servers 1-13
Virtual server port translation 1-13
Reset connections on service down 1-14
Cookie persistence hash mode 1-14
Enhancements to configuration and monitoring tools 1-14

Chapter 2

Working with Special Features

Introducing special features 2-1
Using specialized load balancing modes 2-1
Understanding individual load balancing modes 2-2
Setting the global load balancing mode 2-4
Controlling network access and traffic flow with filters 2-7
IP filters 2-7
Rate filters and rate classes 2-9
Working with more than two interface cards 2-11
Configuring additional interfaces with the First-Time Boot utility . . . 2-11
Specifying an interface for a virtual address 2-13
Specifying an interface for a NAT address 2-14
Specifying an interface for a SNAT address 2-15
Routing with multiple NICs 2-16

Example configurations for more than two NICs 2-16
Configuration example: One gateway 2-17
Configuration example: Two gateways 2-17
Optimizing large configurations 2-18
Reducing ARP traffic on the external network 2-18
Reducing the number of node pings and service checks issued by the BIG/ip Controller 2-21
Using the versatile interface configuration options 2-24
Destination route and translation processing 2-26
Source translation processing 2-26
Interface security 2-28
Using advanced virtual server options 2-29
Using per-connection routing 2-29
Configuring forwarding virtual servers 2-31
Configuring transparent virtual servers 2-32
Using virtual server port translation 2-33
Resetting connections on service down 2-34

Chapter 3

Working with Intelligent Traffic Control

Introducing Intelligent Traffic Control (ITC) 3-1
More flexible load balancing using pools and members 3-1
Load balancing pools 3-2
Selecting a load balancing pool using a rule (BIG/ip Controller HA versions only) 3-7
Configuring rules from the command line or in the F5 Configuration utility 3-13
Configuring virtual servers that reference rules from the command line or in the F5 Configuration utility 3-13
A comparison of load balancing configuration from previous versions of the BIG/ip Controller 3-16

Chapter 4

Working with Advanced Service Check Options

Introducing advanced service check options 4-1
Setting up ECV service checks for transparent nodes 4-1
Introducing EAV service checks 4-4
Setting up custom EAV service checks 4-5
Using the EAV pingers bundled with the BIG/ip Controller 4-9

Chapter 5

Working with Advanced Persistence Options

Introducing advanced persistence options 5-1
Using HTTP cookie persistence 5-1
Using destination address affinity (sticky persistence) 5-5
Using persist mask on a virtual server 5-7
Maintaining persistence across virtual servers that use the same virtual addresses 5-8
Maintaining persistence across all virtual servers 5-9

Chapter 6

Working with Advanced Redundant System Features

Introducing advanced redundant system options 6-1
Mirroring connection and persistence information 6-1
Using gateway fail-safe 6-4
Using network-based fail-over 6-7
Setting a specific BIG/ip Controller to be the preferred active unit 6-8
Active-active redundant controllers 6-9
Configuring an active-active system 6-10
Active-active system fail-over 6-16
Additional active-active BIG/db configuration parameters 6-18
New active-active bigpipe commands 6-20
Running mixed versions of BIG/ip Controller software in active-active mode 6-20
Returning an active-active installation to active/standby mode 6-21

Chapter 7

Using Advanced Network Configurations

Introducing advanced network configurations 7-1
nPath routing 7-1
Defining a virtual server with address translation disabled 7-2
Setting the route through the BIG/ip Controller 7-4
Setting the idle connection time-out 7-5
Per-connection routing 7-7
ISP load balancing 7-9
Configuring interfaces for the additional internet connection 7-10
Configuring virtual servers for an additional internet connection 7-10
Firewall load balancing 7-12
Configuring interfaces for the firewall sandwich 7-13
Configuring virtual servers for a firewall sandwich 7-14

VPN load balancing 7-17
Configuring interfaces for VPN load balancing 7-18
Configuring virtual servers for VPN load balancing 7-18
VPN and router load balancing 7-20
Configuring interfaces for VPN load balancing 7-21
Configuring virtual servers for VPN and router load balancing 7-22
SNAT and virtual servers combined 7-25
One IP network topology with one interface 7-28
One IP network topology with two interfaces 7-30
Setting up 802.1q VLAN trunk mode 7-33
Adding VLAN tag definitions to /etc/netstart 7-33
Adding VLAN tag definitions to BIG/db 7-34
Configuring multiple VLANs on one interface 7-34
To enable or disable VLAN tags on the command line 7-35
Using ifconfig to add another VLAN 7-35
Using netstat to view VLAN tags 7-35
Disabling and enabling VLAN tags using the F5 Configuration utility 7-36

Chapter 8

Monitoring and Administration

Monitoring and administration utilities provided on the BIG/ip Controller 8-1
Using the BIG/pipe command utility as a monitoring tool 8-2
Monitoring the BIG/ip Controller 8-2
Monitoring virtual servers, virtual addresses, and services 8-10
Monitoring nodes and node addresses 8-12
Monitoring NATs 8-13
Monitoring SNATs 8-13
Working with the BIG/stat utility 8-14
Working with the BIG/top utility 8-15
Working with the Syslog utility 8-16
Removing and returning items to service 8-17
Removing the BIG/ip Controller from service 8-18
Removing individual virtual servers, virtual addresses, and ports from service 8-19
Removing individual nodes and node addresses from service 8-20
Viewing the currently defined virtual servers and nodes 8-20
Viewing system statistics and log files 8-21
Viewing system statistics 8-21
Viewing log files 8-22
Printing the connection table 8-22
Changing passwords for the BIG/ip Controller 8-22

Changing the BIG/ip Controller password 8-23
Changing passwords and adding new user IDs for the BIG/ip web server 8-23
Working with the BIG/db database 8-24
Using bigdba 8-25

Chapter 9

Configuring SNMP

Working with SNMP on the BIG/ip Controller 9-1
Configuring SNMP on the BIG/ip Controller 9-1
Downloading the MIBs 9-2
Understanding configuration file requirements 9-3
Configuring options for the checktrap script 9-9

Glossary

Index

1

Introduction to the BIG/ip Controller Administrator Guide

- **Welcome to the BIG/ip Controller Administrator Guide**
- **BIG/ip Controller specifications**
- **Finding help and technical support resources**
- **What's new in version 3.0**

Welcome to the BIG/ip Controller Administrator Guide

Welcome to the *BIG/ip® Controller Administrator Guide*. This guide describes the advanced features included in the BIG/ip Controller. The Administrator guide also includes the software specifications for the BIG/ip Controller platform and reviews some sample configurations that can help you in planning your own configuration. This book is a 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 configuration utilities, and system utilities.

BIG/ip Controller specifications

The BIG/ip Controller is a network appliance that manages and balances traffic for networking equipment such as web servers, cache servers, routers, firewalls, and proxy servers. A variety of useful features meets the special needs of e-commerce sites, Internet service providers, and managers of large intranets. The system is highly configurable, and its web-based and command line configuration utilities allow for easy system set up and monitoring.

Adding a BIG/ip Controller to your network ensures that your network remains reliable. The BIG/ip Controller continually monitors the servers and other equipment it manages, and never attempts to send connections to servers that are down or too busy to handle the connection. The BIG/ip Controller uses a variety of methods to monitor equipment, from simple pings to more

advanced methods, such as Extended Content Verification that verifies whether a server returns specific site content. The BIG/ip Controller also offers several layers of redundancy that ensure its own reliability.

Internet protocol and network management support

The BIG/ip platform supports both TCP and UDP protocols, and also supports popular network services including:

- HTTP
- SSL
- FTP (Active and Passive)
- SMTP
- NNTP
- POP
- DNS
- IMAP
- Real Audio/TCP
- Telnet

The BIG/ip Controller supports administrative protocols, such as Simple Network Management Protocol (SNMP) and Simple Mail Transfer Protocol (SMTP) (outbound only), for performance monitoring and notification of system events. The BIG/ip Controller's SNMP agent allows you to monitor status and current traffic flow using popular network management tools, including the F5 Configuration utility. The SNMP agent provides useful data such as packets in and out per second, and current connections being handled for each virtual server. You may also want to take advantage of Telnet, FTP, and the F-Secure SSH client (distributed only in the US). The F-Secure SSH client provides a secure UNIX shell connection to the BIG/ip Controller from a remote workstation.

Security features

The BIG/ip Controller offers a variety of security features that protect both the controller itself, and the network equipment that it manages. Each of the following features can help prevent potentially hostile attacks on your site or equipment.

- **IP address protection**
  On its external network, the BIG/ip Controller does not expose the IP addresses of the servers that it manages unless you specifically configure it to do so. Instead, it offers firewall capabilities, translating addresses when servers connect to other hosts on the external network. You can set up either standard Network Address Translations (NATs) that allow both incoming and outgoing traffic, or you can set up Secure Network Address Translations (SNATs) that allow only outgoing connections.
- **Port lockdown**
  The BIG/ip Controller prevents clients from connecting to any port which you have not specifically opened for network traffic. This feature helps prevent a common attack where users try to gain access to the machine using one of the many ephemeral ports that do not host a well-known service.
- **Controlled administrative connections**
  The BIG/ip Controller allows you to make direct administrative connections to the servers it manages, but it prevents direct connections to those servers by random clients, based on their IP address.
- **IP address filtering**
  The IP filtering features allow you to specifically accept or deny connections received from particular IP addresses or ranges of IP addresses.
- **Termination of inactive connections**
  The BIG/ip Controller automatically terminates connections that remain inactive for a period of time you specify, which prevents common denial of service attacks.

In addition to these features, BIG/ip Controllers distributed in the US support encrypted administrative connections using F-Secure SSH for shell connections, and SSL protocol for connections to the web-based configuration utility.

Configuration scalability

The BIG/ip Controller is a highly scalable and versatile solution. You can actually configure a single BIG/ip Controller to manage thousands of virtual servers, though most common configurations are significantly smaller. The number of servers, firewalls, or routers that a single BIG/ip Controller can load balance is limited only by the capacity of your network media, such as Ethernet. The BIG/ip Controller supports a variety of media options, including Fast Ethernet, Gigabit Ethernet, and FDDI. The maximum number of concurrent connections that a BIG/ip Controller can manage is determined by the amount of RAM in your particular BIG/ip Controller hardware configuration.

Configuration and monitoring tools

The BIG/ip platform provides the following web-based and command line administrative tools that make for easy setup and configuration.

The First-Time Boot utility

The First-Time Boot utility is a wizard that walks you through the initial system set up. The utility helps you quickly define basic system settings, such as a root password and the IP addresses for the interfaces that connect the BIG/ip Controller to the network. The First-Time Boot utility also helps you configure access to the BIG/ip web server, which hosts the web-based F5 Configuration utility.

The F5 Configuration utility

The F5 Configuration utility is a web-based application that you use to configure and monitor the load balancing setup on the BIG/ip Controller. In the F5 Configuration utility, you can configure virtual servers, define IP and rate filters, and also configure system objects including the SNMP agent and system settings. The F5 Configuration utility allows you to monitor network traffic, current connections, and the operating system itself, and it also provides

convenient access to downloads such as the SNMP MIB. The F5 Configuration utility requires Netscape Navigator version 4.06 or later, or Microsoft Internet Explorer version 4.0 or later.

The BIG/pipe and BIG/top command line utilities

The BIG/pipe™ utility is the command line counter-part to the F5 Configuration utility. Using BIG/pipe commands, you can configure virtual servers, open ports to network traffic, and configure a wide variety of features. To monitor the BIG/ip Controller, you can use certain BIG/pipe commands, or you can use the BIG/top™ utility, which provides real-time system monitoring. You can use the command line utilities directly on the BIG/ip Controller, or you can execute commands via a remote shell, such as the SSH client (US only), or a Telnet client.

Load balancing options

The BIG/ip Controller offers seven different load balancing modes, including three static modes and four dynamic modes. A load balancing mode defines, in part, the logic that a BIG/ip Controller uses to determine which server should receive a particular connection on a specific port.

Static load balancing

Static load balancing is based on pre-defined user settings, and does not take current performance into account. The BIG/ip Controller supports three static load balancing modes:

- **Round Robin**
  Round Robin mode is a basic load balancing mode that distributes connections evenly across all server ports, passing each new connection to the next server port in line.
- **Ratio**
  The Ratio mode distributes new connections across server ports in proportion to a user-defined ratio. For example, if your array contained one new, high-speed server and two older servers, you could set the ratio so that the high-speed server receives twice as many connections as either of the two older servers.

- **Priority**
  The Priority mode distributes connections in round robin fashion to a specific groups of servers. It begins distributing new connections to the highest priority group. If all servers in that group should go down, it begins distributing connections to servers in the next higher priority group.

Dynamic load balancing

Dynamic load balancing modes use current performance information from each node to determine which node should receive each new connection. The different dynamic load balancing modes incorporate different performance factors:

- **Least Connections**
  In Least Connections mode, the BIG/ip Controller sends each new connection to the node that currently hosts the fewest current connections.
- **Fastest**
  In Fastest mode, the BIG/ip Controller sends each new connection to the node that has the best response time.
- **Observed**
  In Observed mode, the BIG/ip Controller sends each new connection to the node that has the highest performance rating, based on a combination of fewest connections and best response time.
- **Predictive**
  Predictive mode factors in both performance ratings and performance improvement over time.

IP packet filtering, rate classes, and rate filters

The BIG/ip platform supports easy configuration of IP packet filtering. IP packet filtering allows you to control both in-bound and out-bound network traffic. For example, you can specify a single IP address, or a range of IP addresses, from which your site either accepts or denies network traffic. You can also specify one or more IP addresses to which you specifically want to allow or prevent out-bound connections.

The BIG/ip platform also supports rate classes, which are an extension to IP filters. A rate class defines a maximum outgoing packet rate (bits per second) for connections that are destined for a specific IP address or from a range of IP addresses. You can use rate classes to help control the amount and flow of specific network traffic. For example, you can offer faster connection speeds for high priority connections, such as paying customers on an e-commerce site.

Configurable persistence for e-commerce and dynamic content sites

Some e-commerce and other dynamic content sites occasionally require returning users to go the same server that hosted their last connection, rather than being load balanced to a random server. For example, if a customer reserves an airline ticket and holds it for 24 hours, the customer may need to return to a specific back-end server that stores the reservation information in order to purchase the ticket.

The BIG/ip Controller offers a variety of sophisticated persistence options that support this functionality. In addition to simple persistence and standard SSL persistence, the BIG/ip Controller supports cookie persistence. ***Cookie persistence*** is a unique implementation where the BIG/ip Controller stores persistence connection information in a cookie on the client, rather than in a table in its own memory. When the client returns and makes a persistence connection request, the BIG/ip Controller uses the information in the cookie to determine which back-end server should host the client connection.

The BIG/ip Controller supports other useful persistence options, including simple persistence for TCP and UDP (which bases connection information on source and destination IP address) and SSL persistence (which bases connection information on an SSL session ID).

BIG/ip Controller platform options

The BIG/ip Controller platform offers three different systems, each of which can be stand-alone, or can run in redundant pairs:

- **The BIG/ip LB Controller**
  The BIG/ip LB Controller provides basic load balancing features. Note that the BIG/ip LB Controller does not support all of the features documented in this guide.
- **The BIG/ip HA Controller**
  In addition to the basic load balancing features supported on the BIG/ip LB Controller, the BIG/ip HA Controller supports advanced features, such as Extended Content Verification, and also supports high-end security for administrative shell connections. BIG/ip HA Controllers distributed in the US also support encrypted administrative connections using SSH for shell connections and SSL for connections to the web-based F5 Configuration utility.
- **The BIG/ip HA+ Controller**
  The BIG/ip HA+ Controller supports the same features as the BIG/ip HA Controller, but it offers high-end hardware for high traffic sites.

Note

BIG/ip Controllers distributed outside of the United States, regardless of system type, do not support encrypted communications. They do not include the F-Secure SSH client, nor do they support SSL connections to the BIG/ip web server. Instead, you can use the standard Telnet, FTP, and HTTP protocols to connect to the unit and perform administrative functions.

Finding help and technical support resources

In addition to this administrator guide, you can find technical documentation about the BIG/ip Controller in the following locations:

- **Release notes**
  The release note for the current version of the BIG/ip Controller is available on the BIG/ip web server. The release note contains the latest information for the current version, including a list of new features and enhancements, a list of fixes, and, in some cases, a list of known issues.
- **Online help for BIG/ip Controller features**
  You can find help online in three different locations:
  - The BIG/ip web server has a PDF version of this administrator guide. Note that some BIG/ip Controller upgrades replace the online administrator guide with an updated version of the guide.
  - The web-based F5 Configuration utility has online help for each screen. Simply click the Help button in the toolbar.
  - Individual BIG/pipe commands have online help, including command syntax and examples, in standard UNIX man page format. Simply type the command followed by the question mark option (**-?**), and the BIG/ip Controller displays the syntax and usage associated with the command.
- **Third-party documentation for software add-ons**
  The BIG/ip web server contains online documentation for all third-party software included with the BIG/ip Controller, such as GateD.
- **Technical support via the World Wide Web**
  The F5 Networks Technical Support web site, **http://tech.F5.com**, provides the latest technical notes, answers to frequently asked questions, and updates for administrator guides (in PDF format). To access this site, you need to obtain a customer ID and a password from the F5 Help Desk.

What's new in version 3.0

The BIG/ip platform offers the following major new features in version 3.0, in addition to many smaller enhancements.

Active-active redundant controllers

BIG/ip Controller version 3.0 supports a new active-active redundant system where both controllers actively handle connections.

The active-active redundant controller feature allows both controllers to simultaneously manage traffic for different virtual addresses. This option allows you to take advantage of the throughput of both controllers simultaneously. In the event of a failure on one of the controllers, the remaining active controller assumes the virtual servers of the failed machine.

Intelligent traffic control (ITC)

Intelligent traffic control (ITC) is a set of flexible features that increase the level of service and control over Internet traffic. In these features is the ability to identify specific traffic, based on HTTP request data (URLs, HTTP version, HTTP host field), cookies, or client source address and send that traffic to a specific set of servers or devices that can best service the request. These features let you allocate server resources based on the type of application or content requested most.

More flexible load balancing using pools and members

BIG/ip Controller version 3.0 increases load balancing flexibility by introducing the load balancing pool. A load balancing pool is a group of nodes, or other network devices, that are mapped to corresponding virtual server.

In previous versions of the BIG/ip Controller, the commands used to create a virtual server with a node list resulted in a directly associated, node list "pool" with identical function. You can still create node list style virtual servers in this version of the BIG/ip Controller. However, pools are more configurable than node lists. Pools may be configured independently and associated with virtual servers in complex ways.

Selecting a load balancing pool using a rule

With BIG/ip Controller version 3.0, you can create virtual servers indirectly associated with multiple load balancing pools by directly referencing a rule which, in turn, can reference one or more load balancing pools. In other words, you can use a rule to select a pool for a virtual server. When a request that is destined for a virtual server that does not match a current connection, the BIG/ip Controller can select a pool by evaluating a virtual server rule to pick a node pool.

You can create rules that load balance connections based on the client IP address or on the HTTP header information.

- **Client IP address load balancing**
  For client IP address load balancing, you can create a rule that selects a pool for a connection based on the source IP address of the packet.
- **HTTP header load balancing**
  For HTTP header load balancing, you can create a rule that selects a pool for a connection based on different HTTP header values such as: method, uri, version, host, and cookie.

Versatile interfaces

The versatile interfaces features provide the ability to change the security settings of any BIG/ip Controller interface, and to independently enable translation of the source or destination address of IP packets passing through the interface. The virtual interfaces features phase out the concept of **internal** and **external** interfaces. With these features, you can configure a single interface with the source processing attributes of an "internal" interface, and destination processing properties of an "external" interface.

Source and destination processing

The versatile interfaces features provide the ability to change both the source address or destination address and/or route of an IP packet on a BIG/ip Controller interface. The ability to change the source or destination can be turned on independently for each

interface. In practical terms, this means that you can configure an interface to handle traffic going to virtual servers and, independently, you can configure the interface to handle traffic going out from nodes. So, you can have virtual servers and nodes on each interface you have installed in the BIG/ip Controller. This allows the most flexible processing of packets by the BIG/ip Controller. When either the source or destination processing feature is turned off on an interface, the result is a gain in performance. In this way, you can optimize BIG/ip Controller performance with no additional effort.

Interface security

An additional feature of versatile interfaces is the ability to control access to the BIG/ip Controller on any interface. In previous versions of the BIG/ip Controller, the external interface was always in a locked down state, and the internal interface was open. In BIG/ip Controller version 3.0, any interface may be in either a locked down or open state. When an interface is locked down, only the ports essential to the configuration and operation of the BIG/ip Controller and 3DNS Controller are open. When an interface is open, all connections are allowed to and from the BIG/ip Controller through that interface.

Per-connection routing

The per-connection routing option is now available for virtual servers. In situations where the BIG/ip Controller is accepting connections for virtual servers from more than one gateway, you can send the return data back through the same device from which the connection originated. Use this option to spread the load among outbound routers, or to ensure that connections go through the same device if that device is connection-oriented, such as a proxy, cache, firewall, or VPN router. You can do this by defining a pool that contains the list of gateways from which the connections are received, and then associating the pool with a virtual server using the **lasthop** keyword.

Secure forwarding virtual servers

You can now create a forwarding virtual server in BIG/ip Controller, version 3.0. A forwarding virtual server is a type of virtual server that simply forwards all traffic to the specific node or network specified in the virtual IP address. It has no associated nodes. A forwarding virtual server has the added benefit of keeping statistics for the amount of forwarded traffic, and does not perform load balancing.

Transparent virtual servers

You can now create a transparent virtual server in BIG/ip Controller, version 3.0. The new **translate** keyword allows you to turn off address translation for any virtual server. This simplifies and generalizes the Transparent Node mode and bonfire mode from previous versions of the BIG/ip Controller that were used to load balance transparent devices. This can also be useful when the BIG/ip Controller is load balancing devices that have the same IP address. This is typical with the nPath routing configuration where duplicate IP addresses are configured on the loopback device of several servers. You can control whether address translation is enabled for a virtual server using the translate keyword. Address translation is enabled by default. You can enable or disable address translation for any valid address.

Virtual server port translation

A new attribute has been added to virtual servers to control whether port translation is enabled for a Virtual Server. Port translation is turned **on** by default. An exception to this is if the port defined for a member is port zero. Members with a zero port cannot do translation because zero is not a valid port. You can control whether port translation is enabled for a virtual server using the translate keyword. Port translation is enabled by default. You can enable or disable port translation for any valid port.

Reset connections on service down

You can configure individual virtual servers so that the BIG/ip Controller sends connection resets to the end points of TCP connections when the controller determines that the service they are using has gone down. This feature is currently only used in conjunction with service checking. Node pings that time out do not cause connection resets to be sent. Only TCP connections receive the resets. UDP connections are not reset because there is no shutdown mechanism for UDP connections.

Cookie persistence hash mode

The BIG/ip Controller offers a new HTTP cookie persistence hash mode. The hash mode for cookie persistence is a new feature available with this release of the BIG/ip Controller. The hash mode allows you to specify a certain number of bytes in a cookie to determine the destination of the connection. Cookie hash mode is used to map a cookie value to a node. This mode is then used to persistently connect clients presenting cookies to a given node.

Enhancements to configuration and monitoring tools

The F5 Configuration utility

The F5 Configuration utility supports all of the new features in version 3.0. In addition to several new screens, some existing screens have been reorganized to accommodate new settings. For a review of each particular screen, click the **Help** button in the toolbar. In addition to these changes, we made the following improvements:

- The port on the virtual server is now automatically opened when you define virtual servers.
- There is quicker navigation to Global Virtual Ports and Global Node Ports from the left-hand navigation pane.
- We have updated the Virtual Server and SNAT creation step, which assembles groups of nodes through interactive list editors.

- We have improved the high-level listing of objects including Virtual Servers, Global Virtual Ports, Nodes, and Global Node Ports. The new lists present more summary information for each list item. Also, items in the high-level lists that are themselves lists (like the nodes associated with a Virtual Server in a default pool), which means the you can navigate to one of the items on that list by clicking the list and choosing a particular item.
- Now there is a separate extended service check category on ECV/EAV property pages for Simple service check. That category was previously part of the ECV select box menu.
- You can now configure multiple trap communities and trap sinks for SNMP using the new lists editors.
- The load balancing options for node list virtual servers have moved from the BIG/ip System Properties Page to the Virtual Servers List page.
- The network interface card properties page is improved to show shared IP addresses. This page also has new navigation options for VLAN tag information.

BIG/pipe command line utility

The BIG/pipe® command line utility has been updated and streamlined. In addition to new commands for new features, certain existing commands support new syntax to make for more efficient configuration.

System control variables

There are new system control variables, and the default settings for some existing system control variables have changed in certain cases. To view a description of the system control variables used by BIG/ip Controllers, refer to the ***BIG/ip Controller Reference Guide***, *BIG/ip System Control Variables.*

The SNMP MIB

The BIG/ip Controller includes an updated SNMP MIB that supports the new features, as well as enhanced support for existing features.

2

Working with Special Features

- **Introducing special features**
- **Using specialized load balancing modes**
- **Controlling network access and traffic flow with filters**
- **Working with more than two interface cards**
- **Optimizing large configurations**
- **Using the versatile interface configuration options**
- **Using virtual server options for maximum flexibility**

Introducing special features

In addition to the basic setup features available on the BIG/ip Controller, a number of special setup features can be used to optimize your network. This chapter describes special setup options available on the BIG/ip Controller. These features are optional, and may not be required in your implementation of the BIG/ip Controller. The following topics are described in this chapter:

- Using specialized load balancing modes
- Controlling network access and traffic flow with filters
- Working with more than two interface cards
- Optimizing large configurations
- Using the versatile interface configuration options

Using specialized load balancing modes

Load balancing is an integral part of the BIG/ip Controller. A load balancing mode defines, in part, the logic that a BIG/ip Controller uses to determine which node should receive a connection hosted by a particular virtual server. The BIG/ip Controller supports specialized load balancing modes that dynamically distribute the connection load, rather than following a static distribution pattern such as Round Robin. Dynamic distribution of the connection load is based on various aspects of real-time server performance analysis, such as the current number of connections per node or the fastest node response time. The following section describes how each load balancing mode distributes connections, as well as how to set the load balancing mode on the BIG/ip Controller.

◆ Note

These load balancing modes apply globally. For information about the specific load balancing methods designed for use with pools, see Chapter 3, Introducing Intelligent Traffic Control (ITC).

Understanding individual load balancing modes

Individual load balancing modes take into account one or more dynamic factors, such as current connection count. Because each application of the BIG/ip Controller is unique, and node performance depends on a number of different factors, we recommend that you experiment with different load balancing modes, and choose the one that offers the best performance in your particular environment.

◆ Note

It is important to note that the load balancing methods described in this section are the advanced load balancing modes. For more information about Round Robin or Ratio mode, see the ***BIG/ip Controller Getting Started Guide****.*

Fastest mode

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

Least Connections mode

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

Observed mode

Observed mode uses a combination of the logic used in the Least Connection and Fastest modes. In Observed mode, nodes are ranked based on a combination of the number of current connections and the response time. Nodes that have a better balance of fewest connections and fastest response time receive the

a greater proportion of the connections. Observed mode also works well in any environment, but may be particularly useful in environments where node performance varies significantly.

Predictive mode

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

Priority mode

Priority mode is a special type of round robin load balancing. In Priority mode, you define groups of nodes and assign a priority level to each group. The BIG/ip Controller begins distributing connections in a round robin fashion to all nodes in the highest priority group. If all the nodes in the highest priority group go **down** or hit a connection limit maximum, the BIG/ip Controller begins to pass connections on to nodes in the next lower priority group.

For example, in a configuration that has three priority groups, connections are first distributed to all nodes set as priority 3. If all priority 3 nodes are down, connections begin to be distributed to priority 2 nodes. If both the priority 3 nodes and the priority 2 nodes are down, connections then begin to be distributed to priority 1 nodes, and so on. Note, however, that the BIG/ip Controller continuously monitors the higher priority nodes, and each time a higher priority node becomes available, the BIG/ip Controller passes the next connection to that node.

Setting the global load balancing mode

The load balancing mode is a system property of the BIG/ip Controller, and it applies to all standard and wildcard virtual servers defined in the configuration.

To set the global load balancing mode in the F5 Configuration utility

1. In the navigation frame, click Virtual Servers.
   The Virtual Servers screen opens.
2. In the **Node List Load Balance Method** box, choose the desired load balancing mode.
3. Click **Apply**.

WARNING

If you select Ratio mode or Priority mode, be sure to set the ratio weight or priority level for each node address in the configuration.

To set the load balancing mode on the command line

The command syntax for setting the load balancing mode is:

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

Table 2.1 describes the valid options for the **<mode name>** parameter.

Mode Name	Description
`priority`	Sets load balancing to Priority mode.
`least_conn`	Sets load balancing to Least Connections mode.
`fastest`	Sets load balancing to Fastest mode.
`observed`	Sets load balancing to Observed mode.
`predictive`	Sets load balancing to Predictive mode.

Table 2.1 *Options for the* ***<mode name>*** *parameter.*

Note

It is important to note that the load balancing methods described in this section are the advanced load balancing modes. For more information about Round Robin or Ratio mode, see the ***BIG/ip Controller Getting Started Guide****.*

Setting ratio weights and priority levels for node addresses

If you set the load balancing mode to either Ratio mode or Priority mode, you need to set a special property on each node address.

- **Ratio weight**
  The ratio weight is the proportion of total connections that the node address should receive. The default ratio weight for a given node address is **1**. If all node addresses use this default weight, the connections are distributed equally among the nodes.
- **Priority level**
  The priority level assigns the node address to a specific priority group.

To set ratio weights and priority levels in the F5 Configuration utility

1. In the navigation pane, click **Nodes**.

2. In the Nodes list, click the node for which you want to set the ratio weight.
   The Node Properties screen opens.
3. In the Node Properties screen, click the **Address** of the node.
   The Global Node Address Properties screen opens.
4. In the **Ratio or Priority** box, type the ratio weight of your choice.
5. Click the **Apply** button to save your changes.

To set ratio weights on the command line

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

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

The following example defines ratio weights and priority for three node addresses. The first command sets the first node to receive half of the connection load. The second command sets the two remaining node addresses to each receive one quarter of the connection load.

```
bigpipe ratio 192.168.10.01  2
bigpipe ratio 192.168.10.02  192.168.10.03  1
```

WARNING

If you set the load balancing mode to Ratio or Priority, you must define the ratio or priority settings for each node address. The value you define using the ***bigpipe ratio*** *command is used as the ratio value if Ratio is the currently selected load balancing mode, and the same value is used as the priority level if Priority is the currently selected load balancing mode.*

Controlling network access and traffic flow with filters

Filters control network traffic by setting whether packets are forwarded or rejected at the external network interface. Filters apply to both incoming and outgoing traffic. When creating a filter, you define criteria which are applied to each packet that is processed by the BIG/ip Controller. You can configure the BIG/ip Controller to forward or block each packet based on whether or not the packet matches the criteria.

The BIG/ip Controller supports two types of filters, IP filters and rate filters.

IP filters

Typical criteria that you define in IP filters are packet source IP addresses, packet destination IP addresses, and upper-layer protocol of the packet. However, each protocol has its own specific set of criteria that can be defined.

For a single filter, you can define multiple criteria in multiple, separate statements. Each of these statements should reference the same identifying name or number, to tie the statements to the same filter. You can have as many criteria statements as you want, limited only by the available memory. Of course, the more statements you have, the more difficult it is to understand and maintain your filters.

Configuring IP filters

When you define an IP filter, you can filter traffic in two ways:

- You can filter traffic going to a specific destination or coming from a specific destination, or both.
- The filter can allow network traffic through, or it can reject network traffic.

Defining an IP filter in the F5 Configuration utility

1. Click **IP Filters** in the navigation pane.
   The IP Filters screen opens.
2. In the IP Filters screen, click **Add Filter**.
   The Add IP Filter screen opens.
3. On the Add IP Filter screen, in the **Name** box, type a filter name.
4. From the **Type** list, choose **Accept Packet** to allow traffic, or **Deny Packet** to reject traffic.
5. In the **Source IP Address** box, only if you want the filter to be applied to network traffic based on its source, enter the IP address from which you want to filter traffic.
6. In the **Source Port** box, only if you want the filter to be applied to network traffic based on its source, enter the port number from which you want to filter traffic.
7. In the **Destination IP Address** box, enter the IP address to which you want to filter traffic, only if you want the filter to be applied to network traffic based on its destination.
8. In the **Destination Port** box, enter the port number to which you want to filter traffic, only if you want the filter to be applied to network traffic based on its destination.
9. Click **Add** to add the IP filter to the system.

◆ Note

For information on configuring IP filters and rate filter on the command line, refer to the IPFW man page by typing ***man ipfw*** *on the command line. You can configure more complex filtering through the IPFW command line interface.*

Rate filters and rate classes

In addition to IP filters, you can also define rates of access by using a rate filter. Rate filters consist of the basic filter and a rate class. Rate classes define how many bits per second are allowed per connection and the number of packets in a queue.

Configuring rate filters and rate classes

Rate filters are a type of extended IP filter. They use the same IP filter method, but they apply a ***rate class*** which determines the volume of network traffic allowed through the filter.

Tip

You must define at least one rate class in order to apply a rate filter.

Rate filters are useful for sites that have preferred clients. For example, an e-commerce site may want to set a higher throughput for preferred customers, and a lower throughput for random site traffic.

Configuring rate filters involves both creating a rate filter and a rate class. When you configure rate filters, you can use existing rate classes. However, if you want a new rate filter to use a new rate class, you must configure the new rate class before you configure the new rate filter.

To configure a new rate class in the F5 Configuration utility

1. Click **Rate Filters** in the navigation pane.
   The Rate Filters screen opens.
2. In the Rate Filters screen, click **Add Class**.
   The Rate Class screen opens.
3. On the Rate Class screen, in the **Name** box, type a rate class name.
4. In the **Bits Per Second Allowed** box, enter the maximum number of bits per second that you want the class to allow.

5. In the **Minimum Number of Bits Outstanding** box, enter the minimum number of bits required to be sent for processing from the queue at one time.
6. In the **Queue Length (in Packets)** box, enter the maximum number of packets allowed in the queue. Once the BIG/ip Controller fills the queue, it begins to drop subsequent packets received.
7. Click **Add** to add the rate class to the system.

Note

For information on configuring IP filters and rate filter on the command line, refer to the IPFW man page.

After you have added a rate class, you can configure rate filters for your system.

To configure a rate filter in the F5 Configuration utility

1. Click **Rate Filters** in the navigation pane.
   The Rate Filters screen opens.
2. In the Rate Filters screen, click **Add Class**.
   The Add Class screen opens.
3. On the Rate Filter screen, in the **Name** box, type a name for the rate filter.
4. From the **Rate Class** list, choose a rate class. Note that you must have a rate class defined before you can proceed.
5. In the **Source IP Address** box, enter the IP address from which you want to filter traffic, only if you want the filter to be applied to network traffic based on its source.
6. In the **Source Port** box, enter the port number from which you want to filter traffic, only if you want the filter to be applied to network traffic based on its source.
7. In the **Destination IP Address** box, enter the IP address to which you want to filter traffic, only if you want the filter to be applied to network traffic based on its destination.

8. In the **Destination Port** box, enter the port number to which you want to filter traffic, only if you want the filter to be applied to network traffic based on its destination.
9. Click **Add**.

Note

For information on configuring IP filters and rate filter on the command line, refer to the IPFW man page.

Working with more than two interface cards

When you configure a BIG/ip Controller with more than two interface cards installed, you need to address the following issues:

- Additional interfaces should be configured.
- You need to specify an interface for each virtual server.
- You need to define an interface for NATs.
- You need to define an interface for SNATs.
- Verify routing with multiple NICs.
- Edit **httpd.conf** for network administration with the BIG/ip web server.

Configuring additional interfaces with the First-Time Boot utility

The first step in configuring the BIG/ip Controller with additional interfaces is to run the First-Time Boot utility. This utility detects how many NICs are present in the BIG/ip Controller and displays a list of NICS detected. Choose the interface from the list that you want to configure.

You can also designate one of your additional internal NICs with the IP address for which access is permitted for network administration using SSH (or Telnet for international users).

The First-Time Boot utility, **config**, detects and configures additional interfaces if they are present in the BIG/ip Controller.

Running the First-Time Boot utility

As Administrator with root-level permission, enter the following command from the command line:

```
config
```

When asked to configure the web server, you are prompted to define a domain name for the interface.

The First-Time Boot utility creates a new **/etc/netstart** script which supports more than two NICs. It also modifies the **/etc/ethers** and the interface entries in the BIG/db database.

◆ Note

*When you rerun the First-Time Boot utility, it does not replace or change your existing **/etc/bigip.conf** or your **/etc/bigd.conf** files.*

You may need to edit the **/etc/bigip.conf** file using a text editor such as **vi** or **pico** to add the appropriate interface statements. For example, if you want to designate **exp2** as an internal, destination processing interface, add the following lines to the **/etc/bigip.conf** file:

```
interface exp0 dest enable
interface exp0 source disable
interface exp0 adminport lockdown
interface exp2 failsafe disarm
interface exp2 timeout 30
```

Once you are done editing the **bigip.conf** file, reboot the BIG/ip Controller, or restart **bigd** by typing **bigd** on the command line and pressing Enter, in order to implement your changes.

Specifying an interface for a virtual address

When you define a virtual server on a BIG/ip Controller that has more than one external interface (destination processing), you need to specify the external interface (destination processing) that the virtual server's address is associated with.

WARNING

All virtual servers that share a virtual address must be associated with the same external interface.

To specify a virtual server interface in the F5 Configuration utility

1. In the navigation pane, click **Virtual Servers**.
   The Virtual Servers screen opens.
2. In the toolbar, click **Add Virtual Server**.
   The Add Virtual Server screen opens.
3. In the Add Virtual Server screen, type in the properties for the virtual server you want to add including the address and port.
4. Click **Add**.

To specify a virtual server interface on the command line

You can define virtual servers with the **bigpipe vip** command. Normally, a virtual server is added to the external interface with a network address that matches the network of the virtual address. However, with multiple NICs, you can specify which external interface (destination processing) a virtual server is added to using the **bigpipe vip** command. To do this, add the **<ifname>** argument to the command.

```
bigpipe vip <virt addr>:<port>[/<bitmask>] [<ifname>] \
   [unit <ID>] define <node addr>:<port> ... <node addr>:<port>
```

```
bigpipe vip <virt addr>:<port> [<ifname>] [unit <ID>] [netmask \
   <netmask> [broadcast <broadcast_ip>]] \
   define <node addr>:<port> ... \ <node addr>:<port>
```

You can set the **<ifname>** parameter to **none** if you want to prevent BIG/ip from issuing ARP requests for a specific virtual server. The traffic for a virtual server is accepted on any interface with destination processing enabled, even if the BIG/ip Controller only responds to ARP requests on one interface or if you specify **none**.

◆ Note

*This has the same effect as using the **sysctl** variable **bigip.vipnoarp**, but on a server-by-server basis. The **sysctl** variable **bigip.vipnoarp** is deprecated. We recommend defining the interface **none**.*

The following example shows how to define a virtual server that is added to a Gigabit Ethernet NIC.

```
bigpipe vip 210.12.150.100:80/24 sk0 define 192.158.11.100:80 \
   192.158.11.101:80 192.158.11.102:80
```

Specifying an interface for a NAT address

When you define a NAT address on a BIG/ip Controller that has more than one external interface, you need to specify the external interface that the virtual server's address is associated with.

To specify an interface for a NAT in the F5 Configuration utility

1. Click **NATs** in the navigation pane.
   The Network Address Translations (NATs) screen opens.
2. Click the NAT you want to configure.
   The NAT Properties screen opens.
3. In the **Interface** list, choose the destination processing interface to which you want to assign this NAT.
4. Click the **Apply** button.

To specify an interface for a NAT on the command line

When mapping a network address translation with the **bigpipe nat** command, you must now specify which external interface a virtual IP address is added to by using the **<ifname>** parameter.

```
bigpipe nat <internal_ip> to <external_ip> [/<bitmask>] \
   [<ifname>] [unit <ID>]
bigpipe nat <internal_ip> to <external_ip> [netmask \
   <netmask>] [broadcast <broadcast_ip>] |/<bitmask>] \
   [<ifname>] [unit <ID>]
```

The following example shows how to define a NAT where the IP address represented by <**external_ip**> is added to an Intel NIC.

```
bigpipe nat 11.0.0.100 to 10.0.140.100/24 exp0
```

Specifying an interface for a SNAT address

When you define a SNAT address on a BIG/ip Controller that has more than one external interface, you need to specify the external interface that the virtual server's address is associated with.

WARNING

All virtual servers that share a virtual address must be associated with the same external interface.

To specify an interface for a SNAT in the F5 Configuration utility

1. Click **Secure NATs** in the navigation pane.
   The Secure Network Address Translations (SNATs) screen opens.
2. Click the SNAT you want to configure.
   The SNAT Properties screen opens.
3. In the **Interface** list, choose the destination processing interface to which you want to assign this SNAT.
4. Click **Apply**.

To specify an interface for a SNAT on the command line

When mapping a secure network address translation with the **bigpipe snat** command, you must specify which external interface a virtual IP address is added to by using the <**ifname**> parameter.

```
bigpipe snat map <internal_ip> to <external_ip> [/<bitmask>] \
   [<ifname>] [unit <ID>]
bigpipe snat map <internal_ip> to <external_ip> [<ifname>]
   [netmask] <netmask> [broadcast <broadcast_ip>] | /<bitmask>]
   [unit <ID>]
```

The following example shows how to define a SNAT where the IP address represented by <**external_ip**> is added to an Intel NIC.

```
bigpipe snat map 11.0.0.100 to 10.0.140.100/24 exp0
```

Routing with multiple NICs

Use Router Discovery Protocol (RDP) for routing on a BIG/ip Controller with more than one interface. For router configuration information, please refer to documentation included with your router.

Example configurations for more than two NICs

The following two scenarios for configuring your network with more than two NICs contain important details related to creating virtual servers.

- In the first scenario, you have one gateway, with two routers and two BIG/ip Controllers behind it.
- The second scenario involves two routers providing access into the network, with two BIG/ip Controllers behind them.

Each scenario configuration has advantages and disadvantages related to how you set up your virtual servers, which are detailed in the following descriptions. For instructions on how to create virtual servers on specific interfaces, see *Specifying an interface for a virtual address*, on page 2-13.

Configuration example: One gateway

When your network is configured with one gateway to the Internet, and has two routers connected to two BIG/ip Controllers behind that gateway, we recommend that you connect the first router to one of the external interfaces on each BIG/ip Controller and the other router to the remaining external interfaces on each BIG/ip Controller for maximum redundancy.

If the gateway is running OSPF, it maintains redundancy by ensuring that there is only one path from your network to the Internet. In the unlikely event that the active router should fail, OSPF determines that the router is not functioning properly, and sends subsequent connections to the second router. Existing connections will persist by going through the second router. Under such conditions, when you create a virtual server, we recommend that you create it to use the default interface.

By using the default interface, the virtual server is guaranteed to handle connections in an efficient manner by cooperating with OSPF's attempts to compensate for the failed router. Otherwise, if the virtual server is configured to use one specific external interface, there is no way for connections to arrive at the virtual server when the router leading to it fails.

Configuration example: Two gateways

When your network is configured with two routers to the Internet, and has two BIG/ip Controllers behind them, we also recommend that you connect the first router to one of the external interfaces on each BIG/ip Controller and the other router to the remaining external interfaces on each BIG/ip Controller. However, in this configuration, you have two entry points into your network, one through each router.

You have the flexibility to decide how you want clients to access various web sites on your virtual servers based on how the virtual servers are created. For example, your research department uses an intranet site to exchange information and that sensitive material needs to be protected. Then, by restricting access to the external interface to only your researchers, you protect the information. On

the other hand, you want your employees, including the researchers, to have access to the Human Resources information on the intranet.

You would then create the virtual server that hosts the Human Resources intranet using the default external interface so that any employee connecting from any location can make a connection to that virtual server.

Optimizing large configurations

The BIG/ip Controller supports up to 40,000 virtual servers and nodes combined. Larger configurations on a BIG/ip Controller, such as those that exceed 1,000 virtual servers or 1,000 nodes, introduce special configuration issues. To ensure a high performance level, you need to change certain aspects of the BIG/ip Controller's management of virtual servers and nodes. The following steps can be taken to optimize a large configuration.

- Reduce ARP traffic on the external network
- Reduce the number of node pings and service checks issued by the BIG/ip Controller

Reducing ARP traffic on the external network

The BIG/ip Controller maintains an IP alias on its external interface for each virtual address that it manages. IP aliases are broadcast on the network when a virtual server is defined, and also each time a BIG/ip Controller switches from standby mode to active mode in a redundant system. If you have defined thousands of virtual addresses in the BIG/ip Controller configuration, the corresponding ARP requests may lead to a significant increase in network traffic.

This type of configuration also increases fail-over recovery time in BIG/ip redundant systems. When a fail-over occurs, the BIG/ip Controller that becomes the active machine creates an IP alias for each virtual server that it manages. Normally, this process takes less than one second. However, if the BIG/ip Controller has 8,000

virtual servers, this process can take as long as 90 seconds. The active BIG/ip Controller is unresponsive during the time it creates the IP aliases, and it cannot begin processing connections until the IP aliasing is complete.

To ensure a fast fail-over process, and to help reduce the amount of ARP requests a router must make, you should run the BIG/ip Controller in **bigip.vipnoarp** mode. In bigip.vipnoarp mode, the BIG/ip Controller does not create IP aliases for virtual servers. Instead, network traffic bound for virtual servers configured on the BIG/ip Controller are routed using the BIG/ip Controller's external interface as a gateway. Configuring bigip.vipnoarp mode is a two-step process:

- On the router, you must configure a gateway route to the virtual servers using the BIG/ip Controller's external interface IP address.
- On the BIG/ip Controller itself, you must change the **bigip.vipnoarp** system control variable. Note that you can use either the F5 Configuration utility, or the **sysctl** command, to change system control variables.

Note

You can enable the noarp option on a virtual server basis. For more information about enabling this option on an individual virtual server basis, see Specifying an interface for a virtual address, on page 2-13

Note

You can enable bigip.vipnoarp mode only if you have the ability to add a gateway route to your router. Note that in redundant systems, you need to use the shared external IP address as the gateway address for the virtual servers configured on the BIG/ip Controller.

Configuring the router

In the router configuration, you need to define a static route as the gateway for each virtual address managed by the BIG/ip Controller. The static route should set the gateway address to the IP address of the external interface on the BIG/ip Controller. For example, if the shared external address of a BIG/ip redundant system is 11.0.0.100, and all virtual servers configured on the BIG/ip redundant system use IP addresses 11.0.1.50 through 11.0.1.55, you need to configure the router to use 11.0.0.100 as a gateway to the 11.0.1.* subnet. Such a definition on a UNIX-like router would read:

```
route add -net 11.0.1.0 gw 11.0.0.100
```

Activating bigip.vipnoarp mode in F5 Configuration utility

In the F5 Configuration utility, the **bigip.vipnoarp** mode setting is under BIG/ip `sysctl` configuration. To turn this mode on, simply check the **Disable IP Aliases on Virtual Servers** box. To turn this mode off, clear the **Disable IP Aliases on Virtual Servers** box.

WARNING

We recommend that you do not toggle this mode on or off while the virtual servers are defined. Resetting the variable at that time may result in system anomalies.

Activating bigip.vipnoarp mode on the command line

You can activate **bigip.vipnoarp** mode in one of two ways:

- You can edit the **/etc/rc.sysctl** file in a text editor, and then reboot the system.
- You can immediately enable or disable the mode using **sysctl** commands.

If you choose to edit the **/etc/rc.sysctl** file, you simply need to add the following line to the file to activate vipnoarp mode:

```
sysctl -w bigip.vipnoarp=1
```

To deactivate bigip.vipnoarp mode, you can either comment the line out, or delete it from the **/etc/rc.sysctl** file altogether. Once you edit the file, the changes do not take affect until you reboot the system.

To immediately activate bigip.vipnoarp mode, type the following on the command line:

```
bigpipe -r
sysctl -w bigip.vipnoarp=1
bigpipe -f /etc/bigip.conf
```

To immediately deactivate bigip.vipnoarp mode, type the following on the command line:

```
bigpipe -r
sysctl -w bigip.vipnoarp=0
bigpipe -f /etc/bigip.conf
```

WARNING

*We recommend that you do not toggle the bigip.vipnoarp mode **on** or **off** while the virtual servers are defined. Resetting the sysctl variable at that time may lead to a system crash.*

Reducing the number of node pings and service checks issued by the BIG/ip Controller

The BIG/ip Controller checks node status at user-defined intervals in two different ways:

- The BIG/ip Controller can issue a ***node ping*** to all node addresses that it manages. If the BIG/ip Controller receives a response to a node ping from a specific node address, all nodes associated with that node address are marked **up** and available for connections. The node ping the ICMP protocol.
- The BIG/ip Controller can also perform a ***service check***. For each node that uses service check, the BIG/ip Controller connects to the node and attempts to establish a connection with the service configured on the node port. If the BIG/ip Controller is able to establish a connection with the service, the BIG/ip

Controller marks the node **up**. If the BIG/ip Controller cannot establish a connection with the service, the BIG/ip Controller marks the node **down**. It is important to note that the node is marked **down**, even if the node's address is able to respond to the BIG/ip Controller's simple node ping.

If a BIG/ip Controller's configuration includes thousands of nodes, the node pings and service checks begin to take up more resources on both the BIG/ip Controller and the servers than is preferred. You can significantly reduce the number of node pings and service checks in configurations that have a group of node addresses which are all IP aliases on the same server. For each group of node addresses that points to a given server, you can select one node address out of the group to represent all node addresses in the group. The representative node address is referred to as the ***node alias***. When the BIG/ip Controller issues a node ping or service check, it sends the ping or performs the service check only on the node alias, rather than on all nodes in the group. If the BIG/ip Controller receives a valid response before the time-out expires, it marks all nodes associated with the node alias as **up** and available to receive connections. If the BIG/ip Controller does not receive a valid response before the time-out expires, it marks all of the nodes associated with the node alias as **down**.

An important note about service checks

You can set the BIG/ip Controller to use a node alias for nodes that are configured for service checks; however, there are some limitations to this implementation. Service checks are port-specific, unlike node pings which are merely sent to a node address. If you assign a node alias to a node that uses service check, the node alias must be configured to support the port number associated with the node. If the node alias is not configured properly, the BIG/ip Controller can not establish a conversation with the service that the specific node supports, and the service check is invalid.

◆ Note

If you have configured different ports on each node to handle a specific Internet service and you want to use IP aliases, you can use BIG/pipe commands to work around the situation. Refer to the

***BIG/ip Controller Reference Guide**, BIG/pipe Command Reference for more information about the* **`bigpipe alias`** *command.*

Setting up node aliases in the F5 Configuration utility

In the F5 Configuration utility, each node address has a set of properties associated with it, including the **Node Alias** property. Note that before you define a node alias for a specific node address, you may want to check the properties for each node that uses the node alias. The node alias must support each port used by a node that is configured for service check, otherwise the service check results are invalid.

1. In the navigation pane, select **Nodes**
   The Virtual Servers screen opens.
2. In the Node Properties table, select the node address.
   The Node Address Properties page opens.
3. In the **Node Alias** box, type the node alias.
4. Click the **Apply** button.

Setting up node aliases using the BIG/pipe command line utility

The BIG/pipe command line utility allows you to set node aliases for multiple nodes at one time. With the **`bigpipe alias`** command, you can do three things:

- View all node aliases defined in the current configuration
- View the node alias associated with a specific node address
- Define a node alias for one or more node addresses

For details about working with the **`bigpipe alias`** command, refer to the ***BIG/ip Controller Reference Guide**, BIG/pipe Command Reference*.

Using the versatile interface configuration options

The versatile interfaces option adds more flexibility for configuring interfaces. You can now change both the source address or destination address and/or route of an IP packet.

In previous versions of the BIG/ip Controller, interfaces were designated as internal or external. With this version of the BIG/ip Controller you can configure specific interface properties based on the properties in Table 2.2.

Interface type	Interface properties
Internal	Processes source addresses Administrative ports open
External	Processes destination addresses Administrative ports locked down

Table 2.2 *The properties for internal and external interfaces*

The ability to change the source or destination can be turned on independently. Essentially, this means you can configure an interface so that it handles traffic going to virtual servers and, independently, you can configure the interface to handle traffic coming in from nodes. You can configure virtual servers and nodes on each interface installed on the BIG/ip Controller. This allows for the most flexible processing of packets by the BIG/ip Controller. When either the source or destination processing feature is turned off on an interface, there is a gain in performance.

When you enable destination processing on a BIG/ip Controller interface, the interface functions in the following manner:

- When the destination address on the packet is a virtual server, then the interface routes the packet to the Node that is handling the connection that the packet is a part of, picking one if necessary, and depending on how the virtual server is configured, the interface translates the destination address to the node address.

- When the destination address on the packet is the external address of a NAT, then the interface translates the destination address to the internal address of the NAT.
- When the destination address on the packet is the external address of a SNAT, then the interface translates the destination address to the internal address from which the SNAT connection originated.

When you enable source processing on a BIG/ip Controller interface, the interface functions in the following manner:

- When the source address on the packet is a node, and the packets are destined to a client for whom there is an existing connection, and depending on how the virtual server is configured, the interface translates the source address to the address of the virtual server.
- When the source address on the packet is the internal address of a NAT, then the interface translates the source address to the external address of the NAT.
- When the source address on the packet is the internal address of a SNAT, then the interface translates the source address to the external address of the SNAT.

You can turn on both source and destination processing for an interface. This is possible because their functions do not overlap. For example, a NAT changes the source address on packets coming from clients so that they look like they have a different IP address, and virtual servers change the destination address to load balance the destination. There is no reason why you cannot do both the NAT translation and the virtual server translation. There are some combinations of virtual server and NAT source processing and virtual server and NAT destination processing that do not make sense. For example, if a virtual server processes a packet during source processing, the packet is not handled by virtual server destination processing. Also, if a virtual server processes a packet during destination processing, the packet is not handled by virtual server source processing.

Destination route and translation processing

When destination processing is enabled on an interface, the BIG/ip Controller processes packets arriving at the interface when those packets are addressed to a virtual server, SNAT, or NAT external address.

It is useful to note that there are two independent activities associated with destination processing: routing and translation. For example, wildcard virtual servers load balance connections across transparent network devices (such as a router or firewall), but they do not perform translation. In fact, translation can be turned off for all virtual servers (see *Configuring transparent virtual servers*, on page 2-32). Also, with the new forwarding virtual servers, neither next hop load balancing nor translation will occur for connections. These virtual servers only forward packets and so connections can pass through BIG/ip Controller without being manipulated in any way.

When you plan which type of processing to use in the BIG/ip Controller configuration, consider these questions:

- What traffic is translated?
- When those packets reach the BIG/ip Controller interface, does the source IP address, or destination IP address need to be translated?
- Which connections are load balanced across multiple devices?

These questions help identify what kind of processing is required for the network interfaces on the BIG/ip Controller.

Source translation processing

When source translation processing is enabled on an interface, then the BIG/ip Controller processes packets arriving at the interface when those packets are coming from a node, SNAT, or NAT internal address. In this situation, the interface rewrites the source address of the IP packet, changing it from the real server's IP address, or internal NAT address, to the virtual server or external NAT address, respectively. Also, when the new last hop feature is

enabled on a virtual server (see *Using per-connection routing*, on page 2-29), the packet is routed back to the network device that first transmitted the connection request to the virtual server.

To configure source and destination processing from the command line

Use the following syntax to configure source and destination processing on the specified interface:

```
bigpipe interface <interface> dest [ enable |
    disable ]
bigpipe interface <interface> source [ enable |
    disable ]
```

The following example command enables destination processing on the interface **exp0**:

```
bigpipe interface exp0 dest enable
```

The following example command enables source processing on the interface **exp1**:

```
bigpipe interface exp1 source enable
```

To configure source and destination processing in the F5 Configuration utility

1. In the navigation pane, click NICs.
   The Network Interface Cards screen opens. You can view the current settings for each interface in the Network Interface Card table.
2. In the Network Interface Card table, click the name of the interface you want to configure.
   The Network Interface Card Properties screen opens.
   - To enable source processing for this interface, click the **Enable Source Processing** check box.
   - To enable destination processing for this interface, click the **Enable Destination Processing** check box.
3. Click the **Apply** button.

Interface security

You can use the **adminport** option to control the security on an interface. The **lockdown** keyword configures the port lockdown used in previous versions of the BIG/ip Controller on the specified interface. If you use this option when you configure an interface, only ports essential to the configuration and operation of BIG/ip Controller and 3DNS Controller are opened. The **open** keyword allows all connections to and from BIG/ip Controller through the interface you specify.

To configure interface security from the command line

Use the following syntax to configure interface security on the specified interface:

```
bigpipe interface <interface> adminport lockdown
bigpipe interface <interface> adminport open
```

Use the following example command to lock down connections to all ports except the administration ports on **exp0**:

```
bigpipe interface exp0 adminport lockdown
```

Use the following example command to allow connections to all ports on **exp1**:

```
bigpipe interface exp1 adminport open
```

To configure interface security in the F5 Configuration utility

1. In the navigation pane, click NICs.
   The Network Interface Cards screen opens. You can view the current settings for each interface in the Network Interface Card table.
2. In the Network Interface Card table, click the name of the interface you want to configure.
   The Network Interface Card Properties screen opens.
3. To set the administration properties, click the Enable Admin list. Choose one of the following options:

- **Lockdown**
  Choose this option to lock down all ports except the ports used for administrative access on this interface.
- **Open**
  Choose this option to open allow connections to all ports on this interface.

4. Click the **Apply** button.

Using advanced virtual server options

You can use the BIG/ip Controller virtual server options in combinations that match the hardware and load balancing needs of your network. This section describes how advanced virtual server configurations including:

❖ Using per-connection routing

❖ Configuring forwarding virtual servers

❖ Configuring transparent virtual servers

❖ Using virtual server port translation

❖ Resetting connections when a service is down

Using per-connection routing

In situations where the BIG/ip Controller accepts connections for virtual servers from more than one router or firewall, you can send the return data back through the same device from which the connection originated. You can use this option to spread the load among outbound routers or firewalls, or to ensure that connections go through the same device if that device is connection-oriented, such as a proxy, cache, or VPN router.

The device from which a connection originated is sometimes referred to as the ***last hop*** to the BIG/ip Controller. You can configure the BIG/ip Controller to send packets back to the device from which the connection originated when that device is part of a last hop pool of devices associated with a virtual server.

To set up per-connection routing, you must first set up a last hop pool. A last hop pool defines the list of routers as a pool from which the BIG/ip Controller receives packets. For detailed information about setting up a pool, see Chapter 3, *Working with Intelligent Traffic Control*.

The BIG/ip Controller determines the MAC address of the routers when the pool is defined. Then the pool is associated with the virtual server by using the **lasthop** keyword to specify the last hop pool for the virtual server. Then, when a packet arrives for the virtual server, the MAC address that the packet came from is matched up with the MAC address of the members of the last hop pool. The IP address of the matching member is stored with the connection as the last hop address. Then, connections coming from nodes and heading out towards the client are sent to the last hop address, instead of to the default route.

To configure last hop pools for virtual servers from the command line

Use the following syntax to configure last hop pools for virtual servers:

```
bigpipe vip <vip>:<port> lasthop pool <pool_name>
```

For example, you might use the following command:

```
bigpipe vip 192.168.1.10:80 lasthop pool
   secure_routers
```

To configure last hop pools for virtual servers in the F5 Configuration utility

Before you follow this procedure, you must configure at least one pool (for your routers) and one virtual server.

1. In the navigation pane, click Virtual Servers.
   The Virtual Servers screen opens.
2. In the virtual server list, click the virtual server for which you want to set up a last hop pool.
   The properties screen for the virtual server you clicked opens.

3. Click the Last Hop Pool list. Select the pool you created containing your routers.
4. Click the **Apply** button.

Configuring forwarding virtual servers

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

To configure a forwarding virtual server from the command line

Use the following syntax to configure forwarding virtual servers:

```
bigpipe vip <vip>:<port> [ netmask <netmask> ] forward
```

For example, to allow only one service in:

```
bigpipe vip 206.32.11.6:80 forward
```

Use the following command to allow only one server in:

```
bigpipe vip 206.32.11.5:0 forward
```

To forward all traffic:

```
bigpipe vip 0.0.0.0:0 forward
```

To configure forwarding virtual servers in the F5 Configuration utility

1. In the navigation pane, click Virtual Servers.
   The Virtual Servers screen opens.
2. In the toolbar, click the **Add Virtual Server** button.
   The add virtual server screen opens.
   Type in the virtual server attributes including address and port. Use the IP address/port combination guidelines in the previous section, *To configure a forwarding virtual server from the command line*, to determine what these entries should be.

3. In Resources, click the **Forwarding** button.
4. Click the **Apply** button.

Currently, there can be only one wildcard virtual server, whether that is a forwarding virtual server or not. In some of the configurations described here, you need to set up a wildcard virtual server on one side of the BIG/ip Controller to load balance connections across transparent devices. Another wildcard virtual server is required on the other side of the BIG/ip Controller to forward packets to virtual servers receiving connections from the transparent devices and forward them to their destination. You can use another new feature, the transparent device persistence, with forwarding virtual servers, to route connections back through the device from which the connection originated. For more information about per-connection routing, see *Using per-connection routing*, on page 2-29. In these configurations, there you would need to create a forwarding virtual server for each possible destination network or host if a wildcard virtual server is already defined to handle traffic coming from the other direction. For an example of these configuration settings in a network, see *VPN load balancing*, on page 7-17.

Configuring transparent virtual servers

A new option for virtual servers adds the ability to control whether address translation is enabled for a virtual. By default, wildcard virtual servers have translation turned off. A new translate keyword allows you to turn off address translation for non-wildcard virtual servers. 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.

To configure address translation for virtual servers from the command line

Use the following syntax to configure address translation for virtual servers:

```
bigpipe vip <vip>:<port> translate addr [ enable | disable ]
```

For example, use the following syntax to configure address translation for a virtual server 209.43.224.213:80:

```
bigpipe vip 209.43.224.213:80 translate addr disable
```

To configure address translation for virtual servers in the F5 Configuration utility

1. In the navigation pane, click Virtual Servers.
   The Virtual Servers screen opens.
2. In the virtual server list, click the virtual server for which you want to set up a transparent virtual server.
   The properties screen for the virtual server you clicked opens.
3. In the Enable Translation options, clear the **Address** check box. This turns off address translation for the virtual server.
4. Click the **Apply** button.

Using virtual server port translation

A new option for virtual servers adds the ability to control whether port translation is enabled for a virtual server except for wildcard virtual servers. Port translation is turned on by default. An exception to this is if the port defined for a member is port zero. Members with a zero port cannot do translation because zero is not a valid port.

To configure virtual server port translation from the command line

Use the following syntax to configure virtual server port translation:

```
bigpipe vip <vip>:<port> translate port [ enable |
   disable ]
```

For example, if you want to disable port translation for the virtual server/port combination 209.43.224.213:0, type the following command:

```
bigpipe vip 209.43.224.213:0 translate port
   disable
```

To configure virtual server port translation in the F5 Configuration utility

1. In the navigation pane, click Virtual Servers.
   The Virtual Servers screen opens.
2. In the virtual server list, click the virtual server for which you want to set up a transparent virtual server.
   The properties screen for the virtual server you clicked opens.
3. In the Enable Translation options, clear the **Port** check box.
   This turns off port translation for the virtual server.
4. Click the **Apply** button.

Resetting connections on service down

A new option for virtual servers provides the ability to reset connections if a service is down. When this attribute is enabled for a virtual server, the BIG/ip Controller sends resets to the end points of TCP connections when it is determined that the service they are using has gone down.

This is currently only enabled for service checks that mark a node down. Node pings that time out do not cause resets to be sent.

Only TCP connections can receive a Reset. UDP connections are not aborted because there is no shutdown mechanism for UDP connections.

To configure reset connections on service down from the command line

Use the following syntax to reset connections when a service is down:

```
bigpipe vip <vip:port> svcdown_reset [enable |
   disable]
```

To reset connections on service down in the F5 Configuration utility

1. In the navigation pane, click Virtual Servers.
   The Virtual Servers screen opens.
2. In the virtual server list, click the virtual server for which you want to reset connections if the service is down.
   The properties screen for the virtual server you clicked opens.
3. Click the **Enable Reset on Service Down** check box. To disable this feature, clear the Enable Reset on Service Down check box.
4. Click the **Apply** button.

3

Working with Intelligent Traffic Control

- **Introducing Intelligent Traffic Control (ITC)**
- **More flexible load balancing using pools and members**

Introducing Intelligent Traffic Control (ITC)

Intelligent traffic control (ITC) is a set of flexible features that increase the level of service and control over Internet traffic. In these features is the ability to identify specific traffic, based on HTTP request data (URLs, HTTP version, HTTP host field), cookies, or client source address and send that traffic to a specific set of servers or devices that can best service the request. These features let you allocate server resources based on the type of application or content requested most. The following features are ITC features:

- Pools and Members
- Load balancing rules

More flexible load balancing using pools and members

BIG/ip Controller version 3.0 increases load balancing flexibility by introducing the load balancing pool. A load balancing pool is a group of nodes, or other network devices, that are mapped to corresponding virtual servers.

In previous versions of the BIG/ip Controller, the commands used to create a virtual server with a node list resulted in a directly associated, node list "pool" with identical functions. You can still create node list style virtual servers in this version of the BIG/ip Controller. However, pools are more configurable than node lists. Pools may be configured independently and associated with virtual servers in many different ways.

Load balancing pools

A pool is identified by a 1- to 31-character name. Each pool contains its own load balancing method. Also, a pool contains a member list, or list of network devices that handles connections sent to the pool.

You can configure up to 65,000 pools in BIG/ip Controller version 3.0. Adding a new pool also adds a member list with at least one member, and, if needed, the server nodes referenced by the members. You can redefine an existing pool to change the load balancing method, add new members, delete old members, modify member configuration, and reset member statistics. Deleting a pool also deletes the members in the pool. You can view statistics for a pool and its members. You can also reset statistics for an individual pool.

If a virtual server directly references a pool, no pool selection is necessary. If a virtual server references a rule, the rule is evaluated to select a pool (for information about configuring rules, see *Selecting a load balancing pool using a rule (BIG/ip Controller HA versions only)*, on page 3-7). A pool can be referenced by a number of virtual servers or rules, or none at all. A virtual server may reference a single pool, a single rule, or a node list.

◆ Note

If a virtual server uses a pool, the pool may not be deleted from the BIG/ip Controller configuration. Also, a virtual server or a rule cannot be added to the BIG/ip Controller configuration if it references a pool that you have not defined. This means that pools must be added before, and deleted after, the virtual servers or rules that reference them.

Pools are ***independent***, meaning that they can exist in the BIG/ip Controller configuration without being referenced by a virtual server or a rule. If a pool is not directly or indirectly associated with a virtual server, it is not used for load balancing. However, node lists are ***dependent***. This means that they are added as part of

a virtual server, and they are deleted when the virtual server is deleted. Node lists are created by using the standard virtual server syntax to create a virtual server with a node list.

◆ Note

*The **bigpipe pool show** command does not display node list virtual servers.*

Load balancing members

Each member in a member list is a server node or network device to which connections are load balanced. There are three new load balancing methods you can use with pools. These methods are identical to the previous ratio, priority, and least connection methods on the BIG/ip Controller with the exception that the new methods use the configuration, state, and statistics associated with each member in the pool.

BIG/ip Controller, version 3.0 supports three new load balancing methods. You can apply these load balancing options directly in the pools you define. The new options are:

- **ratio_member**
  You can assign ratio weights to each node in the pool with this load balancing method. The ratio weight is the proportion of total connections that the node address should receive. The default ratio weight for a given node address is **1**. If all node addresses use this default weight, the connections are distributed equally among the nodes in the pool.
- **priority_member**
  You can assign priority values to each node in the pool with this load balancing mode. Priority mode is a special type of round robin load balancing. In Priority mode, you define groups of nodes and assign a priority level to each group in a pool. The BIG/ip Controller begins distributing connections in a round robin fashion to all nodes in the highest priority group of the pool. If all the nodes in the highest priority group go **down** or hit a connection limit maximum, the BIG/ip Controller begins to pass connections on to nodes in the next lower priority group in the pool.

- **least_conn_member**
  You can use this load balancing method to load balance connections to the member within the pool that has the least number of current connections. Least Connections mode works best in environments where the servers or other equipment you are load balancing have similar capabilities. Note that current member connections may be less than the current connection of the node to which the member refers in the case where more than one member refers to the same server node.

Defining pools

You can define pools with the command line, or type them directly into the **bigip.conf**. You can also define pools in the web-based F5 Configuration utility. This section describes how to define a pool using each of these configuration methods.

To define a pool in the /etc/bigip.conf

You can define a pool under the server pools section in the **/etc/bigip.conf file**. Figure 3.1 contains the syntax of a pool definition:

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

Figure 3.1 *Pool definition syntax*

For example, the entry for a pool named **my_pool** that uses the **ratio_member** load balancing method, and has two members, might look like this in the **/etc/bigip.conf** (Figure 3.2):

```
pool my_pool {
    lb_mode ratio_member
    member 11.12.1.101:80 ratio 1 priority 1
    member 11.12.1.100:80 ratio 3 priority 1
}
```

Figure 3.2 *An example entry for a pool named* ***my_pool***

It is important to note that the ratio and priority settings in the pool definition apply only for the member load balancing methods: **ratio_member** and **priority_member**.

To define a pool from the command line

To define a pool from the command line, use the following syntax:

```
bigpipe pool <pool_name> {lb_mode <lb_mode_specification> member
   <member_definition> ... member <member_definition>}
```

To add or remove a pool member from the command line

To add or remove a pool member from the command line, use the following syntax:

```
bigpipe pool <pool_name> add | delete {
   <member_definition> [...] }
```

To modify a pool from the command line

To modify a pool from the command line, use the following syntax:

```
bigpipe pool <pool_name> modify {
   <lb_mode_specification> | <member_definition>
   [...]}
```

Each of these elements is described in Table 3.1.

Command line element	Description
<pool_name>	The name you assign to the pool
<lb_mode_specification>	The load balancing method for this pool
member <member_definition>	A member definition. You can include a member definition for each node you want to add to the pool. The member definition consists of the following elements: **`<node_address>:<node_port> [ratio <ratio_value>] [priority <priority_value>]`**

Table 3.1 *The elements that make up a pool*

To configure a pool in the F5 Configuration utility

1. Click Pools in the navigation pane.
   This opens the Pools screen.
2. In the toolbar, click the **Add Pool** button.
   The Add Pool screen opens.
3. In the **Pool Name** box, type in the name you want to use for the pool.
4. Click on the load balancing method list and select the load balancing method you want to use for this pool.
5. Use the resources options to add nodes to the pool. To add a node to the pool, type the IP address in the **Node Address** box, type the port number in the **Port** box, and then type in the ratio or priority for this node. Finally, to add the node to the list, click the add (>>**)** button.
   - **Node Address**
     Type in the IP address of the node you want to add to the pool.
   - **Port**
     Type in the port number of the port you want to use for this node in the pool.

- **Ratio**
  Type in a number to assign a ratio to this node within the pool. For example, if you are using the ratio load balancing method and you type a **1** in this box, the node will have a lower priority in the load-balancing pool than a node marked **2**.
- **Priority**
  Type in a number to assign a priority to this node within the pool. For example, if you are using a priority load-balancing method and you type a **1** in this box, the node will have a lower priority in the load-balancing pool than a node marked **2**.
- **Current Members**
  This is a list of the nodes that are part of the load balancing pool.

6. Click the **Apply** button.

Selecting a load balancing pool using a rule (BIG/ip Controller HA versions only)

With BIG/ip Controller version 3.0, you can create a rule that references two or more load balancing pools. In other words, a rule selects a pool for a virtual server. A rule is referenced by 3- to 31-character name. When a packet arrives that is destined for a virtual server that does not match a current connection, the BIG/ip Controller can select a pool by evaluating a virtual server rule to pick a node pool. The rule is configured to ask true or false questions like:

- HTTP header load-balancing: Does the packet data contain an HTTP request with a URI ending in cgi?
- Client IP address load-balancing: Does the source address of the packet begin with the octet 206?

WARNING

Standard BIG/ip Controller persistence modes, such as application persistence, do not work with rules.

Pool selection based on HTTP request data

The rule tells what should happen when a question is answered true or false. The rule may either select a pool or ask another question. For example, you may want a rule that states if the packet data contains an HTTP request with a URI ending in **cgi**, then load balance using the pool **cgi_pool**. Otherwise, load balance using the pool **default_pool**.

Figure 3.3 shows a rule with an HTTP request variable that illustrates this example:

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

Figure 3.3 *A rule based on an HTTP header variable*

Load balancing normally happens right after the BIG/ip Controller receives a packet that does not match a current connection. However, in the case of an HTTP request, the first packet is a TCP SYN packet that does not contain the HTTP request. In this case, the BIG/ip Controller proxies the TCP handshake with the client and begins evaluating the rule again when the HTTP request is received. When a pool has been selected and a server node selected, the BIG/ip Controller proxies the TCP handshake with the server node and then passes traffic normally.

Pool selection based on client IP address

In addition to the HTTP variables, you can also use the **client_addr** variable to select a pool. You can use this variable to load balance connections based on part of the client's address. For example, you may want a rule that states:

"All client requests with the first byte of their source address equal to **206** will load balance using a pool you create named **clients_from_206** pool. All other requests will load balance using a pool you create named **other_clients_pool**."

Figure 3.4 shows a rule based on the client IP address variable that illustrates this example:

```
rule clients_from_206_rule {
    if ( client_addr equals 206.0.0.0 netmask 255.0.0.0 ) {
       use ( clients_from_206 )
    }
    else {
       use ( other_clients_pool )
    }
}
```

Figure 3.4 *A rule based on the client address variable*

Statements

A rule begins with a statement. An **if** statement allows the conditional evaluation of other statements. A **use** statement selects a load balancing pool.

A statement may do one of three things:

- Ask a true or false question and depending on the answer decide what to do next
- Discard the request
- Use a selected pool for load balancing

The three possible statements expressed in command line syntax are:

```
if (<question>) {<statement>} [else {<statement>}]
discard
use ( <pool_name> )
```

Questions (expressions)

A question or expression is asked by an **if** statement and has a true or false answer. A question or expression has two parts: a predicate, or **operator**, and one or two subjects, or **operands**.

There are two types of subjects (operands); some subjects change and some subjects stay the same.

- Changing subjects are called **variable operands**.
- Subjects that stay the same are called **constant operands**.

A question, or ***expression***, asks questions about variable operands by comparing their current value to constant operands with relational operators.

Constant operands (literals)

Possible constant operands are:

- IP addresses expressed in masked dot notation, for example: **206.0.0.0 netmask 255.0.0.0**
- Strings of ASCII characters, for example: **"pictures/bigip.gif"**
- Regular expression strings

Variable operands (variables)

Since variable operands change their value, they need to be referred to by a constant descriptive name. The variables available depend on the context in which the rule containing them is evaluated. For example, the evaluation of a rule is triggered by the arrival of a packet. Therefore, variables in the rule may refer to features of the triggering packet. In the case of a rule containing questions about an HTTP request, the rule is evaluated in the context of the triggering TCP SYN packet until the first HTTP request question is encountered. After the proxy, the rule continues evaluation in the context of the HTTP request packet, and variables may refer to this packet. Before a variable is compared to the constant in a relational expression, it is replaced with its current value. Possible variable operands are:

- Client request source IP address with the **client_addr** variable.

❖ HTTP request strings (see HREF HTTP request string variables.)

The client_address variable is replaced with an unmasked IP address literal. All HTTP request string variables are replaced with string literals.

Operators

In a rule, relational operators compare two operands to form relational expressions. Possible relational operators and expressions are described in Table 3.2:

Expression	Relational Operator
Are two IP addresses equal?	**`<address> equals <address>`**
Do a string and a regular expression match?	**`<string> matches_regex <regular_expression>`**
Are two strings identical?	**`<string> equals <string>`**
Is the second string a suffix of the first string?	**`<string> ends_with <string>`**
Is the second string a prefix of the first string?	**`<string> starts_with <string>`**

Table 3.2 *The relational operators*

In a rule, logical operators modify an expression or connect two expressions together to form a logical expression. Possible logical operators and expressions are described in Table 3.3:

Expression	Logical Operator
Is the expression not true?	**`not <expression>`**
Are both expressions true?	**`<expression> and <expression>`**
Is either expression true?	**`<expression> or <expression>`**

Table 3.3 *The logical operators*

HTTP request string variables

HTTP request variables are referred to in command line syntax by a predefined set of names. Internally, an HTTP request variable points to a method for extracting the desired string from the current HTTP request header data. Before an HTTP request variable is used in a relational expression, it is replaced with the extracted string. The allowed variable names are:

http_method
The **http_method** is the action of the HTTP request. Common values are **GET** or **POST**.

http_uri
The **http_uri** is the URL, but does not include the protocol and FQDN. For example, if the URL is "http://www.url.com/buy.asp", then the URI is "/buy.asp".

http_version
The **http_version** is the HTTP protocol version string. Possible values are **HTTP/1.0** or **HTTP/1.1**.

http_host
The **http_host** is the value in the **Host:** header of the HTTP request. It indicates the actual FQDN that the client requested. Possible values are a FQDN or a host IP address in dot notation.

http_cookie <cookie name>
The HTTP cookie header is value in the **Cookie:** for the specified cookie name. An HTTP cookie header line can contain one or more cookie name value pairs. The **http_cookie <cookie name>** variable evaluates to the value of the cookie with the name **<cookie name>**.

For example, given a request with the following cookie header line:

```
Cookie: green-cookie=4; blue-cookie=horses
```

The variable **http_cookie blue-cookie** evaluates to the string **horses**. The variable **http_cookie green-cookie** evaluates to the string **4**.

Configuring rules from the command line or in the F5 Configuration utility

You can create rules in the **/etc/bigip.conf** file, from the command line, or with the F5 Configuration utility. Each of these methods is described in this section.

To define a rule from the command line

To define a rule from the command line, use the following syntax:

```
bigpipe rule <pool_name> { <if statement> | <use statement>}
```

For more information about the elements of a rule, see Table 3.5, on page 3-15.

To add a rule in the F5 Configuration utility

1. Click Rules in the navigation pane.
   This opens the Rules screen.
2. In the toolbar, click the **Add Rule** button.
   The Add Rule screen opens.
3. In the **Rule Name** box, type in the name you want to use for the rule.
4. In the Text box, type in a rule. Note that you should not enclose the rule with curly braces { } as you do when you create a rule directly in the **bigip.conf** file.
5. You can type in the rule as an unbroken line, or you can use the Enter key to add line breaks.

Click the **Add** button to add the rule to the BIG/ip Controller configuration.

Configuring virtual servers that reference rules from the command line or in the F5 Configuration utility

In BIG/ip Controller version 3.0, you may use either backward compatible or a new more flexible syntax to define a virtual server. The new syntax allows a rule name in place of the node list.

The possibilities for defining a virtual server that references a rule are:

```
bigpipe vip <virt_serv_key> { <vip_options> <rule_name_reference> }
```

Each of these elements is described in Table 3.4:

Rule element	Description
<virt_serv_key>	A virtual server key definition: `<virtual_address>:<virt_port> [<interface_name>] [unit <ID>]`
<vip_options>	Virtual server options such as IP netmask and broadcast address. For more information, see the ***BIG/ip Controller Reference Guide***, *BIG/pipe Command Reference*.
<rule_name_reference>	A rule name reference. Rule names are strings of 3 to 31 characters. `use rule <rule_name>`

Table 3.4 *The command line rule elements*

Note

You must define a pool before you can define a rule that references the pool.

To configure a virtual server that references a rule in the F5 Configuration utility

1. Click Virtual Severs in the navigation pane.
   The Virtual Servers screen opens.
2. Add the attributes you want for the virtual server such as Address, Port, Unit ID, and Interface.
3. In the Resources section, click **Rule**.
4. In the Rule list, select the rule you want to apply to the virtual server.
5. Click the **Apply** button.

Table 3.5 contains descriptions of all the elements you can use to create rules.

Rule element	Description
Rule definition	`rule { <statement> }`

Statement	`<use_statement>`
	`<if_statement>`
	`discard`

Use statement	`use ( <pool_name> )`
If statement	`if ( <expression> ) { <statement> }` `[ else { <statement> } ]`

Expression	`<literal>`
	`<variable>`
	`( <expression> )`
	`exist <variable>`
	`not <expression>`
	`<expression> <binary_operator> <expression>`

Literal	`<regex_literal>`
	`<string_literal>`
	`<address_literal>`

Regular expression literal	A string of 1 to 63 characters enclosed in quotes that may contain regular expressions
String literal	A string of 1 to 63 characters enclosed in quotes
Address literal	`<dot_notation_longword> [ netmask` `<dot_notation_longword> ]`
Dot notation longword	`<0-255>.<0-255>.<0-255>.<0-255>`

Variable	`http_method`
	`http_version`
	`http_uri`
	`http_host`
	`http_cookie <cookie name>`
	`client_addr`

Binary operator	`or`
	`and`
	`equals`
	`starts_with`
	`ends_with`
	`matches_regex`

Table 3.5 *The elements you can use to construct rules*

A comparison of load balancing configuration from previous versions of the BIG/ip Controller

You can use the method from previous versions of the BIG/ip Controller to define a virtual server with a single node list virtual server. Node lists use the global load balancing method set on the BIG/ip Controller. The global method cannot be set to the **ratio_member**, **priority_member**, or **least_conn_member** load balancing modes. For an example of a node list, see Figure 3.5:

```
lb ratio
vip 15.0.140.1:80 {
    define 12.0.0.44:80 12.0.0.45:80
}
ratio {
    12.0.0.44
} 1
ratio {
    12.0.0.45
} 2
```

Figure 3.5 *The node list method of defining virtual servers*

You can share pools with a number of virtual servers in BIG/ip Controller version 3.0. For example, Figure 3.6 shows a rule shared by a number of virtual servers:

```
pool cgi_pool {
    lb_mode ratio_member
    member 12.0.0.44:80 ratio 1
    member 12.0.0.45:80 ratio 2
}
pool gif_pool {
    lb_mode ratio_member
    member 12.0.0.44:80 ratio 1
    member 12.0.0.45:80 ratio 3
}
rule http_rule {
    if ( http_uri ends_with "gif" ) {
        use ( gif_pool )
    }
    else {
        use ( cgi_pool )
    }
}

vip 15.0.140.1:80 {
    netmask 255.255.0.0 broadcast 15.0.255.255
    use rule http_rule
}
vip 15.0.140.2:80 {
    netmask 255.255.0.0 broadcast 15.0.255.255
    use pool gif_pool
}
```

Figure 3.6 *An example of a rule that references multiple pools*

4

Working with Advanced Service Check Options

- **Introducing advanced service check options**

Introducing advanced service check options

You can use advanced service check options to verify that your content servers are functioning properly. There are two types of advanced service checks: Extended Content Verification (ECV) and Extended Application Verification (EAV). This section describes how to set up, and use, these types of service checking. This section also includes information for setting up EAV service checks for SQL based services.

Setting up ECV service checks for transparent nodes

In addition to verifying content on web servers, you can use Extended Content Verification (ECV) service checks to verify connections to mail servers and FTP servers through transparent nodes. If you want to set up ECV service checks through a transparent node to these types of servers, there are certain special issues that you need to address.

◆ Note

*For information about setting up standard ECV service checks, see the **BIG/ip Controller Getting Started Guide**, Configuring Extended Content Verification service checking, on page 3-29.*

Configuring ECV for transparent nodes

You can set up ECV to verify that a transparent node is functioning properly. To check if a transparent node is functioning, you can add an entry to the **/etc/bigd.conf** file that allows you to retrieve content through the node.

You can use a text editor, such as **vi** or **pico**, to manually create the **/etc/bigd.conf** file, which stores ECV information. To create the entry for checking a transparent node, use the following syntax:

```
transparent <node ip>:<node port> <url> ["recv_expr"]
```

You can also use the following syntax for this entry:

```
transparent <node ip>:<node port> <dest ip>[:<dest port>][/<path>]
   ["recv_expr"]
```

For more information about these configuration entries, please refer to Table 4.1.

Configuration Entry	Description
`transparent`	Transparent is required at the beginning of the entry.
`node ip`	The IP address, in dotted decimal notation, of the transparent firewall or proxy. This IP cannot be a wild card IP (0.0.0.0). Note that the node must be defined as a node in a virtual server definition. Typically this would be a wild card virtual server (0.0.0.0). This entry can also be specified as a fully qualified domain name (FQDN). In order to use an FQDN, the BIG/ip Controller must be configured for name resolution.
`node port`	This entry is the node port to use for the ECV check. This port can be zero. This entry can be numeric or can use a well-known service name, such as **http** or **smtp**.
`dest ip:dest port`	This is the combination of the destination, in dotted decimal notation, and port number of the destination against which the ECV service check is performed. The IP address cannot be a wild card (0.0.0.0). The port number is optional. The port can be specified as any non-zero numeric port number, or specified as a well-known port name, such as **http** or **smtp**.
`url`	The URL is an optional standard HTTP URL. If you do not specify a URL, a default URL is retrieved using the HTTP 1.0 request format. This entry can also be specified using a complete URL with an embedded FQDN. This entry cannot be longer than 4096 bytes. In order to resolve an FQDN, the BIG/ip Controller must be configured for name resolution.
`recv string`	This string is optional. If you specify a string, the string you specify is used to perform standard ECV verification. This entry must be enclosed in quotation marks, and cannot be longer than 128 bytes.

Table 4.1 *Extended content verification configuration entries.*

◆ **Note**

*The **/etc/bigd.conf** file is read once at startup. If you change the file on the command line, you must reboot or restart **bigd** for the changes to take effect. If you make changes in the F5 Configuration utility, clicking the apply button makes changes and restarts **bigd**. See the **BIG/ip Controller Reference Guide**, System Utilities, for details.*

Setting up ECV through transparent nodes with the F5 Configuration utility

New ECV syntax facilitates using ECV with transparent nodes. With it, you can test whether a transparent node is functioning properly by retrieving content through it. You can enable this feature in the F5 Configuration utility or from the command line. This section describes how to enable this feature from the F5 Configuration utility.

◆ Note

You must have at least one wildcard virtual server configured in order to configure ECV through a transparent node.

To set up ECV through a transparent node using the F5 Configuration utility

There are two procedures required to set up ECV through a transparent node. First, set up the frequency and timeout for the port:

1. In the navigation pane, click **Nodes**.
   The Node screen opens.
2. In the **Node** list, click the node you want to configure.
   The Node Properties screen opens.
3. In the **Port** box, click the port for which you want to enable the service check.
   The Global Node Port Properties screen opens.
4. In the **Frequency (seconds)** box, type in the interval (in seconds) at which the BIG/ip Controller performs a service check on the node.
5. In the **Timeout (seconds)** box, type in the time limit (in seconds) that a node has to respond to a service check issued by the BIG/ip Controller.
6. Click the **Apply** button.

After you configure the frequency and timeout settings for the port, set the specific settings for the transparent node:

1. In the navigation pane, click **Nodes**.
   The Node Properties screen opens.
2. In the **Node** list, click the node you want to configure.
   The Node Properties screen opens.
3. In the **Service Check Extended** section, click the ECV button to enable ECV.
4. In the **Type** list, select Transparent.
   By default, the list is set to Transparent.
5. In the **Dest-IP:Dest-Port/url** box, you must type the destination IP address of the node you are checking on the other side of the transparent device. The port number/port name argument is optional. The URL entry is also optional. For more information about what to type in this box, see Table 4.1.
6. In the **Receive String (optional)** box, you can type an ECV check receive string. The receive string is optional.
7. Click the **Apply** button.

Introducing EAV service checks

Extended Application Verification (EAV) is a sophisticated type of service check typically used to confirm whether an application running on a node is responsive to client requests. To determine whether a node application is responsive, the BIG/ip Controller uses a custom program referred to as an ***external service checker***. An external service checker program essentially provides the option to customize service check functionality for the BIG/ip Controller. It is external to the BIG/ip system itself, and is usually developed by the customer. However, the BIG/ip Controller ships with several external service check programs. These include service check programs for FTP, POP3, SMTP, NNTP, and SQL.

You can use an external service checker to verify Internet or intranet applications, such as a web application that retrieves data from a back-end database and displays the data in an HTML page.

An external service checker program works in conjunction with the **bigd** daemon, which verifies node status using node pings and service checks. If you configure external service check on a specific node, the **bigd** daemon checks the node by executing the external service checker program. Once the external service checker executes, the **bigd** daemon looks for output written by the external service checker. If the **bigd** daemon finds output from the external service checker, it marks the node **up**. If it does not find output from the external service checker, it marks the node **down**. Note that **bigd** does not actually interpret output from the external service checker; it simply verifies that the external service checker created output.

◆ Note

Custom external service checker programs are custom programs that are developed either by the customer, or by the customer in conjunction with F5 Networks.

Setting up custom EAV service checks

An ***Extended Application Verification service check*** is a service check that is performed on an application running on a host on the network connected to the BIG/ip Controller. You can create a custom application for this purpose. Complete the following four tasks to implement a custom EAV service check program on the BIG/ip Controller:

- If you use a custom EAV service check program, verify that your external service checker program meets certain requirements, such as creating a **pid** file.
- Install the external service checker program on the BIG/ip Controller.
- Allow EAV service checks in the BIG/ip configuration.
- Configure the specific nodes to use the EAV service check.

Verifying external service checker requirements

Extended Application Verification (EAV) is intended to provide maximum flexibility. The external service checker programs that you create can use any number of methods to determine whether or not a service or an application on a node is responsive. The external service checker must, however, meet the following minimum requirements:

- The external service checker must use a **pid** file to hold its process ID, and the **pid** file must use the following naming scheme:
  `/var/run/pinger.<ip>..<port>.pid.`
- As soon as the external service checker starts, if the **pid** file already exists, the external service checker should read the file and send a **SIGKILL** command to the indicated process.
- The external service checker must write its process ID to the **pid** file.
- If the external service checker verifies that the service is available, it must write standard output. If the external service checker verifies that the service is not available, it cannot write standard output.
- The external service checker must delete its **pid** file before it exits.

The BIG/ip Controller includes a several sample external service checker scripts for HTTP, NNTP, SMTP, and POP3. These scripts can be found in the following location:

`/usr/local/lib/pingers/sample_pinger`

The sample service checker, shown in Figure 4.1, is included with the BIG/ip Controller.

```
# these arguments supplied automatically for all external
pingers:
# $1 = IP (nnn.nnn.nnn.nnn notation or hostname)
# $2 = port (decimal, host byte order)
# $3 and higher = additional arguments
#
# In this sample script, $3 is the regular expression
#

pidfile="/var/run/pinger.$1..$2.pid"

if [ -f $pidfile ]
then
   kill -9 `cat $pidfile` > /dev/null 2>&1
fi

echo "$$" > $pidfile

echo "GET /" | /usr/local/lib/pingers/nc $1 $2 2> /dev/null | \
grep -E -i $3 > /dev/null

status=$?
if [ $status -eq 0 ]
then
    echo "up"
fi
rm -f $pidfile
```

Figure 4.1 *The HTTP external service checker program*

Installing the external service checker on the BIG/ip Controller

To install an EAV service check script, place it in the **/usr/local/lib/pingers** directory. This is the default location for external service checker applications. You can install external service checker applications to other directory locations if desired.

Allowing EAV service checks

Once you install an external service checker on the BIG/ip Controller, you need to add an entry to the **/etc/bigd.conf** file.

To allow external service checking, you need to add the following entry to the **/etc/bigd.conf** file:

```
external  [<node_ip>:]<port> [<path>]<pinger_name> \
   ["<argument_string>"]
```

The **<path>** variable can be an absolute or a relative path to the external checker application. Absolute paths should begin with a slash ("/"). Other paths are relative to the directory default directory, **/usr/local/lib/pingers**. The **<pinger_name>** argument is the name of the pinger script to use for service checking.

The **"<argument_string>"** variable must consist of exactly one string in quotation marks. The string may include any number of arguments, delimited in the usual way by white space, for example:

```
external n1:8000 my_pinger "-a 600 -b"
```

In the above example, the BIG/ip Controller uses HTTP to check port 80, but runs the script **/usr/local/lib/pingers/my_pinger** to check port 8000, with additional arguments.

In the following example, there are three nodes on which the BIG/ip Controller checks port 8000. The BIG/ip Controller runs a separate copy of the external service checker named **my_pinger** for each node:

```
external n1:8000 my_pinger "-a -b"
external 8000 my_pinger "-b"
```

In this example, the first entry specifies how to ping port 8000 on node **n1**. The second entry specifies how to ping port 8000 on any other node.

Command line arguments for EAV service checks

The BIG/ip Controller performs the external service check at specified intervals. The BIG/ip Controller actually uses the service ping interval, which is set using the **bigpipe tping_svc** command.

The external service checker runs as root. The BIG/ip Controller starts an external service checker using the following shell command:

```
<path> <node_ip> <port> [ <additional_argument> ... ]
```

For the case of the example shown above, the appropriate command would be:

```
/usr/local/lib/pingers/my_pinger n1 8000 -a 600 -b
```

The BIG/ip Controller inserts the node IP and port number before the additional arguments that are specified in the **/etc/bigd.conf** file.

Note that the standard input and output of an external service checker are connected to **bigdnode**. The **bigdnode** does not write anything to the external service checker's standard input, but it does read the external service checker's standard output. If **bigdnode** is able to read any data from the external service checker program, the particular service is considered **up**.

Using the EAV pingers bundled with the BIG/ip Controller

The BIG/ip Controller includes a several sample external service checker scripts for HTTP, NNTP, SMTP, POP3, and SQL. These scripts can be found in this location:

```
/usr/local/lib/pingers/
```

The following sections describe how to set up each of these service checkers.

EAV service check for FTP

This section describes how to set up the BIG/ip Controller to perform EAV service checks on FTP services.

The FTP pinger requires three arguments: a full path to the file on any given server, a user name, and a password. Here are example **bigd.conf** entries:

```
external 10.0.0.57:21 /usr/local/lib/pingers/FTP_pinger
   "/pub/demo/file.txt anonymous user@company.com"
```

```
external 10.0.0.62:21 /usr/local/lib/pingers/FTP_pinger
   "/pub/spool/incoming.doc carol carols_password"
```

The FTP pinger attempts to download the specified file to the **/var/tmp** directory. A successful retrieval of any file with the name indicated is considered a successful ping.

To configure the FTP EAV check in the F5 Configuration utility

1. In the navigation pane, click **Nodes**.
   The Node Properties screen opens.
2. In the **Extended** section, click the EAV box to enable EAV service checking. The service check frequency and service check timeout must be set in order to access this option.
3. In the **Type** list, select the FTP service checker. The External Program Path is automatically filled in when you select a pinger from the list.
4. In the **External Program Arguments** box, type in the arguments required for the FTP service checker: a full path to the file on any given server, a user name, and a password. For example:

   `/pub/demo/file.txt anonymous user@company.com`

   `/pub/spool/incoming.doc carol carols_password`
5. Click the **Apply** button.

EAV service check for POP3

This section describes how to set up the BIG/ip Controller to perform EAV service checks on POP3 services.

The POP3_pinger for Post Office Protocol requires only two arguments, a user name and a password. This check is considered successful if it successfully connects to the server, logs in as the indicated user, and logs out again. Here are example **bigd.conf** entries:

```
external 10.0.0.57:109 /usr/local/lib/pingers/POP3_pinger "alice
   alices_password"
external 10.0.0.57:109 /usr/local/lib/pingers/POP3_pinger "bob
   bobs_password"
```

To configure the POP3 EAV check in the F5 Configuration utility

1. In the navigation pane, click **Nodes**.
   The Node Properties screen opens.
2. In the **Extended** section, click the EAV box to enable EAV service checking. The service check frequency and service check timeout must be set in order to access this option.
3. In the **Type** list, select the POP3 service checker. The External Program Path is automatically filled in when you select a service checker from the list.
4. In the **External Program Arguments** box, type in a user name and password. For example:

   ```
   alice alices_password
   bob bobs_password
   ```

5. Click the **Apply** button.

EAV service check for SMTP

This section describes how to set up the BIG/ip Controller to perform EAV service checks on SMTP services.

The SMTP_pinger for mail transport servers requires only one argument, a string identifying the server from which the EAV is originating. This is an extremely simple pinger that checks only that the server is up and responding to commands. It counts a success if the mail server it is connecting to responds to the standard SMTP **HELO** and **QUIT** commands. Here is an example **bigd.conf** entry:

```
external 10.0.0.57:25 /usr/local/lib/pingers/SMTP_pinger
   "bigip@internal.net"
```

To configure the SMTP EAV check in the F5 Configuration utility

1. In the navigation pane, click **Nodes**.
   The Node Properties screen opens.
2. In the **Extended** section, click the EAV box to enable EAV service checking. The service check frequency and service check timeout must be set in order to access this option.
3. In the **Type** list, select the SMTP service checker. The External Program Path is automatically filled in when you select a service checker from the list.
4. In the **External Program Arguments** box, type in a string identifying the server from which the EAV is originating. For example:

   `bigip@internal.net`
5. Click the **Apply** button.

EAV service check for NNTP

This section describes how to set up the BIG/ip Controller to perform EAV service checks on NNTP services.

The NNTP_pinger for Usenet News requires only one argument, a newsgroup name to check for presence. If the NNTP server being queried requires authentication, the user name and password can be provided as additional arguments. This pinger counts a success if it successfully retrieves a newsgroup identification line from the server. Here are example **bigd.conf** entries, the second showing the optional login parameters:

```
external 10.0.0.57:119 /usr/local/lib/pingers/NNTP_pinger
   "comp.lang.java"
external 10.0.0.62:119 /usr/local/lib/pingers/NNTP_pinger
   "local.chat username password"
```

To configure the NNTP EAV check in the F5 Configuration utility

1. In the navigation pane, click **Nodes**.
   The Node Properties screen opens.

2. In the **Extended** section, click the EAV box to enable EAV service checking. The service check frequency and service check timeout must be set in order to access this option.
3. In the **Type** list, select the NNTP service checker. The External Program Path is automatically filled in when you select a service checker from the list.
4. In the **External Program Arguments** box, type in the news group name for which you want to check. If the NNTP server being queried requires authentication, you can provide the user name and password as additional arguments. For example:

   ```
   comp.lang.java
   ```

   ```
   local.chat username password
   ```

5. Click the **Apply** button.

EAV service check for SQL-based services

This section describes how to set up the BIG/ip Controller to perform EAV service checks on SQL-based services such as Microsoft SQL Server versions 6.5 and 7.0, and also Sybase.

The service checking is accomplished by performing an SQL login to the service. If the login succeeds, the service is considered up, and if it fails, the service is considered down. An executable program, **tdslogin** performs the actual login.

1. Test the login manually:

   ```
   cd /usr/local/lib/pingers
   ```

   ```
   ./tdslogin 192.168.1.1 1433 mydata user1
   mypass1
   ```

   Replace the IP address, port, database, user, and password in this example with your own information.

   You should receive the message:

   ```
   Login succeeded!
   ```

   If you receive the connection refused message, verify that the IP and port are correct. See the Troubleshooting SQL based EAV service checks section for more tips.

2. Create and entry in the **/etc/bigd.conf** with the following syntax:

   ```
   external 192.168.1.1:1433
   "/usr/local/lib/pingers/SQL_pinger" "mydata
   user1 mypass1"
   ```

   In this entry, **mydata** is the name of the database, **user1** is the login name, and **mypass1** is the password.

3. Add entries in the **/etc/bigip.conf** for the service checking:

   ```
   tping_svc 1433 5
   ```

   ```
   timeout_svc 1433 15
   ```

4. Reload the **/etc/bigip.conf** and restart **bigd**:

   ```
   bigpipe -f /etc/bigip.conf
   ```

   ```
   bigd
   ```

5. Verify that the service check is being performed correctly: If the service is **UP**, change the password in **/etc/bigd.conf** to an invalid password and restart **bigd**. The service should go down after the timeout period elapses.

   Correct the password and restart **bigd** and the service should go up again.

Troubleshooting SQL-based service checks

If you are having trouble, you should verify that you can login using another tool. For example, if you have Microsoft NT SQL Server version 6.5, there is a client program **ISQL/w** included with the SQL software. This client program performs simple logins to SQL servers. Use this program to test whether you can login using the ISQL/w program before attempting logins from the BIG/ip Controller.

Creating a test account for Microsoft SQL

On the SQL server, you can run the SQL Enterprise Manager to add logins. When first entering the SQL Enterprise Manager, you may be prompted for the SQL server to manage.

You can register servers by entering the machine name, user name, and password. If these names are correct, the server will be registered and you will be able to click on an icon for the server. When you expand the subtree for the server, there will be an icon for Logins.

Underneath this subtree, you can find the SQL logins. Here, you can change passwords or add new logins by right-clicking on the Logins icon. Click this icon to open an option to **Add login**. After you open this option, enter the user name and password for the new login, as well as which databases the login is allowed to access. You must grant the test account access to the database you specify in the EAV configuration.

5

Working with Advanced Persistence Options

- **Introducing advanced persistence options**

Introducing advanced persistence options

In addition to the simple persistence and SSL persistence options provided by the BIG/ip Controller, several advanced persistence options are available. These options include:

- HTTP Cookie persistence
- Destination Address Affinity (Sticky persistence)
- Persist masking
- Maintaining persistence across virtual servers with the same address
- Maintaining persistence across all virtual servers

Note

This chapter describes advanced peristence options. For more information about SSL persistence and simple persistence, see the ***BIG/ip Controller Getting Started Guide****, Configuring persistence for e-commerce and other dynamic content sites, page 3-36.*

Using HTTP cookie persistence

You can set up the BIG/ip Controller to use HTTP cookie persistence. This method of persistence uses an HTTP cookie stored on a client's computer to allow the client to reconnect to the same server previously visited at a web site. This method of persistence can be used only with unencrypted HTTP 1.0 or 1.1 communication. This means cookie persistence does not work with SSL traffic.

There are four types of cookie persistence available: Hash mode, Insert mode, Rewrite mode, and Passive mode. The mode you choose affects how the cookie is handled by the BIG/ip Controller when it is returned to the client.

Hash mode

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.

To configure the hash cookie persistence option from the command line

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

```
bigpipe vip <vip:port> define <pool_or_nodes> special cookie hash
   <CookieName> <Offset> <Length>
```

The <**CookieName**>, <**Offset**>, and <**Length**> values are described in Table 5.1:

Hash mode values	Description
`<CookieName>`	This is the name of an HTTP cookie being set by a Web site.
`<Offset>`	This is the number of bytes in the cookie to skip before calculating the hash value.
`<Length>`	This is the number of bytes to use when calculating the hash value.

Table 5.1 *The cookie hash mode values*

To configure the cookie persistence hash option in the F5 Configuration utility

Before you follow this procedure, you must configure at least one virtual server.

1. In the navigation pane, click Virtual Servers.
   The Virtual Servers screen opens.

2. In the virtual server list, click the virtual server for which you want to set up hash mode persistence.
   The properties screen for the virtual server you clicked opens.
3. In the **Application Persistence** box, click **None**. The Application Persistence screen opens.
4. Click the **HTTP Cookie Persistence** button.
   Set the following values:
   - **Method** Click the list and select **Hash**.
   - **Timeout** The timeout value is not used with hash mode.
   - **Cookie Name (Hash Method)** Type in the name of an HTTP cookie being set by the Web site. This could be something like Apache or SSLSESSIONID. It depends on the type of web server your site is running.
   - **Hash Values** The **Offset** is the number of bytes in the cookie to skip before calculating the hash value. The Length is the number of bytes to use when calculating the hash value.
5. Click the **Apply** button.

Insert mode

If you specify Insert mode, the information about the server to which the client connects is written in the header of the HTTP response from the server. This mode creates a cookie, named **BIGipServer**, and inserts it in the HTTP response from the server that contains the information about the chosen server. The expiration date for the cookie is set based on the time-out configured on the BIG/ip Controller.

Rewrite mode

If you specify Rewrite mode, the BIG/ip Controller intercepts a cookie, named **BIGipCookie**, sent from the server to the client and rewrites the name of the cookie to **BIGipServer**. When the BIG/ip Controller rewrites the cookie, the server information and time-out value are reset.

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 BIG/ip 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=000000000000000000000000000000000
    00000000...
```

(The cookie should contain a total of 75 zeros.)

Passive mode

If you specify Passive mode, the BIG/ip Controller does not insert or search for existing 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 time-out.

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=184658624.20480.000; expires=Sat,
    19-Aug-2000 19:35:45 GMT; path=/
```

In this example, **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.

To activate Insert, Rewrite, or Passive HTTP cookie persistence in the F5 Configuration utility

1. In the navigation pane, click Virtual Servers.
   The Virtual Servers Properties screen opens.

2. In the **Application Persistence** box, click **None**. The Application Persistence screen opens.
3. Select HTTP Cookie Persistence.
4. Select the mode you want to use: Hash, Insert, Rewrite, or Passive. Each mode handles the cookie in a different manner (see the explanations preceding).
5. Select the time-out value in days, hours, minutes, and seconds. This value determines how long the cookie lives on the client computer before it expires.
6. Click **Apply**.

To activate Insert, Rewrite, or Passive HTTP cookie persistence from the command line

To activate HTTP cookie persistence from the command line, use the following syntax:

```
bigpipe vip <virt addr>:<service> define <node addr>:<service>
   [...<node addr>:<service>] special cookie <mode name> <timeout>
```

For the **`<mode name>`**, type Insert, Rewrite, Passive, or Hash. The **`<timeout>`** value applies only for Insert and Rewrite modes. The **`<timeout>`** value for the cookie is written using the following format:

`<days>d hh:mm:ss`

Using destination address affinity (sticky persistence)

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

This enhancement provides the most benefits when load balancing caching proxy servers. A caching proxy server intercepts web requests and returns a cached web page if it is available. In order to improve the efficiency of the cache on these proxies, it is necessary to send similar requests to the same proxy server repeatedly. Destination address affinity can be used to cache a given web page

on one proxy server instead of on every proxy server in an array. This saves the other proxies from having to duplicate the web page in their cache, wasting memory.

WARNING

In order to prevent sticky entries from clumping on one server, use a static load balancing mode, such as Round Robin.

To activate destination address affinity in the F5 Configuration utility

You can only activate destination address affinity on wildcard virtual servers. For information on setting up a wildcard virtual server, see the ***BIG/ip Getting Started Guide***, *Defining wildcard virtual servers, on page 3-8*. Follow these steps to configure destination address affinity:

1. In the navigation pane, click Virtual Servers.
   The Virtual Servers screen opens.
2. Click the wildcard virtual server you want to configure.
   The Virtual Server Properties screen opens.
3. Click the **Destination Address Affinity Enable** box to enable destination address affinity.
4. In **Destination Address Affinity Mask** box, type in the mask you want to apply to sticky persistence entries.
5. Click the Apply button.

To activate sticky persistence from the command line

Use the following command to enable sticky persistence for the virtual server:

```
bigpipe vip 0.0.0.0:<port> sticky enable
```

Use the following command to disable sticky persistence for the virtual server:

```
bigpipe vip 0.0.0.0:<port> sticky disable
```

Use the following command to show whether the sticky persistence is enabled or disabled for the virtual server:

bigpipe vip 0.0.0.0:<port> sticky show

Use the following command to list sticky persistence entries for the specified virtual server.

bigpipe vip 0.0.0.0:<port> sticky dump

Use the following command to delete sticky entries for the specified virtual server:

bigpipe vip 0.0.0.0:<port> sticky clear

Use the following command to define the sticky mask for the virtual server:

bigpipe vip 0.0.0.0:<port> sticky mask <mask>

For example <**mask**> could be 255.255.255.0. To remove the sticky mask for the virtual server:

bigpipe vip 0.0.0.0:<port> sticky mask none

To show the sticky mask for the virtual server:

bigpipe vip 0.0.0.0:<port> sticky mask show

To clear all sticky connections on a BIG/ip Controller, issue the following **bigpipe** command:

bigpipe vip sticky clear

Using persist mask on a virtual server

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

◆ Note

You cannot use persist masks on virtual servers that use rules.

Applying a persist mask

The complete syntax for the **bigpipe vip persist mask** command is:

```
bigpipe vip <virt addr>:<port> persist mask <ip> | none | show
```

Use the following syntax to specify a range of IP addresses to be included in persistence of the specified virtual port. The command adds a persist mask to a port:

```
bigpipe vip <virt addr>:<port> persist mask <ip>
```

For example, the following command would keep persistence information together for all clients within a C class network that connect to the virtual server 10.10.10.90:80:

```
bigpipe vip 10.10.10.90:80 persist mask 255.255.255.0
```

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

```
bigpipe vip 10.10.10.90:80 persist mask none
```

To display all persist masks, use the **show** option:

```
bigpipe vip 10.10.10.90:80 persist mask show
```

Maintaining persistence across virtual servers that use the same virtual addresses

The BIG/ip Controller platform provides a similar persistence mode that is less granular. The BIG/ip Controller can maintain persistence for all connections requested by the same client, as long as the virtual server hosting each request uses the same virtual address. When this mode is turned on, the BIG/ip Controller attempts to send all persistent connection requests received from the same client, within the persistence time limit, to the same node only when the virtual server hosting the connection has the same virtual address as the virtual server hosting the initial persistent connection. Connection requests from the client that go to other virtual servers with different virtual addresses, or those connection requests that do not use persistence, are load balanced according to the currently selected load balancing mode.

Using the preceding example, if a BIG/ip Controller configuration includes the following virtual server mappings, where each virtual server uses persistence:

```
bigpipe vip v1:http define n1:http n2:http
bigpipe vip v1:ssl  define n1:ssl  n2:ssl
```

For example, a client makes an initial connection to **v1:http** and the BIG/ip Controller's load balancing mechanism chooses **n1:http** as the node. If the same client then connects to **v1:ssl**, the BIG/ip Controller starts tracking a new persistence session, and it uses the load balancing mode to determine which node should receive the connection request because the requested virtual server uses a different virtual address (**v1**) than the virtual server hosting the first persistent connection request (**v1**). However, if the client subsequently connects to **v1:ssl**, the BIG/ip Controller uses the persistence session established with the first connection to determine the node that should receive the connection request, rather than the load balancing mode. The BIG/ip Controller should send the third connection request to **n1:ssl**, which uses the same node address as the **n1:http** node that currently hosts the client's first connection with which it shares a persistent session.

WARNING

In order for this mode to be effective, virtual servers that use the same virtual address, as well as those that use TCP or SSL persistence, should include the same node addresses in the virtual server mappings.

The system control variable **bigip.persist_on_any_port_same_vip** turns this mode on and off. To activate the persistence mode, type:

```
sysctl -w bigip.persist_on_any_port_same_vip=1
```

To deactivate the persistence mode, type:

```
sysctl -w bigip.persist_on_any_port_same_vip=0
```

Maintaining persistence across all virtual servers

You can set the BIG/ip Controller to maintain persistence for all connections requested by the same client, regardless of which virtual server hosts each individual connection initiated by the client. When this mode is turned on, the BIG/ip Controller attempts to send all persistent connection requests received from the same

client, within the persistence time limit, to the same node. Connection requests from the client that do not use persistence are load balanced according to the currently selected load balancing mode.

For example, if a BIG/ip Controller configuration includes the following virtual server mappings, where each virtual server uses persistence:

```
bigpipe vip v1:http define n1:http n2:http
bigpipe vip v1:ssl define n1:ssl  n2:ssl
bigpipe vip v2:http define n1:http n2:http
bigpipe vip v2:ssl  define n1:ssl  n2:ssl
```

Say that a client makes an initial connection to **`v1:http`** and the BIG/ip Controller's load balancing mechanism chooses **`n1:http`** as the node. If the same client subsequently connects to **`v1:ssl`**, the BIG/ip Controller would send the client's request to **`n1:ssl`**, which uses the same node address as the **`n1:http`** node that currently hosts the client's initial connection.

WARNING

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

The system control variable **bigip.persist_on_any_vip** turns this mode **on** and **off**. To activate the persistence mode, type:

```
sysctl -w bigip.persist_on_any_vip=1
```

To deactivate the persistence mode, type:

```
sysctl -w bigip.persist_on_any_vip=0
```

6

Working with Advanced Redundant System Features

- **Introducing advanced redundant system features**

Introducing advanced redundant system options

In addition to the simple redundant features available on the BIG/ip Controller, several advanced redundant features are available. Advanced redundant system features provide additional assurance that your content is available if a BIG/ip Controller experiences a problem. These advanced redundant system options include:

- Mirroring connection and persistence information
- Gateway fail-safe
- Network-based fail-over
- Setting a specific BIG/ip Controller to be the active controller
- Setting up active-active redundant controllers

Mirroring connection and persistence information

When the fail-over process puts the active controller duties onto a standby controller, the connection capability of your site returns so quickly that it has little chance to be missed. By preparing a redundant system for the possibility of fail-over, you effectively maintain your site's reliability and availability in advance. But fail-over alone is not enough to preserve the connections and transactions on your servers at the moment of fail-over; they would be dropped as the active controller goes down unless you have enabled ***mirroring***.

The mirror feature on BIG/ip Controllers is a specialized, ongoing communication between the active and standby controllers that duplicates the active controller's real-time connection or persistence information state on the standby controller. If mirroring has been enabled, fail-over can be seamless to such an extent that file transfers can proceed uninterrupted, customers making orders can complete transactions without interruption, and your servers can generally continue with whatever they were doing at the time of fail-over.

The mirror feature is intended for use with long-lived connections, such as FTP, Chat, and Telnet sessions. Mirroring is also effective for persistence information.

WARNING

If you attempt to mirror all connections, the performance of the BIG/ip Controller may degrade.

Commands for mirroring

Table 6.1 contains the commands that support mirroring capabilities. For complete descriptions, syntax, and usage examples, see the ***BIG/ip Controller Reference Guide***, *BIG/pipe Command Reference*.

BIG/pipe command	Options
`bigpipe mirror`	Options for global mirroring
`bigpipe vip mirror`	Options for mirroring connection and persistence information on a virtual server.
`bigpipe snat mirror`	Options for mirroring secure NAT connections

Table 6.1 *Mirroring command in BIG/pipe*

Global mirroring on the BIG/ip Controller redundant system

You should enable mirroring on a redundant system at the global level before you can set mirroring of any specific types of connections or information. However, you can set specific types of mirroring and then enable global mirroring to begin mirroring. The syntax of the command for setting global mirroring is:

```
bigpipe mirror enable | disable | show
```

To enable mirroring on a redundant system, use the following command:

```
bigpipe mirror enable
```

To disable mirroring on a redundant system, use the following command:

```
bigpipe mirror disable
```

To show the current status of mirroring on a redundant system, use the following command:

```
bigpipe mirror show
```

Mirroring virtual server state

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

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

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

```
bigpipe vip <virt addr>:<port> mirror [ persist | conn ] \
     enable | disable
```

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

```
bigpipe vip <virt addr>:<port> mirror [ persist | conn ] show
```

If you do not specify either **persist**, for persistent information, or **conn**, for connection information, the BIG/ip Controller assumes that you want to display both types of information.

Mirroring SNAT connections

SNAT connections are mirrored only if specifically enabled. You can enable SNAT connection mirroring by specific node address, and also by enabling mirroring on the default SNAT address. Use the following syntax to enable SNAT connection mirroring on a specific address:

```
bigpipe snat <node addr> [...<node addr>] mirror enable | disable
```

In the following example, the **enable** option turns on SNAT connection mirroring to the standby controller for SNAT connections originating from 192.168.225.100.

```
bigpipe snat 192.168.225.100 mirror enable
```

Use the following syntax to enable SNAT connection mirroring the default SNAT address:

```
bigpipe snat default mirror enable | disable
```

Using gateway fail-safe

Fail-safe features on the BIG/ip Controller provide network failure detection based on network traffic. Gateway fail-safe monitors traffic between the active controller and the gateway router, protecting the system from a loss of the internet connection by triggering a fail-over when the gateway is unreachable for a specified duration.

You can configure gateway fail-safe in the F5 Configuration utility or in BIG/db. If you configure gateway fail-safe in BIG/db, you can toggle it on and off with **bigpipe** commands.

Adding a gateway fail-safe check

You can set up a gateway fail-safe check using the F5 Configuration utility:

- the name or IP address of the router (only one gateway can be configured for fail-safe)
- the time interval (seconds) between pings sent to the router
- time-out period (seconds) to wait for replies before proceeding with fail-over

To configure gateway fail-safe in the F5 Configuration utility

1. Click the BIG/ip logo in the navigation pane.
   The BIG/ip System Properties screen opens.

2. In the Gateway Fail-safe section of the screen, make the following entries:
   - Click the **Enabled** box.
   - In the **Router** box, type the IP address of the router you want to ping.
   - In the **Ping (seconds)** box, type the interval, in seconds, you want the BIG/ip Controller to wait before it pings the router.
   - In the **Timeout (seconds)** box, type the timeout value, in seconds. If the router does not respond to the ping within the number of seconds specified, the gateway is marked **down**.
3. Click the **Apply** button.

To configure gateway fail-safe in BIG/db

To enable gateway fail-safe in BIG/db, you need to change the settings of three specific BIG/db database keys using the **bigdba** utility. The keys set the following values:

❖ The IP address of the router

❖ The ping interval

❖ The timeout period

To set these keys, type this command to open the BIG/db database:

```
bigdba
```

To set the IP address of the router, type the following entry, where **<gateway IP>** is the IP address, or host name, of the router you want to ping:

```
Local.Bigip.GatewayPinger.Ipaddr=<gateway IP>
```

To set the ping interval, type the following entry, where **<seconds>** is the number of seconds you want the BIG/ip Controller to wait before pinging the router:

```
Local.Bigip.GatewayPinger.Pinginterval=<seconds>
```

To set the timeout, type the following entry, where **<seconds>** is the number of seconds you want the BIG/ip Controller to wait before marking the router **down**:

```
Local.Bigip.GatewayPinger.Timeout=<seconds>
```

To close **bigdba** and save your changes, type this command and press the Enter key:

```
quit
```

For more information about BIG/db and using **bigdba**, see *Working with the BIG/db database*, on page 8-24.

◆ Note

*After you make these changes, you must restart **bigd** to activate the gateway pinger.*

Enabling gateway fail-safe

Gateway fail-safe monitoring can be toggled **on** or **off** from the command line using the **bigpipe gateway** command.

For example, arm the gateway fail-safe using the following command:

```
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
```

Gateway fail-safe messages

The destination for gateway fail-safe messages is set in the standard syslog configuration (**/etc/syslog.conf**), which directs these messages to the file **/var/log/bigd**. Each message is also written to the BIG/ip Controller console (**/dev/console**).

Using network-based fail-over

Network-based fail-over allows you to configure your redundant BIG/ip Controller to use the network to determine the status of the active controller. Network-based fail-over can be used in addition to, or instead of, hard-wired fail-over.

To configure network fail-over in the F5 Configuration utility

1. Click the BIG/ip logo in the navigation pane.
   The BIG/ip System Properties screen opens.
2. In the Redundant Configuration section of the screen, make the following entry:
   Click the **Network Failover Enabled** box.
3. Click the **Apply** button.

To Configure network-based fail-over in BIG/db

To enable network-based fail-over, you need to change the settings of specific BIG/db database keys using the **bigdba** utility. To enable network-based fail-over, the **Common.Sys.Failover.Network** key must be set to one (**1**). To set this value to one, type this command to open the BIG/db database:

```
bigdba
```

At the **bigdba** prompt, type the following entry:

```
Common.Sys.Failover.Network=1
```

To close **bigdba** and save your changes, type this command and press the Enter key:

```
quit
```

Other keys are available to lengthen the delay to detect the fail-over condition on the standby controller, and to lengthen the heart beat interval from the active unit. To change the time required for the standby unit to notice a failure in the active unit, set the following value using the **bigdba** utility (the default is three seconds):

```
Common.Bigip.Cluster.StandbyTimeoutSec=<value>
```

To change the heart beat interval from the active BIG/ip Controller, change the following value using **bigdba** (the default is one second):

```
Common.Bigip.Cluster.ActiveKeepAliveSec=<value>
```

For more information about BIG/db and using **bigdba**, see *Using bigdba*, on page 8-25.

Setting a specific BIG/ip Controller to be the preferred active unit

Setting a preferred active controller means overlaying the basic behavior of a BIG/ip Controller with a preference toward being active. A controller that is set as the active controller becomes active whenever the two controllers negotiate for active status.

To clarify how this differs from default behavior, contrast the basic behavior of a BIG/ip Controller in the following description. Each of the two BIG/ip Controllers in a redundant system has a built-in tendency to try to become the active controller. Each system attempts to become the active controller at boot time; if you boot two BIG/ip Controllers at the same time, the one that becomes the active controller is the one that boots up first. In a redundant configuration, if the BIG/ip Controllers are not configured with a preference for being the active or standby controller, either controller can become the active controller by becoming active first.

The active or standby preference for the BIG/ip Controller is defined by setting the appropriate startup parameters for **sod** (the switch over daemon) in **/etc/rc.local**. For more details on **sod** startup and functioning, see the ***BIG/ip Controller Reference Guide***, *System Utilities.*

The following example shows how to set the controller to standby:

```
echo " sod.";    /usr/sbin/sod -force_slave 2> /dev/null
```

A controller that prefers to be standby can still become the active controller if it does not detect an active controller. It is not possible for both controllers in a redundant system to be in active mode or in standby mode at the same time.

This example shows how to set controller to active:

```
echo " sod.";    /usr/sbin/sod -force_master 2> /dev/null
```

A controller that prefers to be active can still serve as the standby controller when it is on a live redundant system that already has an active controller. For example, if an active controller that preferred to be active failed over and was taken out of service for repair, it could then go back into service as the standby controller until the next time the redundant system needed an active controller, for example, at reboot.

Active-active redundant controllers

You can use the active-active feature to simultaneously load balance traffic for different virtual addresses on redundant BIG/ip Controllers. Performance improves when both BIG/ip Controllers are in active service at the same time. In active-active mode, you configure virtual servers to be served by one of the two controllers. If one controller fails, the remaining BIG/ip Controller assumes the virtual servers of the failed machine. For this configuration to work, each controller has its own unit ID number. Each virtual server, NAT, or SNAT you create includes a unit number designation that determines which active controller handles its connections.

◆ Note

If you do not want to use this feature, redundant BIG/ip Controllers operate in active/standby mode by default.

◆ WARNING

MAC masquerading is not supported in active-active mode.

Configuring an active-active system

The default mode for BIG/ip Controller redundant systems is active/standby. You must take several steps in order to use active-active mode on the redundant BIG/ip Controller system. Details follow this brief list.

1. Configure an additional shared IP alias on the internal interface for each unit. You must have two shared aliases for the redundant system.
2. Set the routing configuration on the servers load balanced by the active-active BIG/ip Controller system.
3. Make sure the BIG/db key **Local.Bigip.Failover.UnitId** is 1 for one of the controllers, and 2 for the other.
4. Enable active-active mode by setting the BIG/db key **Common.Bigip.Failover.ActiveMode** to **1**.
5. Define the virtual servers, NATs, and/or SNATs to run on either unit 2 or on unit 1.
6. Update the fail-over daemon (**/sbin/sod**) with the configuration changes made in BIG/db.
7. Synchronize the configuration.
8. Transition from active/standby to active-active.

◆ Note

We recommend making all of these configuration changes on one controller and then synchronizing the configuration.

Step 1: Configure an additional shared IP alias

When you configure a redundant system, you enter a shared IP alias. In active/standby mode, this shared IP alias runs on the active controller. You can determine if you already have a shared IP alias by running the **bigpipe interface** command. If you have one, it is probably configured as belonging to unit one.

In an active-active configuration, each BIG/ip Controller must have a shared IP alias on the internal, source processing, interface. This is the address to which the servers behind the BIG/ip Controller route traffic. Since you already have a shared IP alias for one controller, add a shared IP alias for the other controller by using the **bigpipe ipalias** command. For example:

```
bigpipe ipalias exp1 172.20.10.2 netmask
    255.255.0.0 unit 2
```

If you do not have a shared IP alias for unit 1, add one using this command. To view the IP aliases for the controller, type the **bigpipe interface** command on the command line.

If the BIG/ip Controller fails over, its shared IP address is assumed by the remaining unit and the servers continue routing through the same IP address.

You can configure additional shared IP aliases on an external, destination processing, interface of each BIG/ip Controller, as well. This makes it possible for routers to route to a virtual server using vip noarp mode.

To configure the additional shared IP alias in the F5 Configuration utility

1. On the NICs screen, click the name of the interface you want to configure.
   The Network Interface Card properties screen opens. You must choose an internal (source processing) interface.
2. In the Redundant Configuration section, check for a Unit 1 Alias and a Unit 2 Alias.
3. If one of the unit aliases is not present, type in an alias for the unit.
4. Click the **Apply** button.

Repeat this procedure on the other controller or use the Sync Configuration option in the tool bar of the BIG/ip System Properties page. Note that these settings should be identical on both controllers.

Step 2: Configuring servers for active-active

The active-active feature causes some restrictions on the servers behind the BIG/ip Controllers. The servers must be logically segregated to accept connections from one BIG/ip Controller or the other. To do this, set the default route to the BIG/ip Controller IP alias (see Step 1) from which it accepts connections. In the case of a fail-over, the surviving BIG/ip Controller assumes the internal IP alias of the failed machine, providing each server a default route.

Step 3: Check the BIG/ip Controller unit number

Using the **bigdba** utility, check the value of the BIG/db key **Local.Bigip.Failover.UnitId**. This value should be 1 for one of the controllers, and 2 for the other.

Each BIG/ip Controller in an active-active configuration requires a unit number: either a 1 or a 2. The First-Time Boot utility allows a user to specify a unit number for each BIG/ip Controller. In an active-active configuration, specify the unit number when you configure virtual addresses, NATs, and SNATs.

◆ Note

You can only set this value directly in BIG/db. It cannot be set in the F5 Configuration utility.

To check the BIG/ip Controller unit number in the F5 Configuration utility

Follow this procedure on each BIG/ip Controller in a redundant system to check the BIG/ip Controller unit number with the F5 Configuration utility:

1. Open the F5 Configuration utility.
2. In the navigation pane, check the description next to the BIG/ip logo.
   The status of the controller is Active and the unit number is either 1 or 2.

Step 4: Active-active BIG/db configuration parameters

To enable active-active, you must set the **Common.Bigip.Failover.ActiveMode** key to one (1). To set this value to one, follow these steps:

Type the following command to open the BIG/db database:

```
bigdba
```

At the **bigdba** prompt, type the following entry:

```
Common.Bigip.Failover.ActiveMode=1
```

Type **quit** to exit BIG/db and save the configuration.

The default for this entry is **off** and fail-over runs in active/standby mode.

To enable active-active in the F5 Configuration utility

Perform this procedure on the active controller first. After the active box is enabled, follow this procedure on the standby controller. After you perform this feature on the standby controller, wait 30 seconds and click the **Refresh** button (Microsoft IE) or **Reload** button (Netscape Navigator) on the browser for both controllers.

1. In the navigation pane, click the BIG/ip logo.
2. The BIG/ip Controller System Properties screen opens.
3. Click the **Active-Active Mode Enabled** check box.
4. Click the **Apply** button.

Step 5: Virtual address configuration

Both BIG/ip Controllers must have the exact same configuration file (**bigip.conf**). When a virtual server is defined, it must be defined with a unit number that specifies which BIG/ip Controller handles connections for the virtual server. Each BIG/ip Controller has a unit number, 1 or 2, and serves the virtual servers with corresponding unit numbers. If one of the BIG/ip Controllers fails over, the remaining BIG/ip Controller processes the connections for virtual servers for both units.

Defining virtual servers, NATs, and SNATs on active-active controllers

Use the following commands to define virtual servers, NATs, and SNATs on active-active controllers:

```
bigpipe vip <virt addr>:<port> define [unit <1|2>]
   <node addr>:<port>
bigpipe nat <internal_ip> to <external_ip> ... [unit <1|2>]
bigpipe snat map <orig_ip> to <trans_ip> ... [unit <1|2>]
```

Note

If not specified, the unit number defaults to 1.

Each BIG/ip Controller in an active-active configuration requires a unit number: either a 1 or a 2. Use the First-Time Boot utility to specify a unit number for each BIG/ip Controller. If you do not specify a unit number, the unit number for the virtual server defaults to 1.

Note

You must specify the unit number when defining virtual servers, NATs, and SNATs. You cannot add the unit number at a later time without redefining the virtual server, NAT, or SNAT.

To define virtual servers, NATs, and SNATs on active-active controllers in the F5 Configuration utility

The following example illustrates the unit ID number in a virtual server definition. Although the steps to create a NAT or SNAT are slightly different, the unit ID number serves the same purpose.

1. In the navigation pane, click Virtual Servers.
   The Virtual Servers screen opens.
2. In the toolbar, click the **Add Virtual Server** button.
3. Type in the address, netmask, and port for the virtual server.
4. Click the **Unit ID** list. Select the unit number for the virtual server.

5. The connections served by this virtual server are managed by the controller assigned this unit ID.
6. Complete the Resources section of the screen. For more information about individual settings, refer to the online help.
7. Click the **Apply** button.

Step 6: Update the fail-over daemon (/sbin/sod) with the configuration changes made in BIG/db

Active-active mode is implemented by the fail-over daemon. If you change a BIG/db key that affects the fail-over daemon (when Failover is part of the key) the fail-over daemon needs to be updated with the change. To update the fail-over daemon, type the following command:

```
bigpipe failover init
```

Step 7: Synchronize the configuration

After you complete steps 1 through 5 on each controller in the active-active system, synchronize the configurations on the controllers with the following command:

```
bigpipe configsync all
```

To synchronize the configuration in the F5 Configuration utility

1. In the navigation pane, click the BIG/ip logo.
   The BIG/ip Properties screen opens.
2. In the toolbar, click the **Sync Configuration** button.
   The Synchronize Configuration screen opens.
3. Click the **Synchronize** button.

Step 8: Transition from active/standby to active-active

To transition from active/standby to active-active, type the following command on the active BIG/ip Controller:

```
bigpipe failover standby
```

This command puts the active BIG/ip Controller into partial active-active mode. To complete the transition, type in the following command on the other BIG/ip Controller which now considers itself the active unit

```
bigpipe failover standby
```

Now both units are in active-active mode.

To transition from active/standby to active-active in the F5 Configuration utility

This step is not required if you enable active-active in the F5 Configuration utility. The transition is made during *Step 4: Active-active BIG/db configuration parameters*, on page 6-13.

Active-active system fail-over

Before a failure in an active-active installation, one BIG/ip Controller is servicing all requests for virtual servers configured on unit 1, and the other BIG/ip Controller is servicing all requests for virtual servers configured on unit 2. If one of the BIG/ip Controllers fails, the remaining BIG/ip Controller handles all requests for virtual servers configured to run on unit 1 and also those configured to run on unit 2. In other words, the surviving BIG/ip Controller is acting as both units 1 and 2.

If the BIG/ip Controller that failed reboots, it re-assumes connections for the unit number with which it was configured. The BIG/ip Controller that was running as both units stops accepting connections for the unit number that has resumed service. Both machines are now active.

When the unit that was running both unit numbers surrenders a unit number to the rebooted machine, all connections are lost that are now supposed to run on the rebooted machine. (The connections are lost unless they were mirrored connections.)

Disabling automatic fail back

In some cases, you may not want connections to automatically fail back. The fact that a machine has resumed operation may not be reason enough to disrupt connections that are running on the BIG/ip Controller serving as both units. Note that because of addressing issues, it is not possible to slowly drain away connections from the machine that was running as both units, giving new requests to the recently rebooted machine.

To disable automatic fail back, set the BIG/db key **Common.Bigip.Failover.ManFailBack** to 1. When you set this key to **1**, a BIG/ip Controller running as both units does not surrender a unit number to a rebooted peer until it receives the **bigpipe failover failback** command. By default, this key is not set.

Taking an active-active controller out of service

You can use the bigpipe failover standby command to place an active controller in standby mode. In active-active mode, type the following command to place a one of the active controllers in standby mode:

```
bigpipe failover standby
```

This command causes the BIG/ip Controller to surrender its unit number to its peer. That is, its peer now becomes both units 1 and 2, the BIG/ip Controller appears out of service from a fail-over perspective, it has no unit numbers. You can make any changes, such as configuration changes, before causing the machine to resume normal operation.

Placing an active-active controller back in service if automatic failback is disabled

If the **Common.Bigip.Failover.ManFailBack** key is set to **0** (off), normal operation is restored when you issue a **bigpipe failover failback** command on the controller with no unit number.

In active-active mode, type the following command to place a standby controller back in service:

```
bigpipe failover failback
```

This command causes the BIG/ip Controller to resume its unit number. That is, the peer now relinquishes the unit number of the controller that has resumed service.

However, if the **Common.Bigip.Failover.ManFailBack** key is set to **1** (on), normal operations are restored when you issue a **bigpipe failover failback** command on the controller running with both unit numbers.

Additional active-active BIG/db configuration parameters

There are several new BIG/db parameters for active-active mode.

Common.Bigip.Failover.ActiveMode
Set this BIG/db parameter to **1** to enable active-active mode. The default setting is off, and redundant systems run in active/standby mode.

Local.Bigip.Failover.UnitId
This is the default unit number of the BIG/ip Controller. This value is set by the First-Time Boot utility or when you upgrade your controllers to BIG/ip Controller, version 3.0.

Common.Bigip.Failover.ManFailBack
This is set to **1** so that manual intervention is required (the **bigpipe failover failback** command is issued) before a BIG/ip Controller running both unit numbers surrenders a unit number to its peer. This feature is **off** by default, fail-back is automatic. For more details, see the section Active-active fail-over.

Common.Bigip.Failover.NoSyncTime
Set this to **1** if you do not want to synchronize the time of the two BIG/ip Controllers. Normally, their time is synchronized. For some cases, this is not desirable, for example, if you are running **ntpd**.

Common.Bigip.Failover.AwaitPeerDeadDelay
The BIG/ip Controller checks to see that its peer is still alive at this rate (in seconds). The default value for this parameter is one second.

Common.Bigip.Failover.AwaitPeerAliveDelay
Check status of a peer BIG/ip Controller while waiting for it to come to life with this frequency (in seconds). The default value of this parameter is three seconds.

Common.Bigip.Failover.DbgFile
If a file name is specified, the fail-over daemon logs state change information in this file. This value is not set by default.

Common.Bigip.Failover.PrintPeerState
Causes the fail-over daemon to periodically write the state of its connections to its peer (hard-wired and/or network) to the log file **Common.Bigip.Failover.DbgFile**.

Additional commands for displaying active vs. mirrored data

The **dump** commands explicitly show those connections (and other objects) that are active on the BIG/ip Controller, and those that are standby connections for the peer BIG/ip Controller. In prior versions of the BIG/ip Controller, one controller is the active unit and the other is the standby. When the **bigpipe conn dump** command is issued on the active unit, each of the connections shown is active. Similarly, when the **bigpipe conn dump** command is issued on the standby unit, it is clear that each of the connections listed is a standby connection. These standby connections are created by mirroring the active connections on the standby unit.

In an active-active installation, each unit can be considered a standby for its 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. You can use the following commands with the mirror option:

```
bigpipe conn dump [mirror]
bigpipe vip persist dump [mirror]
bigpipe sticky dump [mirror]
```

Also, the **bigpipe snat show** command output has been modified to show whether a connection listed is an active connection or a mirror connection.

New active-active bigpipe commands

Several new commands have been added to **bigpipe** to reflect new or changed functionality.

`bigpipe failover init`

This command causes the fail-over daemon (**/sbin/sod**) to read the BIG/db database and refresh its parameters.

bigpipe failover failback

After a **bigpipe failover standby** command is issued, issue this command to allow the BIG/ip Controller to resume normal operation. If manual fail back is enabled, this command causes a BIG/ip Controller that is running as both units to release a unit number to its peer unit when the peer becomes active.

`bigpipe unit [show]`

Displays the unit number(s) of the BIG/ip Controller.

`bigpipe unit peer [show]`

Displays the unit number(s) of the peer BIG/ip Controller.

Running mixed versions of BIG/ip Controller software in active-active mode

The BIG/ip Controller provides the option to install a new version of the BIG/ip Controller software on one BIG/ip Controller, while the other BIG/ip Controller runs a previous production version of the software. This allows you to fail back and forth between the two units, testing the new software yet having the ability to return to the prior installation.

This is possible with the new fail-over software in version 3.0, whether using active-active mode or active/standby mode. However, there are some exceptions:

State mirroring is not compatible between the version 3.0 and prior versions of the software. Network fail-over is also not compatible.

If you are running the BIG/ip Controller version 3.0 in active-active mode, you should assign unit two to the BIG/ip Controller running version 3.0.

Returning an active-active installation to active/standby mode

Returning to active/standby mode from active-active mode is relatively simple in that only a few things need be undone.

1. Enable active/standby mode by setting the BIG/db key **Common.Bigip.Failover.ActiveMode** to **0**.
2. Update the fail-over daemon with the change by typing **bigpipe failover init**.
3. To synchronize the configuration, type the command **bigpipe configsync all**.
4. Since each BIG/ip Controller is an active unit, type the command bigpipe failover standby on each controller. This transitions each controller into active/standby mode.

When in active/standby mode, the active BIG/ip Controller runs all objects (virtual servers, SNATs and NATs) that are defined to run on unit 1 or unit 2. It is not necessary to redefine virtuals servers, SNATS, or NATs when you transition from active-active mode to active/standby mode.

7

Using Advanced Network Configurations

- **Introducing advanced network configurations**
- **nPath routing**
- **Per-connection routing**
- **ISP load balancing**
- **Firewall load balancing**
- **VPN load balancing**
- **VPN and router load balancing**
- **SNAT and virtual servers combined**
- **One IP network topology with one interface**
- **One IP network topology with two interfaces**
- **Setting up 802.1q VLAN trunk mode**

Introducing advanced network configurations

In addition to the basic setup features available on the BIG/ip Controller, a number of special network configurations can be used to optimize your network. This chapter describes a number of special network configurations possible with the BIG/ip Controller. These configurations are optional, and may not be required in your implementation of the BIG/ip Controller. The following topics are described in this chapter:

- nPath routing
- Per-connection routing
- The firewall sandwich configuration
- ISP load balancing
- Firewall load balancing
- VPN load balancing
- VPN and router load balancing
- One IP network topology with one interface
- One IP network topology with two interfaces
- 802.1q VLAN trunk mode

nPath routing

nPath routing allows you to route outgoing server traffic around the BIG/ip Controller directly to an outbound router. This method of traffic management increases outbound throughput because packets do not need to be transmitted to the BIG/ip Controller for translation and forwarding to the next hop.

To use nPath routing, you must configure the BIG/ip Controller so that it does not translate the IP address or port of incoming packets. This is important because packets must not be translated when they are outbound to the router. To avoid translation of incoming, or destination packets, you must define virtual servers with address translation turned **off**.

The following tasks are required to configure the BIG/ip Controller to use nPath routing:

- Define a virtual server
- Turn address translation **off** for the virtual server
- Set a route on your routers to the virtual server with the BIG/ipController as the gateway
- Set the idle connection time-out value to remove stale connections
- Configure the servers

Defining a virtual server with address translation disabled

You can disable address translation on any virtual server. Turning off address translation is necessary for nPath routing. The following two procedures describe how to create a virtual server in the F5 Configuration utility and then how to turn address translation off for the virtual server.

To define a standard virtual server mapping in the F5 Configuration utility

1. In the navigation pane, click Virtual Servers.
2. On the toolbar, click **Add Virtual Server**.
   The Add Virtual Server screen opens.
3. In the **Address** box, enter the virtual server's IP address or host name.
4. In the **Netmask** box, type an optional netmask. If you leave this setting blank, the BIG/ip Controller uses a default netmask based on the IP address you entered for the virtual server. Use the default netmask unless your configuration requires a different netmask.
5. In the **Broadcast** box, type the broadcast address for this virtual server. If you leave this box blank, the BIG/ip Controller generates a default broadcast address based on the IP address and netmask of this virtual server.

6. In the **Port** box, either type a port number, or select a service name from the drop-down list.
7. For **Interface**, select the external (destination processing) interface on which you want to create the virtual server. Select **default** to allow the F5 Configuration utility to select the interface based on the network address of the virtual server.
8. In Resources, click the **Node List** button.
9. In the **Node Address** box, type the IP address or host name of the first node to which the virtual server maps. If you have already defined a node address, you can choose it from the list.
10. In the **Node Port** box, type the node port number, or select the service name from the drop-down list. If you have already defined a node port, you can choose it from the list.
11. Click the add button (>>) to add the node the Current Members list for the virtual server.
12. To add additional nodes to the virtual server mapping, type in a Node Address, Node Port, and click the add button (>>).
13. To remove nodes from the virtual server mapping, click the node listed in the Current Members list and click the remove button (<<).
14. After you have added or removed nodes from the Current Members list, click the **Add** button to save the virtual server.

To configure address translation for virtual servers in the F5 Configuration utility

After you create a virtual server, you must turn address translation for the virtual server **off**.

1. In the navigation pane, click Virtual Servers.
   The Virtual Servers screen opens.

2. In the virtual server list, click the virtual server for which you want to set up a transparent virtual server.
   The properties screen for the virtual server you clicked opens.
3. In the Enable Translation options, clear the **Address** check box. This turns address translation **off** for the virtual server.
4. Click the **Apply** button.

To define a virtual server mapping on the command line

Enter the **bigpipe vip** command as shown below to create the virtual server mapping. Note that you must turn off address translation for the virtual server you create.

```
bigpipe vip <virtual IP>:<port> define <node IP>:<port> \
   <node IP>:<port>...  <node IP>:<port>
```

For example, the following command defines a virtual server that maps to three nodes. After you create the virtual server, you must turn off address translation. Use the following syntax to turn off address translation for the virtual server.

```
bigpipe vip <vip>:<port> translate addr [ enable | disable ]
```

For example, use the following command to turn off address translation for the virtual server 11.1.1.1:80.

```
bigpipe vip 11.1.1.1:80 translate addr disable
```

Setting the route through the BIG/ip Controller

A route must be defined through the BIG/ip Controller on the inbound router in your network configuration. This route should be the IP address (or alias) for the server, or servers, for which you want to set up nPath routing. The gateway should be the external shared IP alias of the BIG/ip Controller.

For information about how to define this route, please refer to the documentation provided with your router.

Setting the idle connection time-out

With nPath routing, the BIG/ip Controller cannot track the normal FIN/ACK sequences made by connections. Normally, the BIG/ip Controller shuts down closed connections based on this sequence. With nPath routing, the idle connection time-out must be configured to clean up closed connections. You need to set an appropriate idle connection time-out value so that valid connections are not disconnected, and closed connections are cleaned up in a reasonable time.

To set the idle connection time-out in the F5 Configuration utility

1. In the navigation pane, click Virtual Servers.
2. In the Virtual Servers list, click the wildcard virtual server you created for nPath routing.
   The Virtual Server Properties screen opens.
3. In the **Port** box, click the port.
   The Global Virtual Port Properties screen opens.
4. In the **Idle connection timeout TCP (seconds)** box, type a time-out value for TCP connections. The recommended time-out setting is 10 seconds.
5. In the **Idle connection timeout UDP (seconds)** box, type a time-out value for TCP connections. The recommended time-out setting is 10 seconds.
6. Click **Apply**.

To set the idle connection time-out in the /etc/bigip.conf file

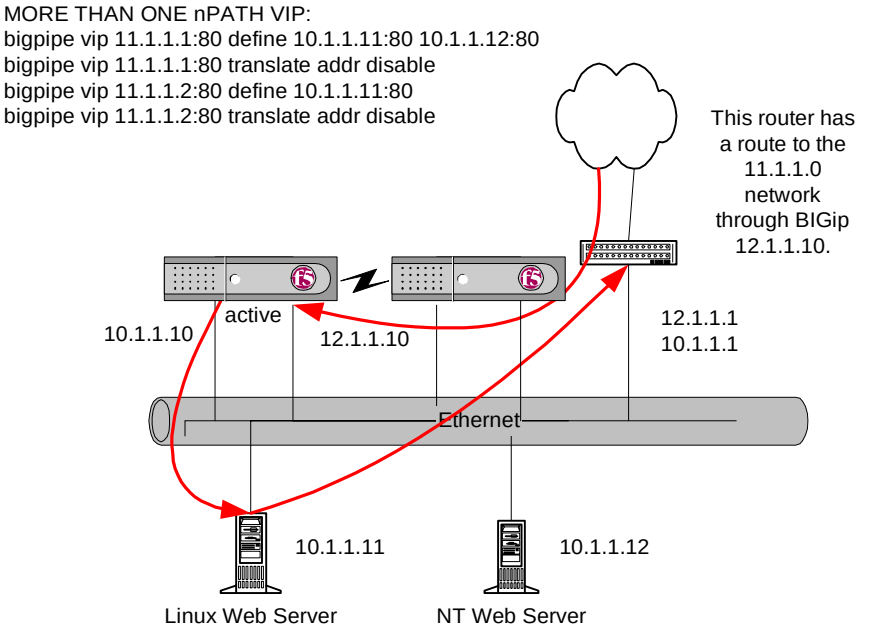
To set the idle connection time-out in the **/etc/bigip.conf** file, edit the following lines:

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

The **<seconds>** value is the number of seconds a connection is allowed to remain idle before it is terminated. The **<port>** value is the port on the wildcard virtual server for which you are configuring out of path routing. The recommended value for the TCP and UDP connection timeouts is 10 seconds.

Configure the Servers

You must configure your servers differently to work in nPath mode. The IP address of the server (11.1.1.1 in Figure 7.1) must be placed on what is known as the ***loopback*** interface. A loopback interface is a software interface that is not associated with an actual network card. It allows a server to respond to an IP address without advertising it on a network. Most UNIX variants have a loopback interface named **lo0**. Microsoft Windows has an MS Loopback interface in its list of network adaptors. Consult your server operating system documentation for information about configuring an IP address on the loopback interface. The ideal loopback interface for the nPath configuration does not participate in the ARP protocol, because that would cause packets to be routed incorrectly.



These servers have an HTTP service listening on address 11.1.1.1, port 80. The address 11.1.1.1 is bound to the loopback device. Their default route is 10.1.1.1.

Figure 7.1 *An example nPath configuration with more than one virtual server*

Per-connection routing

In situations where the BIG/ip Controller accepts connections for virtual servers from more than one router, you can send the return data back through the same device from which the connection originated. You can use this option to spread the load among outbound routers, or to ensure that connections go through the same device if that device is connection-oriented, such as a proxy, cache, firewall, or VPN router.

To set up last hop pools, define a list of routers as a pool from which the BIG/ip Controller receives packets. For information about creating a pool, see *Defining pools*, on page 3-4. The BIG/ip Controller determines the MAC address of the routers when you define the pool. Then the pool is associated with the virtual server by using the **lasthop** keyword to specify the last hop pool for the virtual server. When a packet arrives for the virtual server, the MAC address that the packet came from is matched up with the MAC address of the members of the last hop pool. The IP address of the matching member is stored with the connection as the last hop address. Then, connections coming from nodes and heading out towards the client are routed to the router with that last hop address, instead of to the default route.

◆ Note

The packets must come from a member of the last hop pool or they are rejected.

To configure last hop pools for virtual servers from the command line

Use the following syntax to configure last hop pools for virtual servers:

```
bigpipe vip <vip>:<port> lasthop pool <pool_name>
```

For example, you might use the following command:

```
bigpipe vip 192.168.1.10:80 lasthop pool secure_routers
```

To configure last hop pools for virtual servers in the F5 Configuration utility

Before you follow this procedure, you must configure at least one pool (for your routers or firewalls) and one virtual server that references the pool.

1. In the navigation pane, click Virtual Servers.
   The Virtual Servers screen opens.
2. In the virtual server list, click the virtual server for which you want to set up a last hop pool.
   The properties screen for the virtual server you clicked opens.

3. Click the **Last Hop Pool** list. Select the pool you created containing your routers.
4. Click the **Apply** button.

ISP load balancing

You may find that as your network grows, or network traffic increases, you need to add an additional connection to the internet. You can use this configuration to add an additional internet connection to your existing network. Figure 7.2 shows an additional internet connection:

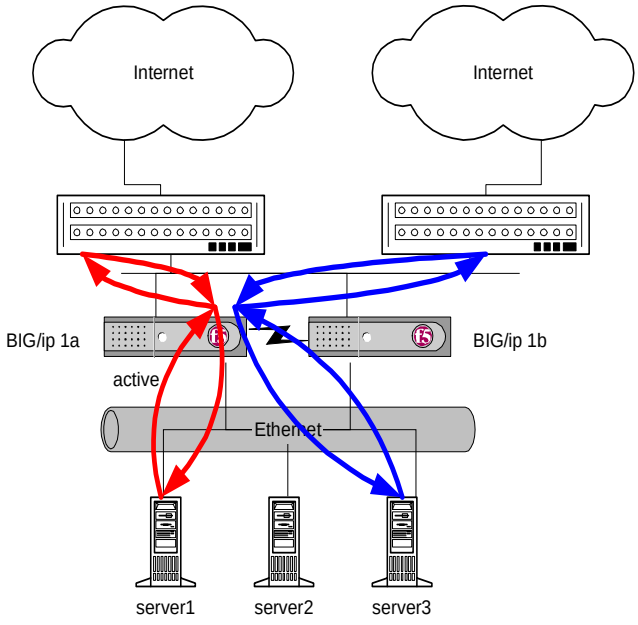


Figure 7.2 *An example of an additional internet connection*

Configuring interfaces for the additional internet connection

An additional internet connection requires special interface configuration. You must set interfaces on the redundant BIG/ip Controller system (1a and 1b in Figure 7.2) to process source and destination addresses. Note that in a basic controller configuration, one interface is configured as an internal interface (source processing), and the other interface is configured as an external interface (destination processing).

In order to load balance outbound connections, you must turn destination processing **on** for the internal interface, and source processing **on** for the external interface. Use the following command to turn destination processing on for the internal interface, in this example, the interface name is **exp1**:

```
bigpipe interface exp1 dest enable
```

Use the following command to turn source processing on for the external interface, in this example, the interface name is **exp0**:

```
bigpipe interface exp0 source enable
```

Configuring virtual servers for an additional internet connection

An additional internet connection requires you to create a pool for the inside interfaces of the routers. After you create the pool, you can create the virtual servers that reference these pools.

Defining the pools for the additional internet connection

First, define the pool **router_insides** for the internal addresses of the routers. Use the following command to create the pool **router_insides**:

```
bigpipe pool router_insides { lb_mode rr member <router1>:0 member
   <router2>:0 }
```

Replace **<router1>** and **<router2>** with external IP address of the respective routers. Also note that this example uses the global round robin load balancing method.

Finally, define the pool **server_pool** for the nodes that handle the requests to virtual server 205.100.19.22:80:

```
bigpipe pool server_pool { lb_mode rr member <server1>:80 member
   <server2>:80 member <server3>:80 }
```

Replace **<server1>**, **<server2>**, and **<server3>** with internal IP address of the respective server. Also note that this example uses the global round robin load balancing method.

Defining the virtual servers for the additional internet connection

After you define the pools for the inside IP addresses of the routers, you can define the virtual servers for the redundant BIG/ip Controllers 1a and 1b.

- Configure the redundant controllers to load balance inbound connections
- Configure the redundant controllers to load balance outbound connections

Inbound configuration

First, configure the controllers to handle inbound traffic.

Create the virtual server for controllers 1a and 1b with the following command:

```
bigpipe vip 205.100.92.22:80 use pool server_pool
```

Configure the virtual server to use the last hop pool with the routers inside addresses:

```
bigpipe vip 205.100.92.22:http lasthop pool
   router_insides
```

Outbound configuration

Next, configure controllers 1a and 1b to handle outbound traffic. Create a virtual server that sends traffic to the pool you created for the internal interfaces of the routers (**router_insides**). Use the following command to create the virtual server:

```
bipipe vip 0.0.0.0:0 exp1 use pool router_insides
```

Firewall load balancing

You may find that as your network grows, or network traffic increases, you need to add an additional firewall. You can use the firewall sandwich configuration to add an additional firewall to your existing network. Figure 7.3 shows the firewall sandwich:

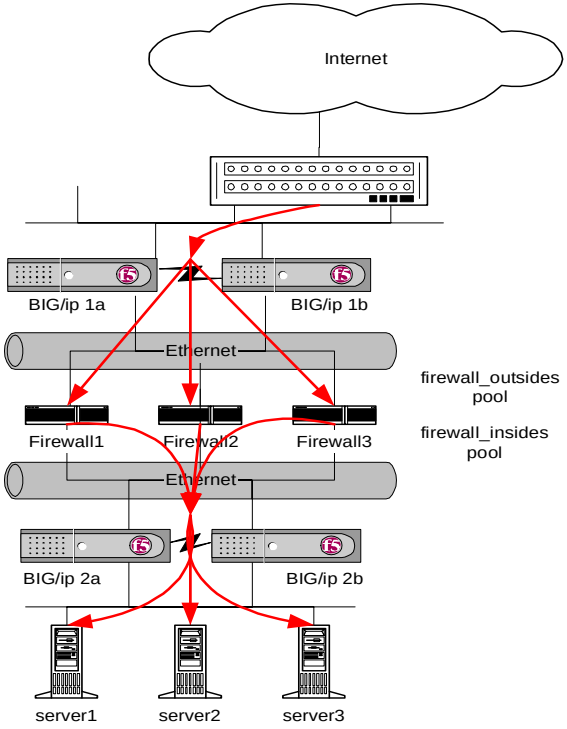


Figure 7.3 *An example of a firewall sandwich configuration*

Configuring interfaces for the firewall sandwich

The firewall sandwich requires additional interface configuration. You must set the interfaces on both redundant BIG/ip Controller systems (1a and 1b, and 2a and 2b, in Figure 7.3) to process source and destination addresses. Note that in a basic controller configuration, one interface is configured as an internal interface (source processing), and the other interface is configured as an external interface (destination processing).

In order for the firewall sandwich to work, you must turn destination processing **on** for the internal interface, and source processing **on** for the external interface. Use the following command to turn destination processing on for the internal interface, in this example, the interface name is **exp1**:

```
bigpipe interface exp1 dest enable
```

Use the following command to turn source processing on for the external interface, in this example, the interface name is **exp0**:

```
bigpipe interface exp0 source enable
```

Configuring virtual servers for a firewall sandwich

The firewall sandwich requires you to create a pool for the inside and outside interfaces on the firewalls. You can also create a pool for the servers. After you create the three pools, you can create the virtual servers that reference these pools.

Defining the pools for the firewall sandwich

First, define the pool **firewall_outsides** for the external addresses of the firewall. Use the following command to create the pool **firewall_outsides**:

```
bigpipe pool firewall_outsides { lb_mode rr member <Firewall1>:0
   member <Firewall2>:0 member <Firewall3>:0 }
```

Replace <**Firewall1**>, <**Firewall2**>, and <**Firewall3**> with external IP address of the respective firewall. Also note that this example uses the global round robin load balancing method.

Next, define the pool **firewall_insides** for the internal addresses of the firewall. Use the following command to create the pool **firewall_insides**:

```
bigpipe pool firewall_insides { lb_mode rr member <Firewall1>:0
   member <Firewall2>:0 member <Firewall3>:0 }
```

Replace <**Firewall1**>, <**Firewall2**>, and <**Firewall3**> with internal IP address of the respective firewall. Also note that this example uses the global round robin load balancing method.

Finally, define the pool **server_pool** for the nodes that handle the requests to virtual server 205.100.19.22:80:

```
bigpipe pool server_pool { lb_mode rr member <server1:>80 member
   <server2>:80 member <server3>:80 }
```

Replace <**server1**>, <**server2**>, and <**server3**> with the IP address of the respective server. Also note that this example uses the global round robin load balancing method.

Defining the virtual servers for the firewall sandwich

After you define the pools for the inner and outer interfaces of the firewalls, you can define the virtual servers for the redundant BIG/ip Controllers 1a and 1b, and 2a and 2b. To do this, you must complete these steps:

- Configure the redundant controllers in both configurations to load balance inbound connections
- Configure the redundant controllers in both configurations to load balance outbound connections

Inbound configuration

Since the 1a and 1b redundant controller configuration need to load balance inbound connections across the pool of firewalls, you need to configure a virtual server that references the pool you defined called **firewall_outsides**. To create this virtual server for the controllers 1a and 1b, use the following command:

```
bigpipe vip 205.100.92.22:http use pool firewall_outsides
```

After you create the virtual server that references the **firewall_outsides** pool, you need to disable address translation for that virtual server. Use the following command to disable address translation:

```
bigpipe vip 205.100.19.22:http translate addr disable
```

Turning address translation off prevents the virtual server from translating the destination address on the packet to the IP address of the firewalls.

After you configure the controllers 1a and 1b to handle inbound traffic, then you should configure controllers 2a and 2b to handle inbound traffic.

First, create the virtual server for controllers 2a and 2b with the following command:

```
bigpipe vip 205.100.92.22:80 use pool server_pool
```

Configure the virtual server to use the last hop pool with the inside firewall addresses:

```
bigpipe vip 205.100.92.22:http lasthop pool firewall_insides
```

Outbound configuration

After you configure the virtual servers for the inbound traffic, you can define the virtual servers to handle the outbound connections.

To configure the virtual server for outbound connections on controllers 1a and 1b, you need to create a virtual server that accepts traffic from the pool handling outbound traffic from the firewalls (**firewall_outsides**), and a virtual server that forwards all traffic out to the internet. First, create the virtual server that forwards traffic to the internet:

```
bigpipe vip 0.0.0.0:0 exp1 forward
```

Then, configure the virtual server to use the pool that contains the outside addresses of the firewalls:

```
bigpipe vip 0.0.0.0:0 lasthop pool firewall_outsides
```

After you configure controllers 1a and 1b, configure controllers 2a and 2b to handle outbound traffic. Create a virtual server that sends traffic to the pool you created for the internal interfaces of the firewalls (**firewall_insides**). Use the following command to create the virtual server:

```
bipipe vip 0.0.0.0:0 exp1 use pool firewall_insides
```

◆ Note

The default route must be set on the BIG/ip Controller for wildcard forwarding to work.

VPN load balancing

You can use the BIG/ip Controller to load balance virtual private network (VPN) routers used to connect two private networks. Since neither translation nor load balancing is required, you can combine a ***forwarding*** virtual server with a **lasthop** pool.

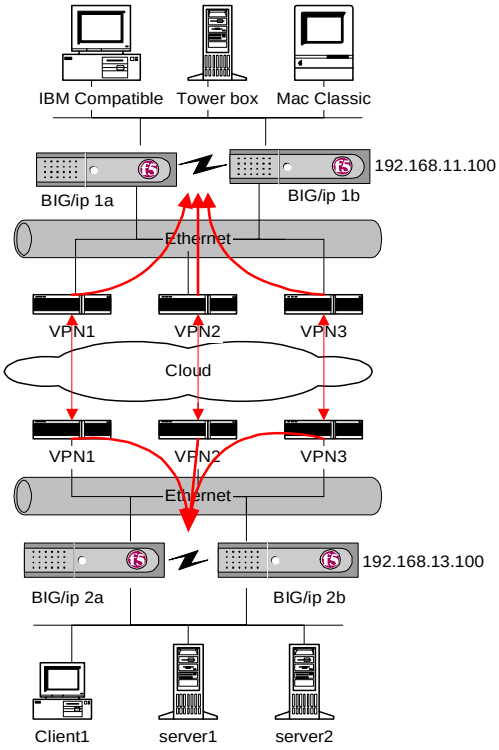


Figure 7.4 *An example of a VPN load balancing configuration*

Configuring interfaces for VPN load balancing

A VPN load balancing configuration requires special interface configuration. You must configur e the interfaces on the redundant BIG/ip Controller system (1a and 1b, and 2a and 2b, in Figure 7.4) to process source and destination addresses. Note that in a basic controller configuration, one interface is configured as an internal interface (source processing), and the other interface is configured as an external interface (destination processing).

In order for the VPN load balancing to work, you must turn destination processing **on** for the internal interface, and source processing **on** for the external interface. Use the following command to turn destination processing on for the internal interface, in this example, the interface name is **exp1**:

```
bigpipe interface exp1 dest enable
```

Use the following command to turn source processing on for the external interface, in this example, the interface name is **exp0**:

```
bigpipe interface exp0 source enable
```

Configuring virtual servers for VPN load balancing

In the following examples only the configuration for the BIG/ip Controller on network 192.168.11 are shown (controllers 2a and 2b). The configuration for 192.168.13 is the same, only with different network numbers. Since VPNs are connection-oriented, you must set up a last hop pool for sending the return traffic back through the VPN that originated the traffic. After you create the pools, you can create the virtual servers that reference these pools.

Defining the pools for VPN load balancing

First, define the pool **vpn_insides** for the internal addresses of the VPN routers. Use the following command to create the pool **vpn_insides**:

```
bigpipe pool vpn_insides { lb_mode rr member <vpn1>:22 member
   <vpn2>:22 member <vpn3>:22 }
```

Replace <**vpn1**>, <**vpn2**>, and <**vpn3**> with internal IP address of the respective routers. In this example the routers are service checked on port 22. Also note that this example uses the global round robin load balancing method.

Finally, define the pool **server_pool** for the nodes that handle the requests to virtual server 205.100.19.22:80:

```
bigpipe pool server_pool { lb_mode rr member <server1>:80 member
   <server2>:80 member <server3>:80 }
```

Replace <**server1**>, <**server2**>, and <**server3**> with internal IP address of the respective server. Also note that this example uses the global round robin load balancing method.

Defining the virtual servers for VPN load balancing

After you define the pools for the inside IP addresses of the routers, you can define the virtual servers for the redundant BIG/ip Controllers 2a and 2b.

- Configure the redundant controllers to load balance inbound connections
- Configure the redundant controllers to load balance outbound connections

Inbound configuration

First, configure the controllers to handle inbound traffic from the remote network.

Create the virtual servers for controllers 2a and 2b with the following commands:

```
bigpipe vip 192.168.13.1:0 exp1 forward
bigpipe vip 192.168.13.2:0 exp1 forward
bigpipe vip 192.168.13.3:0 exp1 forward
```

Configure the virtual servers to use the last hop pool with the inside VPN router addresses:

```
bigpipe vip 192.168.13.1:0 lasthop pool vpn_insides
bigpipe vip 192.168.13.2:0 lasthop pool vpn_insides
bigpipe vip 192.168.13.3:0 lasthop pool vpn_insides
```

Outbound configuration

Next, configure controllers 2a and 2b to handle outbound traffic. Create a virtual server that sends traffic to the pool you created for the internal interfaces of the VPN routers (**vpn_insides**). Use the following commands to create virtual servers for connecting to the machines on the remote network:

```
bipipe vip 192.168.11.1:0 exp1 use pool vpn_insides
bipipe vip 192.168.11.1:0 translate addr disable
bipipe vip 192.168.11.1:0 translate port disable
bipipe vip 192.168.11.2:0 exp1 use pool vpn_insides
bipipe vip 192.168.11.2:0 translate addr disable
bipipe vip 192.168.11.2:0 translate port disable
bipipe vip 192.168.11.3:0 exp1 use pool vpn_insides
bipipe vip 192.168.11.3:0 translate addr disable
bipipe vip 192.168.11.3:0 translate port disable
```

The addresses 192.168.11.1, 192.168.11.2, and 192.168.11.3 correspond to the IBM Compatible, Tower box, and Mac Classic on the remote network in Figure 7.4. Note that port translation has been turned off because the members in the **vpn_insides** pool were defined with port 22 for service checking. If port translation is not disabled, then all outbound connections would be translated to port 22.

VPN and router load balancing

You can use the transparent device load balancing feature in the BIG/ip Controller to connect two private networks as well as load balance internet connections through multiple routers. Figure 7.5 is an example of this network configuration.

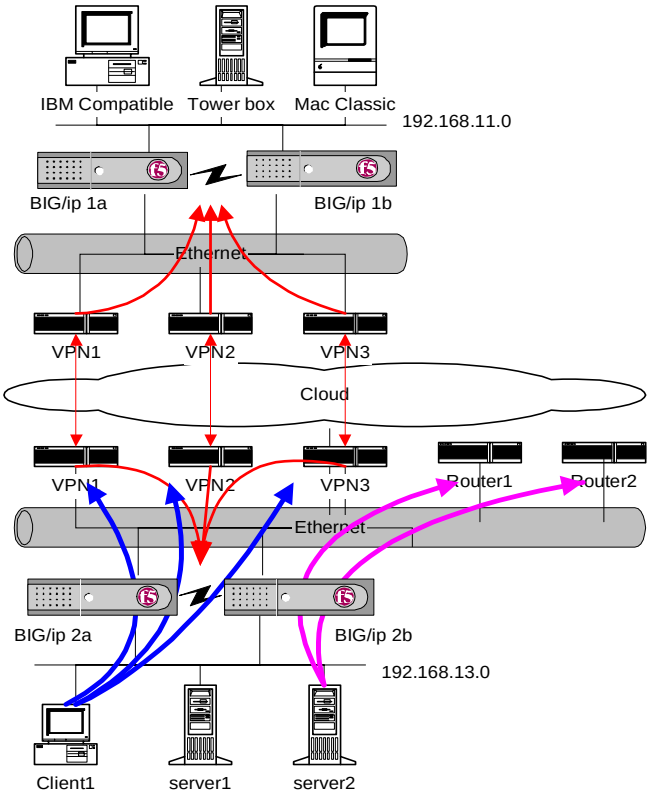


Figure 7.5 *An example of a VPN and multiple router load balancing configuration*

Configuring interfaces for VPN load balancing

A VPN load balancing configuration requires special interface configuration. The interfaces on the redundant BIG/ip Controller system (1a and 1b, and 2a and 2b, in Figure 7.5) must be set to process source and destination addresses. Note that in a basic controller configuration, one interface is configured as an internal interface (source processing), and the other interface is configured as an external interface (destination processing).

In order for VPN load balancing to work, you must turn destination processing **on** for the internal interface, and source processing **on** for the external interface. Use the following command to turn destination processing on for the internal interface, in this example, the interface name is **exp1**:

```
bigpipe interface exp1 dest enable
```

Use the following command to turn source processing on for the external interface, in this example, the interface name is **exp0**:

```
bigpipe interface exp0 source enable
```

Configuring virtual servers for VPN and router load balancing

In the following examples, only the configuration for the BIG/ip Controller on network 192.168.11 are shown (controllers 2a and 2b). The configuration for 192.168.13 is the same, only with different network numbers. Since VPNs are connection-oriented, VPN and router load balancing requires you to create a pool for the inside interfaces on the VPNs and routers. After you create the pool, you can create the virtual servers that reference these pools.

Defining the pools for VPN load balancing

First, define the pool **vpn_insides** for the internal addresses of the VPN routers. Use the following command to create the pool **vpn_insides**:

```
bigpipe pool vpn_insides { lb_mode rr member <vpn1>:0 member
   <vpn2>:0 member <vpn3>:0 }
```

Replace <**vpn1**>, <**vpn2**>, and <**vpn3**> with external IP address of the respective routers. Also note that this example uses the global round robin load balancing method.

Defining pools for the additional routers

Next, define the pool **routers_insides** for the internal addresses of the routers. Use the following command to create the pool **routers_insides**:

```
bigpipe pool routers_insides { lb_mode rr member <router1>:0 member
   <router2>:0 }
```

Replace <**router1**> and <**router2**> with internal IP address of the respective routers. Also note that this example uses the global round robin load balancing method.

Defining a pool for the servers

Next, define the pool **server_pool** for the nodes that handle the requests to virtual server 205.100.19.22:80:

```
bigpipe pool server_pool { lb_mode rr member <server1>:80 member
   <server2>:80 member <server3>:80 }
```

Replace <**server1**>, <**server2**>, and <**server3**> with IP address of the respective server. Also note that this example uses the global round robin load balancing method.

Defining a pool for all inbound traffic sources

Finally, define the pool **inbound_sources** for all machines that can originate traffic for the virtual server 205.100.19.22:80:

```
bigpipe pool inbound_sources { lb_mode rr member <vpn1>:80 member
   <vpn2>:80 member <vpn3>:80 member <router1>:80 member
   <router2>:80 member <server1>:80 member <server2>:80 member
   <server3>:80 }
```

Replace <**vpn1**>, <**vpn2**>, and <**vpn3**> with internal IP address of the respective routers. Replace <**server1**>, <**server2**>, and <**server3**> with IP address of the respective server. Replace <**router1**> and <**router2**> with internal IP address of the respective routers. Also note that this example uses the global round robin load balancing method.

Defining the virtual servers for the additional internet connection

After you define the pools for the inside IP addresses of the routers, you can define the virtual servers for the redundant BIG/ip Controllers 2a and 2b.

❖ Configure the redundant controllers to load balance inbound connections

- Configure the redundant controllers to load balance outbound connections

Inbound configuration for the VPNs

First, configure the controllers to handle inbound traffic from the remote network.

Create the virtual server for controllers 2a and 2b with the following commands:

```
bigpipe vip 192.168.13.1:0 exp1 forward
bigpipe vip 192.168.13.2:0 exp1 forward
bigpipe vip 192.168.13.3:0 exp1 forward
```

Configure the virtual server to use the last hop pool with the inside VPN router addresses:

```
bigpipe vip 192.168.13.1:0 lasthop pool vpn_insides
bigpipe vip 192.168.13.2:0 lasthop pool vpn_insides
bigpipe vip 192.168.13.3:0 lasthop pool vpn_insides
```

Note that by using the last hop pool **vpn_insides**, only connections that originate from the remote network, through the VPNs, will be allowed to connect to the local 192.168.11 network.

Outbound configuration for the VPNs

Next, configure controllers 2a and 2b to handle outbound traffic. Create a virtual server that sends traffic to the pool you created for the internal interfaces of the VPN routers (**vpn_insides**). Use the following commands to create virtual servers for connecting to the machines on the remote network:

```
bipipe vip 192.168.11.1:0 exp1 use pool vpn_insides
bipipe vip 192.168.11.1:0 translate addr disable
bipipe vip 192.168.11.2:0 exp1 use pool vpn_insides
bipipe vip 192.168.11.2:0 translate addr disable
bipipe vip 192.168.11.3:0 exp1 use pool vpn_insides
bipipe vip 192.168.11.3:0 translate addr disable
```

The addresses 192.168.11.1, 192.168.11.2, and 192.168.11.3 correspond to the IBM Compatible, Tower box, and Mac Classic on the remote network in Figure 7.4.

Inbound configuration for internet traffic

First, configure the controllers to handle inbound traffic.

Create the virtual server for controllers 1a and 1b with the following command:

```
bigpipe vip 205.100.92.22:80 use pool server_pool
```

Configure the virtual server to use the last hop pool with the routers inside addresses:

```
bigpipe vip 205.100.92.22:http lasthop pool inbound_sources
```

Note that by using the last hop pool **inbound_sources**, this virtual server will accept connections that originate from either the remote network via the VPNs, or from the internet via the routers.

Outbound configuration for internet traffic

Next, configure controllers 1a and 1b to handle outbound traffic. Create a virtual server that sends traffic to the pool you created for the internal interfaces of the routers (**router_insides**). Use the following command to create the virtual server:

```
bipipe vip 0.0.0.0:0 exp1 use pool router_insides
```

SNAT and virtual servers combined

In some cases you may want to configure outbound transparent device load balancing and SNAT source translations. In this configuration, the BIG/ip Controller changes the source address of the clients to the external SNAT address. In this way, the actual IP addresses are not exposed to the internet. At the same time the BIG/ip Controller can load balance the same connection across multiple nodes. Therefore, both SNAT translation and virtual server load balancing can operate on the same connection in this configuration.

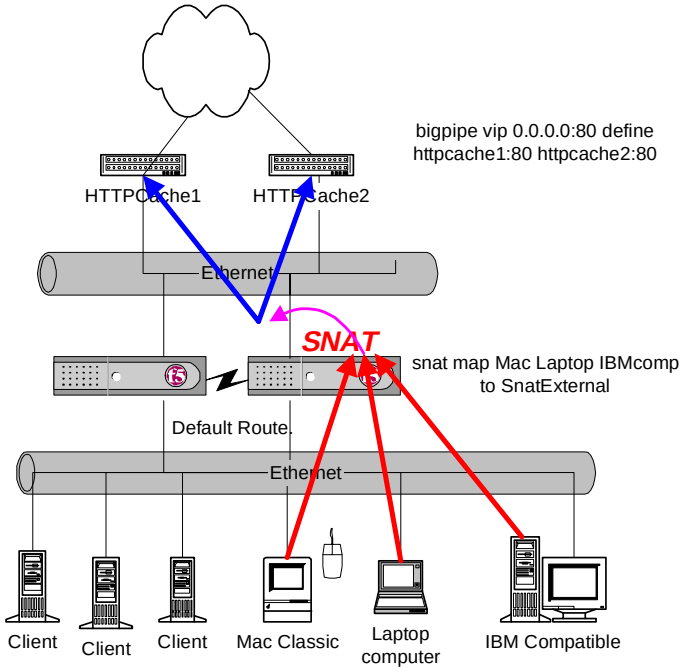


Figure 7.6 *An example of a virtual server/SNAT combination*

Configuring interfaces for the SNAT and virtual server combination

The SNAT and virtual server combination does not require additional interface configuration. However, in this configuration, the destination processing interface must be on the internal network and the source processing interface must be on the external network.

Defining a pool for the HTTP cache servers

Finally, define the pool **cache_pool** for the nodes that handle the requests to virtual server 0.0.0.0:0:

```
bigpipe pool cache_pool { lb_mode rr member <HTTPcache1>:80 member
   <HTTPcache2>:80 }
```

Replace **<HTTPcache1>** and **<HTTPcache2>** with IP address of the respective HTTP cache server. Also note that this example uses the global round robin load balancing method.

Outbound configuration

Next, configure controllers 1a and 1b to handle outbound traffic. Create a virtual server that sends traffic to the pool you created for the internal interfaces of the HTTP cache servers (**cache_pool**). Use the following commands to create a virtual server for connecting to cache servers:

```
bipipe vip 0.0.0.0:0 exp1 use pool cache_pool
bipipe vip 0.0.0.0:0 translate port disable
```

Note that port translation has been turned off because the members in the vpn_insides pool were defined with port 80 for service checking. If port translation is not disabled, then all outbound connections would be translated to port 80.

After you create the virtual server, type the following SNAT command:

```
bipipe snat map client1 client2 client3 to 205.100.19.23
```

Replace **client1**, **client2**, and **client3** with the actual names of the clients in your configuration.

One IP network topology with one interface

The BIG/ip Controller can be used in a single interface configuration when there is only one network in the topology.

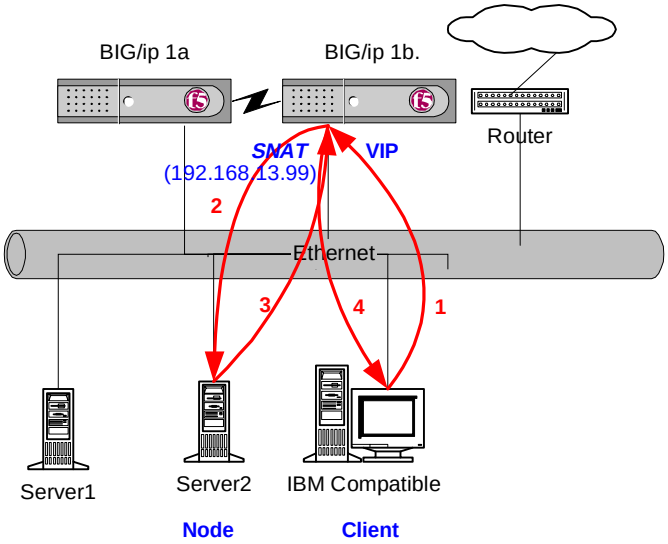


Figure 7.7 *An example of a single interface topology*

Configuring the interface in the single interface topology

A single IP network topology with a single interface requires special interface configuration. You must configure the single interface on the redundant BIG/ip Controller system (1a and 1b, in Figure 7.7) to process source and destination addresses. Note that in a basic controller configuration, one interface is configured as an internal interface (source processing), and the other interface is configured as an external interface (destination processing).

Use the following commands to turn source and destination processing on for the interface; in this example, the interface name is **exp0**:

```
bigpipe interface exp0 source enable
```

```
bigpipe interface exp0 dest enable
```

Defining a pool for the servers

First, define the pool **server_pool** for the nodes that handle the requests to virtual server 192.168.13.1:80

```
bigpipe pool server_pool { lb_mode rr member <server1>:80 member
   <server2>:80 }
```

Replace **<server1>** and **<server2>** with IP address of the respective server. Also note that this example uses the global round robin load balancing method.

Virtual Server Configuration

Next, configure controllers 1a and 1b to load balance connections to the servers. Create a virtual server that sends traffic to the pool you created for the servers (**server_pool**). Use the following commands to create a virtual server for connecting to the servers:

```
bipipe vip 192.168.13.1:80 use pool server_pool
```

Client SNAT Configuration

Finally, configure controllers 1a and 1b to handle connections originating from the client. A SNAT must be defined in order to change the source address on the packet to the SNAT external address, which is located on the BIG/ip Controller. If a SNAT were not defined, the server would return the packets directly to the client without giving the BIG/ip Controller the opportunity to translate the source address from the server address back to the virtual server. The client would not recognize the packet if the source address of the returning packet is the IP address of the real server because the client sent its packets to the IP address of the *virtual* server.

```
bipipe snat map client1 to 192.168.13.99
```

Replace **client1**with the actual name of the client in your configuration.

One IP network topology with two interfaces

The one IP network with two interfaces configuration is similar to the one IP network with one interface configuration, except that it uses two interfaces to optimize throughput.

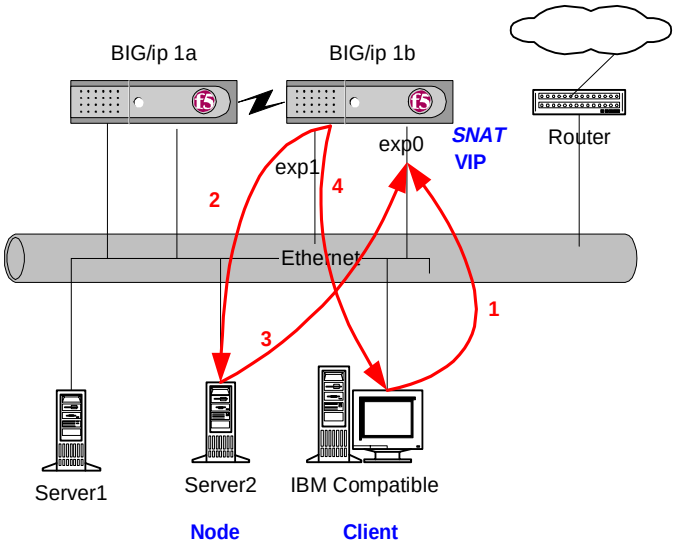


Figure 7.8 *An example of a single IP network with two interfaces topology*

Configuring the interfaces in the single IP network with two interfaces topology

A single IP network with two interfaces topology requires special interface configuration. You must configure both interfaces on the redundant BIG/ip Controller system (1a and 1b, in Figure 7.8) to process source and destination addresses. Note that in a basic controller configuration, one interface is configured as an internal interface (source processing), and the other interface is configured as an external interface (destination processing).

In order for this configuration to work, you must turn destination processing **on** for the internal interface, and source processing **on** for the external interface. Use the following command to turn destination processing on for the internal interface, in this example, the interface name is **exp1**:

```
bigpipe interface exp1 dest enable
```

Use the following command to turn source processing on for the external interface, in this example, the interface name is **exp0**:

```
bigpipe interface exp0 source enable
```

Routing Issues

By setting up the IP addresses and interfaces properly, you can configure the BIG/ip Controller to receive all traffic through one interface and to send all traffic out the other interface. The key to optimizing the throughput in this configuration is routing.

In this example, the administrative IP addresses for the BIG/ip Controller are placed on **exp1**. This is setup when you first configure the BIG/ip Controller, or it can be changed anytime by editing the **/etc/netstart** file. Once the administrative address is configured, the BIG/ip Controller sets up a route to the IP network going through **exp1**. The **exp0** interface should not be configured with an IP address on the same IP network, because that creates a routing conflict and the BIG/ip Controller would not know which interface the IP network is accessible through. Once the route is set up properly, all traffic sent from the BIG/ip Controller to that IP network goes through **exp1**.

In order to receive traffic through **exp0**, the virtual server and SNAT external address in this example are explicilty declared to reside on **exp0**. This causes the BIG/ip Controller to respond to ARP requests for those addresses from the **exp0** interface. Virtual servers and SNAT addresses will not create a routing conflict for the IP network they are declared with. Only administrative or shared IP addresses create routes to the corresponding IP network through the interface that they are configured on. In other words, in this example, the BIG/ip Controller determines that the 192.168.13 network is on interface **exp1** and it sends all traffic to those addresses through that interface.

Defining a pool for the servers

First, define the pool **server_pool** for the nodes that handle the requests to virtual server 192.168.13.1:80

```
bigpipe pool server_pool { lb_mode rr member <server1>:80 member
   <server2>:80 }
```

Replace **<server1>** and **<server2>** with IP address of the respective server. Also note that this example uses the global round robin load balancing method.

Virtual Server Configuration

Next, configure controllers 1a and 1b to load balance connections to the servers. Create a virtual server that sends traffic to the pool you created for the servers (**server_pool**). Use the following commands to create a virtual server for connecting to the servers:

```
bipipe vip 192.168.13.1:80 exp0 use pool server_pool
```

Client SNAT Configuration

Finally, configure controllers 1a and 1b to handle connections originating from the client. A SNAT must be defined in order to change the source address on the packet to the SNAT external address, which is located on the BIG/ip Controller. If a SNAT were not defined, the server would return the packets directly to the client without giving the BIG/ip Controller the opportunity to translate the source address from the server address back to the virtual server. The client would not recognize the packet if the source address of the returning packet is the IP address of the real server because the client sent its packets to the IP address of the *virtual* server.

```
bipipe snat map client1 to 192.168.13.99 exp0
```

Replace **client1**with the actual name of the client in your configuration.

Setting up 802.1q VLAN trunk mode

The BIG/ip Controller supports VLANs based on the IEEE 802.1q Trunk mode on BIG/ip Controller internal interfaces. VLAN tags are not supported on the external interfaces. You can define a single VLAN tag for each IP address defined for each BIG/ip Controller internal interface. This includes node network addresses, administrative addresses, shared administrative aliases, and additional aliases.

Note

In order for 802.1q VLAN trunk mode to operate on a BIG/ip Controller interface, all IP addresses on that interface must have a VLAN tag.

In order to use VLAN tags, you must edit **/etc/netstart**. Additionally, if you plan to use VLAN tags on a redundant BIG/ip system, you must add VLAN tags to the shared IP aliases in BIG/db using the **bigpipe ipalias** command.

Adding VLAN tag definitions to /etc/netstart

The VLAN tag ID for the network needs to be specified at the time the network address is defined for a particular internal interface. This is done by extending the **additional_xxx** definition for the internal interface (where **xxx** is the interface name, such as **exp0**, **exp1**, or **hmc0**). For example, if you have an internal interface IP defined as:

```
ipaddr_exp1="10.1.1.1"
netmask_exp1="255.0.0.0"
linkarg_exp1="media 100BaseTX,FDX"
additional_exp1="broadcast 10.255.255.255"
```

To define a VLAN tag ID 12 for this network (10.0.0.0), extend the **additional_exp1** definition in the following manner:

```
additional_exp1="broadcast 10.255.255.255 vlan 12"
```

Do this for each internal interface for which you want to define a VLAN tag ID.

Adding VLAN tag definitions to BIG/db

For a redundant configuration, the BIG/db database contains the shared IP addresses for the internal and external interfaces for the BIG/ip Controller. If you plan to use VLAN tags on a redundant BIG/ip system, you must the shared IP addresses to this database. Use the following syntax to add VLAN tag definitions to BIG/db.

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

For example, using the previous example, this line is extended with the same VLAN tag defined for its primary address, in this case 12:

```
bigpipe ipalias exp1 10.1.1.10 netmask 255.0.0.0 broadcast
   10.255.255.255 tag 12
```

Configuring multiple VLANs on one interface

In order to set up multiple VLANs on the same interface, you need to add a new IP address for the interface. The BIG/ip Controller only supports one VLAN ID per network.

For example, to support an additional network, 12.0.0.0, with a VLAN tag ID of 15 on the same interface, add the following line to your **/etc/netstart** file after the **ifconfig** command:

```
/sbin/ifconfig exp1 add 12.1.1.1 netmask 255.0.0.0 media
   100BaseTX,FDX broadcast 12.255.255.255 vlan 15
```

Note that you must add a shared address to the BIG/db file with the **bigpipe ipalias** command in a redundant BIG/ip system:

```
 bigpipe ipalias exp1 12.1.1.1 netmask 255.0.0.0 broadcast
   12.255.255.255 tag 15
```

To enable or disable VLAN tags on the command line

Once you have added VLAN tags, you can use the **bigpipe interface** command to enable, disable, or show the current settings for the interface. To globally enable or disable the VLAN tags for an internal interface, use the following syntax:

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

For example, use the following command to enable VLAN tags on the interface **exp1**:

```
bigpipe interface exp1 vlans enable
```

Using ifconfig to add another VLAN

You must use the **ifconfig** command to define multiple, different VLAN tagged networks on the same interface. For example, use the following syntax to add a new VLAN tagged network on the same interface:

```
ifconfig exp1 add <address> netmask <mask> broadcast <address> vlan
   <tag>
```

Note that the BIG/ip Controller allows one VLAN tag per network. In a redundant configuration, you need to add a new shared address on the new network with the identical VLAN tag ID in the BIG/db database with the **bigpipe ipalias** command.

You can also use **ifconfig** to display VLAN information for the interface **exp1** with the following command:

```
ifconfig exp1
```

Using netstat to view VLAN tags

You can also use the **netstat** utility to display VLAN tag information with the route table for the BIG/ip Controller. Use the following syntax to display VLAN tag information with **netstat**:

```
netstat -nrT
```

WARNING

802.1q VLAN tags are currently supported only on Intel EtherExpressPro NICs and SysKonnect SX 9843 and SX 9844 NICs.

Disabling and enabling VLAN tags using the F5 Configuration utility

You can use the F5 Configuration utility to enable or disable VLAN tags once they are configured on the BIG/ip Controller.

1. In the navigation pane, select **NICs**.
   The Network Interface Cards screen opens.
2. In the **Network Interface Cards** list, select the internal NIC for which you want to enable VLAN tags.
   The Network Interface Card Properties screen opens.
3. In the **Network Interface Card Properties** screen, navigate to the **Enable VLANs** check box.
   Click the Enable VLANs check box to enable the VLAN tags for the interface. Clear the check box to disable VLAN tags on the interface.
4. Click the **Apply** button.

Note

*You can only enable or disable VLAN tags in the F5 Configuration utility. VLAN tags must be configured by adding VLAN tag values to the **/etc/netstart** file (and the **BIG/db** with the **bigpipe ipalias** command for redundant configurations). The F5 Configuration utility can only enable or disable VLAN tags that have been configured in those files.*

8

Monitoring and Administration

- **Monitoring utilities provided on the BIG/ip Controller**
- **Using the BIG/pipe command utility as a monitoring tool**
- **Working with the BIG/stat utility**
- **Working with the BIG/top utility**
- **Working with the Syslog utility**
- **Removing and returning items to service**
- **Viewing system statistics and log files**
- **Printing the connection table**
- **Changing passwords for the BIG/ip Controller**
- **Working with the BIG/db database**

Monitoring and administration utilities provided on the BIG/ip Controller

The BIG/ip platform provides several utilities for monitoring and administration of the BIG/ip Controller. You can monitor system statistics, as well as statistics specific to virtual servers and nodes, such as the number of current connections, and the number of packets processed since the last reboot.

The BIG/ip platform provides the following monitoring and configuration and administration utilities:

- **BIG/pipe**
  If you type certain BIG/pipe commands, such as **bigpipe vip** or **bigpipe node,** and use the **show** keyword in the command, the command displays statistical information about the elements that you configure using that command. You can also use **bigpipe** commands to selectively reset any statistic collected by the BIG/ip Controller.
- **The F5 Configuration utility**
  You can use the F5 Configuration utility to configure any feature on the BIG/ip Controller. You can reset any statistic, or all statistics, for virtual servers, nodes, NATs, and SNATs in the F5 Configuration utility.
- **BIG/stat**
  This utility is provided specifically for statistical monitoring of virtual servers, nodes, NATs, SNATs, and services. One benefit of using BIG/stat is that it allows you to customize the display of statistical information.
- **BIG/top**
  BIG/top provides statistical monitoring. You can set a refresh interval, and you can specify a sort order.
- **Syslog**
  Syslog is the standard UNIX system logging utility, which monitors critical system events, as well as configuration changes made on the BIG/ip Controller.
- **BIG/db**
  BIG/db is a database that contains various configuration information for the BIG/ip Controller.

Using the BIG/pipe command utility as a monitoring tool

Using the BIG/pipe utility, you can view information about the BIG/ip Controller itself, as well as elements such as virtual servers, virtual addresses, virtual ports, nodes, and node addresses. Typically, the BIG/pipe utility provides the following statistics:

- Current number of connections
- Maximum number of concurrent connections
- Total number of connections since the last system reboot
- Total number of bits (inbound, outbound, total)
- Total number of packets (inbound, outbound, total)

Monitoring the BIG/ip Controller

The **bigpipe summary** command displays performance statistics for the BIG/ip Controller itself. This display summary includes current usage statistics, such as the amount of time a BIG/ip Controller has been running since the last reboot. Type the following command:

```
bigpipe summary
```

The performance statistics display in the format shown in Figure 8.1 (the output has been truncated for this example).

```
BIG/ip total uptime           = 1 (day) 4 (hr) 40 (min) 8 (sec)
   BIG/ip total uptime (secs)    = 103208
   BIG/ip total # connections    = 0
   BIG/ip total # pkts           = 0
   BIG/ip total # bits           = 0
   BIG/ip total # pkts(inbound)  = 0
   BIG/ip total # bits(inbound)  = 0
   BIG/ip total # pkts(outbound) = 0
   BIG/ip total # bits(outbound) = 0
   BIG/ip error no nodes available          = 0
   BIG/ip tcp port deny                     = 0
   BIG/ip udp port deny                     = 0
   BIG/ip 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 8.1 *The BIG/pipe summary display screen*

Table 8.1 contains descriptions of each individual statistic included in the summary display screen.

Statistic	Description
`total uptime`	Total time elapsed since the BIG/ip Controller was last booted.
`total uptime (secs)`	Total uptime displayed in seconds.
`total # connections`	Total number of connections handled.
`total # pkts`	Total number of packets handled.
`total # bits`	Total number of bits handled.
`total # pkts (inbound)`	Total number of incoming packets handled.
`total # bits (inbound)`	Total number of incoming bits handled.
`total # pkts (outbound)`	Total number of outgoing packets handled.
`total # bits (outbound)`	Total number of outgoing bits handled.
`error no nodes available`	The number of times the BIG/ip Controller tries to make a connection to a node, but no nodes are available.
`tcp port deny`	The number of times a client attempted to connect to an unauthorized TCP port on the BIG/ip Controller (unauthorized port and source IP are logged in the syslog).
`udp port deny`	The number of times a client attempted to connect to an unauthorized UDP port on the BIG/ip Controller (unauthorized port and source IP are logged in the syslog).
`vip tcp port deny`	The number of times a client attempted to connect to an unauthorized TCP port on a virtual address (unauthorized port and source IP are logged in the syslog).
`vip udp port deny`	The number of times a client attempted to connect to an unauthorized UDP port on a virtual address (unauthorized port and source IP are logged in the syslog).
`max connections deny`	The total number of connections denied because the maximum number of connections allowed was exceeded.
`vip duplicate syn ssl`	The number of duplicate connection attempts to existing SSL connections from the same client.
`vip duplicate syn wrong dest`	The number of duplicate connection attempts from the same client (address and port combination) to a different virtual server.
`vip duplicate syn node down`	The number of duplicate connection attempts to a server that is down when a connection to the server was made previously.

Table 8.1 *BIG/pipe monitoring statistics*

Statistic	Description
`vip maint mode deny`	The number of times a connection to a virtual server was denied while the BIG/ip Controller is in maintenance mode.
`virtual addr max connections deny`	The number of virtual address connections dropped because the maximum number of connections was exceeded.
`virtual path max connections deny`	The number of virtual path connections dropped because the maximum number of connections was exceeded.
`vip non syn`	The number of packets received which are not connection requests, and are destined to a virtual address, but not a valid virtual server (port).
`error vip fragment no port`	The number of IP fragments for which there is no port.
`error vip fragment no conn`	The number of IP fragments for which there is no connection.
`error standby shared drop`	The number of packets destined to the shared IP address in a redundant system that are received and ignored by the standby system.
`dropped inbound`	The total number of inbound packets dropped by the BIG/ip Controller.
`dropped outbound`	The total number of outbound packets dropped by the BIG/ip Controller.
`reaped`	The total number of connections that timed-out, and are deleted by the BIG/ip Controller.
`ssl reaped`	The total number of SSL session ID records that timed-out, and are closed by the BIG/ip Controller.
`persist reaped`	The total number of persistence records that timed-out, and are closed by the BIG/ip Controller.
`udp reaped`	The total number of UDP connections that timed-out, and are closed by the BIG/ip Controller.
`malloc errors`	The number of times a connection could not be created because the system is low on memory.
`mem pool total`	The total amount of memory available in all combined memory pools.
`mem pool used`	The total amount of memory, in all combined memory pools, in use by the BIG/ip Controller.
`mem percent used`	The total percentage of memory in use by all combined memory pools.

Table 8.1 *BIG/pipe monitoring statistics*

Resetting statistics on the BIG/ip Controller

The **bigpipe** commands allow you to selectively reset any statistic on the BIG/ip Controller. The statistics you can reset selectively include:

- Virtual address
- Virtual server
- Node address
- Node server
- Virtual port
- Network address translations (NATs)
- Global statistics

When you reset one of these items, the packets in, packets out, bytes in, and bytes out counters of the target item are reset to zero. The max connection count counter is also reset. The current connections counter is not reset and the total connections counter is set equal to the number of current connections.

Note

The statistics are reset for the specified items only. Statistics for dependent items, such as node servers for a given virtual address, are not modified by these commands. The only exception is the global statistics reset option which resets traffic statistics for all items. After an item-level reset, statistics for all other dependent items do not add up.

You can create an audit trail for reset events by setting an optional system control variable. You can set this variable to generate a syslog log entry. To set this variable, type the following command:

```
sysctl -w bigip.verbose_log_level=4
```

Resetting statistics for virtual servers and virtual addresses

Use the following syntax to reset statistics for the virtual address specified by the IP address **<virtual ip>**.

```
bigpipe vip <virtual ip> stats reset
```

For example, if you want to reset statistics for the virtual address 172.20.1.100, type the following command:

```
bigpipe vip 172.20.1.100 stats reset
```

If you want to reset statistics for a list of virtual addresses, type the command with a list of addresses separated by spaces:

```
bigpipe vip 172.20.1.100 172.20.1.101 172.20.1.102 stats reset
```

If you want to reset statistics for all virtual servers, use the following command:

```
bigpipe vip stats reset
```

Use the following syntax to reset statistics for the virtual server IP:Port combination **<virtual IP>:<port>**.

```
bigpipe vip <virtual IP>:<port> stats reset
```

For example, if you want to reset statistics for the virtual address/port combination 172.20.1.100:80, type the following command:

```
bigpipe vip 172.20.1.100:80 stats reset
```

If you want to reset statistics for a list of virtual address/port combinations, type the command with the list of addresses separated by spaces:

```
bigpipe vip 172.20.1.100:80 172.20.1.100:23 172.20.1.101:80 stats
   reset
```

Resetting statistics for node servers and node addresses

Use the following syntax to reset statistics for all node addresses and node servers.

```
bigpipe node stats reset
```

You can reset statistics for the node address specified by the IP address **<node IP>**.

```
bigpipe node <node IP> stats reset
```

For example, to reset the statistics for the node address 10.1.1.1, use the following syntax:

```
bigpipe node 10.1.1.1 stats reset
```

If you want to reset statistics for a list of node addresses, type the command with the list of addresses separated by spaces:

```
bigpipe node 10.1.1.1 10.1.1.2 10.1.1.3 stats reset
```

Use the following syntax to reset statistics for the node server specified by the IP:Port combination **<node IP>:<port>**.

```
bigpipe node <node IP>:<port> stats reset
```

For example, to reset the statistics for the node server 10.1.1.1:80, use the following syntax:

```
bigpipe node 10.1.1.1:80 stats reset
```

If you want to reset statistics for a list of node server addresses, type the command with the list of addresses separated by spaces:

```
bigpipe node 10.1.1.1:80 10.1.1.2:23 stats reset
```

Resetting statistics for virtual ports

Use the following syntax to reset statistics for all virtual ports:

```
bigpipe port stats reset
```

Use the following syntax to reset statistics for the virtual port **<port>**. You can specify a list of virtual ports separated by spaces.

```
bigpipe port <port> stats reset
```

For example, to reset the statistics for the virtual port 80, use the following command:

```
bigpipe port 80 stats reset
```

To reset the statistics for a list of virtual ports, use the following syntax:

```
bigpipe port 23 80 443 stats reset
```

Resetting statistics for network address translations (NATs)

Use the following syntax to reset statistics for all NATs:

```
bigpipe nat stats reset
```

Use the following syntax to reset statistics for the NAT specified by the IP address **<nat IP>**.

```
bigpipe nat <nat IP> stats reset
```

For example, to reset the statistics for the NAT 172.20.3.101, use the following command:

```
bigpipe nat 172.20.3.101 stats reset
```

To reset the statistics for a list of NATs, use the following command where NATs are separated by spaces:

```
bigpipe nat 172.20.3.101 172.20.3.102 stats reset
```

Resetting statistics for secure network address translations (SNATs)

Use the following syntax to reset statistics for all SNATs:

```
bigpipe snat stats reset
```

Use the following syntax to reset statistics for the SNAT specified by the IP address <**snat IP**>.

```
bigpipe snat <snat IP> stats reset
```

For example, to reset the statistics for the SNAT 172.20.3.101, use the following command:

```
bigpipe snat 172.20.3.101 stats reset
```

To reset the statistics for a list of SNATs, use the following command where SNATs are separated by spaces:

```
bigpipe snat 172.20.3.101 172.20.3.102 stats reset
```

Resetting global statistics

Use the following command to reset all statistics for all items.

```
bigpipe global stats reset
```

To reset any statistic in the F5 Configuration utility

New Reset buttons have been added to the following tables in the F5 Configuration utility:

- Virtual address
- Virtual server
- Node address
- Node server
- Virtual port

- Network address translations (NATs)
- Global statistics

To reset a statistic for a particular item, click the **Reset** button next to the item in one of these tables.

Viewing the status of the interface cards

The **bigpipe interface** command displays the current status and the settings for external and internal interface cards. You can also use the **bigpipe interface** command to view information for a specific interface card, using the following command syntax:

```
interface <ifname>
```

Monitoring virtual servers, virtual addresses, and services

You can use different variations of the **bigpipe vip** command, as well as the **bigpipe port** command, to monitor information about virtual servers, virtual addresses, and services managed by the BIG/ip Controller.

Displaying information about virtual servers and virtual addresses

The **bigpipe vip** command displays the status of virtual servers (**up**, **down**, **unchecked**, or **disabled**), the current number of connections to each virtual server, and the status of the member nodes that are included in each virtual server mapping. The status for individual member nodes includes whether the node is **up**, **down**, **unchecked**,

or **disabled**, and also includes the cumulative count of packets and bits received and sent by the node on behalf of the virtual server. The BIG/ip Controller displays the statistics as shown in Figure 8.2.

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

Figure 8.2 *Virtual server statistics*

If you want to view statistical information about one or more specific virtual servers, simply include the virtual servers in the **bigpipe vip show** command as shown below:

`bigpipe vip <virt addr>:<port>... <virt addr>:<port> show`

If you want to view statistical information about traffic going to one or more virtual addresses, specify only the virtual address information in the command:

`bigpipe vip <virt addr>... <virt addr> show`

Displaying information about services

The **bigpipe port show** command allows you to display information about specific virtual ports managed by the BIG/ip Controller. You can use the command to display information about all virtual services, or you can specify one or more particular virtual services.

To view information about all virtual services, use the following syntax:

`bigpipe port show`

To view statistical information about one or more specific virtual services, simply include the service names or port numbers as shown below:

```
bigpipe port <port>... <port> show
```

Monitoring nodes and node addresses

The **bigpipe node** command displays the status of all nodes configured on the BIG/ip Controller. The information includes whether or not the specified node is **up**, **down**, **disabled**, or **unchecked**, and the number of cumulative packets and bits sent and received by each node on behalf of all virtual servers. The BIG/ip Controller displays the statistical information as shown in Figure 8.3.

```
bigpipe node
    NODE 192.168.103.20         UP
     |   (cur, max, limit, tot) = (0, 0, 0, 0)
     |   (pckts,bits) in = (0, 0), out = (0, 0)
     +---PORT 23            UP
         (cur, max, limit, tot) = (0, 0, 0, 0)
         (pckts,bits) in = (0, 0), out = (0, 0)
```

Figure 8.3 *Node statistics screen*

If you want to view statistical information about one or more specific nodes, simply include the nodes in the **bigpipe node show** command as shown below:

```
bigpipe node <node addr>:<port>... <node addr>:<port> show
```

If you want to view statistical information about traffic going to one or more node addresses, specify only the node address information in the command:

```
bigpipe node <node addr>... <node addr> show
```

Monitoring NATs

The **bigpipe nat show** command displays the status of the NATs configured on the BIG/ip Controller. The information includes the number of cumulative packets and bits sent and received by each NAT. 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:

```
bigpipe nat <node addr> [...<node addr>] show
```

An example of the output for this command is in Figure 8.4.

```
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 8.4 *NAT statistics*

Monitoring SNATs

The **bigpipe snat show** command displays the status of the SNATs configured on the BIG/ip Controller. The information includes connections and global SNAT settings. Use the following **bigpipe** command to show SNAT mappings:

```
bigpipe snat [<SNAT addr>] [...<SNAT addr>] show
```

```
bigpipe snat show
```

Use the following command to show the current SNAT connections:

```
bigpipe snat [<SNAT addr>] [...<SNAT addr>] dump [ verbose ]
```

```
bigpipe snat dump [ verbose ]
```

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

The following command prints the global SNAT settings:

```
bigpipe snat globals show
```

Working with the BIG/stat utility

BIG/stat™ is a utility that allows you to quickly view the status of the following elements:

- Virtual servers
- Services
- Nodes
- Network address translations (NATs)

You can customize the BIG/stat utility statistics display. For example, you can customize your output to display statistics for a single element, or for selected elements. You can set the display to automatically update at time intervals you specify.

The **bigstat** command accepts one or more options, which allow you to customize the statistical display. When you use the **bigstat** command without specifying any options, the BIG/stat utility displays the list of virtual servers, services, nodes, NATs, and SNATs only one time. The basic command syntax is:

```
bigstat [ options...]
```

The following table, Table 8.2, describes the options that you can use in the **bigstat** command.

Option	Description
`-bigip`	Displays totals for the BIG/ip Controller overall.
`-c <count>`	Sets the interval at which new information is displayed.
`-h and -help`	Displays the help options.
`-n`	Displays data in numeric format.
`-nat`	Displays network address table (NAT) entries only.
`-no_viptot`	Removes virtual server totals from the display.
`-no_nodetot`	Removes node totals from the display.
`-node`	Displays nodes only.

Table 8.2 *The **bigstat** command options*

Option	Description
-port	Displays ports only.
-v	Displays version information.
-vip	Displays virtual servers only.

Table 8.2 *The* ***bigstat*** *command options*

Working with the BIG/top utility

BIG/top™ is a real-time statistics display utility. The display shows the date and time of the latest reboot and lists activity in bits, bytes, or packets. Similar to BIG/stat, the BIG/top utility accepts options which allow you to customize the display of information. For example, you can set the interval at which the data is refreshed, and you can specify a sort order. The BIG/top displays the statistics as shown in the following figure, Figure 8.5.

```
                    |  bits  since       |  bits in prior     |   current
                    |  Nov 28 18:47:50   |  3 seconds         |   time
BIG/ip      ACTIVE |---In----Out---Conn-|---In----Out---Conn-|   00:31:59
227.19.162.82          1.1G  29.6G 145     1.6K     0      0

VIP ip:port         |---In----Out---Conn-|---In----Out---Conn-|-Nodes Up--
217.87.185.5:80        1.0G  27.4G 139.6K   1.6K     0      0       2
217.87.185.5:20       47.5M  2.1G    3.1K     0      0      0       2
217.87.185.5:20       10.2M 11.5M    2.6K     0      0      0       2

NODE  ip:port       |---In----Out---Conn-|---In----Out---Conn-|--State----
129.186.40.17:80     960.6M  27.4G  69.8K   672      0      0      UP
129.186.40.17:20      47.4M   2.1G   3.1K     0      0      0      UP
129.186.40.18:80     105.3M 189.0K  69.8K   1.0K     0      0      UP
129.186.40.17.21       9.4M  11.1M   1.3K     0      0      0      UP
129.186.40.18:21     700.8K 414.7K   1.3K     0      0      0      UP
129.186.40.18:20       352    320      1      0      0      0      UP
```

Figure 8.5 *The BIG/top screen display*

Using BIG/top command options

The **bigtop** command uses the syntax below, and it supports the options outlined in Table 8.3:

```
bigtop  [options...]
```

Option	Description
-bytes	Displays counts in bytes (the default is bits).
-conn	Sorts by connection count (the default is to sort by byte count).
-delay <value>	Sets the interval at which data is refreshed (the default is four seconds).
-delta	Sorts by count since last sample (the default is to sort by total count).
-help	Displays BIG/top help.
-nodes <value>	Sets the number of nodes to print (the default is to print all nodes).
-nosort	Disables sorting.
-once	Prints the information once and exits.
-pkts	Displays the counts in packets (the default is bits).
-scroll	Disables full-screen mode.
-vips <value>	Sets the number of virtual servers to print (the default is to print all virtual servers).

Table 8.3 *BIG/top command options*

Using runtime commands in BIG/top

Unless you specified the **-once** option, the BIG/top utility continually updates the display at the rate indicated by the **-delay** option, and you can also use the following runtime options at any time:

- The **u** option cycles through the display modes; bits, bytes, and packets.
- The **q** option quits the BIG/top utility.

Working with the Syslog utility

The BIG/ip Controller supports logging via the **Syslog** utility. The logs are generated automatically, and saved in user-specified files. These logs contain all changes made to the BIG/ip Controller

configuration, such as those made with the **bigpipe vip** command, or other BIG/pipe commands, as well as all critical events that occur in the system.

◆ Note

You can configure the Syslog utility to send email or activate pager notification based on the priority of the logged event. For more information, see Working with the Syslog utility, on page 8-16

The Syslog log files track system events based on information defined in the **/etc/syslog.conf** file. You can view the log files in a standard text editor, or with the **less** file page utility.

Sample log messages

Table 8.4 shows sample log messages give you an idea of how the Syslog utility tracks events that are specific to the BIG/ip Controller.

Sample message	Description
`bigd: allowing connections on port 20`	A user specifically allowed connections on virtual port 20
`bigd: node 192.168.1.1 detected up`	The 192.168.1.1 node address was successfully pinged by the BIG/ip Controller
`bigd: added service port 20 to node 192.168.1.1`	A user defined a new node, 192.168.1.1:20.
`kernel: security: port denial 207.17.112.254:4379 -> 192.168.1.1:23`	A client was denied access to a specific port. The client is identified as coming from 207.17.112.254:4379, and the destination node is 192.168.1.1:23.

Table 8.4 *Sample Syslog messages*

Removing and returning items to service

Once you have completed the initial configuration on the BIG/ip Controller, you may want to temporarily remove specific items from service for maintenance purposes. For example, if a specific

network server needs to be upgraded, you may want to disable the nodes associated with that server, and then enable them once you finish installing the new hardware and bring the server back online.

If you specifically disable the nodes associated with the server, the BIG/ip Controller allows the node to go down only after all the current connections are complete. During this time, the BIG/ip Controller does not attempt to send new connections to the node. Although the BIG/ip Controller's monitoring features would eventually determine that the nodes associated with the server are down, specifically removing the nodes from service can prevent interruptions on long duration client connections.

You can remove the entire BIG/ip Controller from service, or you can remove the following individual items from service:

- Virtual servers
- Virtual addresses
- Virtual ports
- Nodes
- Node addresses

Removing the BIG/ip Controller from service

The BIG/ip platform offers a Maintenance mode, which allows you to remove the BIG/ip Controller from network service. This is useful if you want to perform hardware maintenance, or make extensive configuration changes. When you activate Maintenance mode, the BIG/ip Controller no longer accepts connections to the virtual servers it manages. However, the existing connections are allowed to finish processing so that current clients are not interrupted.

The **bigpipe maint** command toggles the BIG/ip Controller into or out of Maintenance mode. Use the following command to put the BIG/ip Controller in maintenance mode:

```
bigpipe maint
```

If the BIG/ip Controller runs in Maintenance mode for less than 20 minutes and you return the machine to the normal service, the BIG/ip Controller quickly begins accepting connections. However, if the BIG/ip Controller runs in Maintenance mode for more than 20 minutes, returning the Controller to service involves updating all network ARP caches. This process can take a few seconds, but you can speed the process up by reloading the **/etc/bigip.conf** file using the following command:

```
bigpipe -f /etc/bigip.conf
```

To activate maintenance mode in the F5 Configuration utility

1. In the navigation pane, click the BIG/ip logo.
   The BIG/ip System Properties screen opens.
2. Click the **Maintenance Mode** box.
3. Click on the **Apply** button.

Removing individual virtual servers, virtual addresses, and ports from service

The BIG/ip Controller also supports taking only selected virtual servers, addresses, or ports out of service, rather than removing the BIG/ip Controller itself from service. Each BIG/pipe command that defines virtual servers and their components supports **enable** and **disable** keywords, which allow you to remove or return the elements from service.

When you remove a virtual address or a virtual port from service, it affects all virtual servers associated with the virtual address or virtual port. Similarly, if you remove a node address from service, it affects all nodes associated with the node address.

Enabling and disabling virtual servers and virtual addresses

The **bigpipe vip** command allows you to enable or disable individual virtual servers, as well as virtual addresses. To enable or disable a virtual server, type the appropriate command:

```
bigpipe vip <virtual addr>:<virtual port> enable
bigpipe vip <virtual addr>:<virtual port> disable
```

To enable or disable a virtual address, type the appropriate command:

```
bigpipe vip <virtual addr> enable
bigpipe vip <virtual addr> disable
```

Enabling and disabling virtual ports

The **bigpipe port** command allows you to allow or deny traffic on a virtual port:

```
bigpipe port <virtual port> enable
bigpipe port <virtual port> disable
```

Removing individual nodes and node addresses from service

Enabling and disabling nodes and node addresses

The **bigpipe node** command allows you to enable or disable individual nodes, as well as node addresses.

To enable or disable a **node**, type the appropriate command:

```
bigpipe node <node addr>:<node port> enable
bigpipe node <node addr>:<node port> disable
```

To enable or disable a **node address**, type the appropriate command:

```
bigpipe node <node addr> enable
bigpipe node <node addr> disable
```

Viewing the currently defined virtual servers and nodes

When used with the **show** parameter, BIG/pipe commands typically display currently configured elements. For example, the **bigpipe vip show** command displays all currently defined virtual servers,

and the **bigpipe node** command displays all nodes currently included in virtual server mappings. For additional information about using BIG/pipe commands on the BIG/ip Controller, see the ***BIG/ip Reference Guide***, *BIG/pipe Command Reference*.

Viewing system statistics and log files

The F5 Configuration utility allows you to view a variety of system statistics and system log files. Note that from each statistics screen, you can access property settings for individual virtual servers, nodes, IP addresses, and ports by selecting the individual item in the statistics table.

Viewing system statistics

The F5 Configuration utility allows you to view the following statistical information:

- BIG/ip system statistics, including the elapsed time since the last system reboot, the number of packets and connections handled by the system, and the number of dropped connections.
- Virtual servers, including virtual servers, virtual address only, or virtual ports only.
- Nodes, including nodes, node addresses only, or node ports only.
- NAT statistics, such as the number of packets handled by each NAT.
- SNAT statistics, such as SNAT mappings.
- IP filter statistics, including the number of packets accepted and rejected by individual IP filters.
- Rate filter statistics, including the number of bits passed through, delayed, and dropped by individual rate filters.
- Information about illegal connection attempts, such as the source IP addresses from which the illegal connection is initiated.

Statistics are displayed in real-time. You can specify the update frequency by setting an interval (in seconds), and then clicking **Update**.

Viewing log files

The F5 Configuration utility allows you to display three different log files:

- The BIG/ip system log, which displays standard UNIX system events
- The BIG/ip log, which displays information specific to BIG/ip events, such as defining a virtual server
- The Pinger log, which displays status information determined by each node ping issued by the BIG/ip Controller

Printing the connection table

The BIG/pipe command line utility also offers a useful diagnostic tool that prints the list of current connections. Normally, the **bigpipe conn** command prints the client, virtual server, and node addresses.

Changing passwords for the BIG/ip Controller

During the First-Time Boot utility, you define a password that allows remote access to the BIG/ip Controller, and you also define a password for the BIG/ip web server. You can change these passwords at any time.

Changing the BIG/ip Controller password

1. At the BIG/ip Controller command line prompt, log on as the root user and use the **passwd** command.
2. At the password prompt, enter the password you want to use for the BIG/ip Controller and press **Return**.
3. To confirm the password, retype it and press **Return**.

Changing passwords and adding new user IDs for the BIG/ip web server

You can create new users for the BIG/ip web server in the F5 Configuration utility.

Creating new users for the BIG/ip web server

The user accounts you create in the F5 Configuration utility can have full, partial, or read-only access to the BIG/ip Controller.

To create user accounts in the F5 Configuration utility

1. In the navigation pane, click User Admin.
   The User Administration screen opens.
2. In the Add User section, type the following information.
   - **User ID**
     Type the user ID you want to assign the user.
   - **Password**
     Type the password you want to assign the user.
   - **Retype Password**
     Retype the password you want to assign the user.
3. In the Access Level list, select the access level for the user. The access levels available are:
   - **Read Only**
     This access level allows the user only to view information in the F5 Configuration utility. Users with

this access level do not have access to **Add** buttons, certain toolbar items, **Apply** buttons, or **Remove** buttons.

- **Partial Read/Write**
  In addition to allowing the user to view information, a Partial Read/Write user can also change the status of node addresses to either **enabled** or **disabled**.
- **Full Read/Write**
  This access level provides the user with full access to all administrative tasks.

4. After you select the access level for the user, click the **Add** button.

The Current User list on the User Administration screen contains all users configured to access the F5 Configuration utility. You can delete any user added through the F5 Configuration utility by clicking the **Remove** button next to the user in the list. The BIG/ip web server administrator account you created with the First-Time Boot utility shows up in this list. However, you cannot edit or delete this account from the F5 Configuration utility. To edit this account, you must run the **reconfig-httpd** command line utility. For more information about this utility, see the ***BIG/ip Controller Reference Guide***, *BIG/ip Controller Configuration Utilities.*

Working with the BIG/db database

The BIG/db™ database holds certain configuration information for the BIG/ip Controller. Most BIG/ip Controller utilities currently use the configuration stored in BIG/db. The **bigdba** utility is provided for loading configuration information into BIG/db. An additional **default.txt** file is included with the BIG/ip Controller which contains default information you can load into the BIG/db database.

Using bigdba

Use the **bigdba** utility to modify the keys in the BIG/db database. The **bigdba** utility allows you to edit the database and insert and modify keys and values. All values are entered into BIG/db as strings.

Accessing and modifying the default database

The default BIG/db database is created when you run the First-Time Boot utility. To use **bigdba** from the command line run **bigdba** with the name of the database.

```
bigdba
Database "/var/f5/bigdb/user.db" opened.
```

Using bigdba commands

Table 8.5 describes the commands you can use in the **bigdba** utility.

Command	Description
subkey <string> **sk <string>**	Add subkey to current key level
p <name> **p <regex>**	Print the value stored at **<name>**. If you use a regular expression **<regex>**, all records under the current subkey which match the regular expression print.
up	Back up one subkey
up <string>	Back up through subkey **<string>**
d <string> **d <regex>**	Delete value stored under current key **<string>**. If you use a regular expression **<regex>**, all records under the current subkey which match the regular expression are deleted.
<name> = <value>	Store **<value>** under name **<name>** within the current key
set confirm on	Confirm delete operations
set confirm off	Do not confirm deletions
set comments on	Show comments. By default, comments are **off**
set comments off	Do not show comments. By default, comments are **off**
dump <file>	Dumps the database to the file name specified

Table 8.5 *The **bigdba** commands*

Command	Description
`load <file>`	Loads the database with the file specified

`quit`
`q`
`EOF`
	Quits the **bigdba** utility

`help` `?`	Display the help text for the **bigdba** utility

Table 8.5 *The **bigdba** commands*

Working with the default.txt file

The **default.txt** file documents the keys that are valid in the BIG/store database. This file is located at **/var/f5/bigdb/default.txt**. This text file, which can be loaded with the **bigdba** program, contains all the possible database keys, comments that document these keys, and the default values used by programs that run on the BIG/ip Controller.

◆ Note

The values in the default.txt file are default values, several of the keys listed are not present in the BIG/db database.

The **default.txt** file is intended to serve as documentation only. Some of the records, such as those that represent IP addresses and port numbers, need to be set to values other than the default values for the system to work. Additionally, some of the key names listed are wildcard keys. These keys are not valid key names.

If you want to load **default.txt** into the BIG/db database, it is recommended that you dump the existing database to another text file. Make a copy of **default.txt**, and then edit the copy so that the records which are present in your dump file match the values contained in the default.txt file. After the values match, you can load the edited copy of **default.txt**.

For a complete list of the keys available in the BIG/db, see the ***BIG/ip Controller Reference Guide***, *BIG/db Configuration Keys*.

9

Configuring SNMP

- **Working with SNMP on the BIG/ip® Controller**
- **Configuring SNMP on the BIG/ip Controller**

Working with SNMP on the BIG/ip Controller

This chapter covers the management and configuration tasks for the simple network management protocol (SNMP) agent and management information bases (MIBs) available with the BIG/ip Controller.

◆ Note

The SNMP agent must be configured on the BIG/ip Controller in order to use the F5 Networks see/IT Network Manager.

The BIG/ip SNMP agent and MIBs allow you to manage the BIG/ip Controller by configuring traps for the SNMP agent or polling the controller with your standard network management station (NMS).

You can configure the BIG/ip SNMP agent to send traps to your management system with the F5 Configuration utility. You can also set up custom traps by editing several configuration files.

Security options are available that let you securely manage information collected by the BIG/ip SNMP agent, including Community names, TCP wrappers, and View access control mechanism (VACM).

Configuring SNMP on the BIG/ip Controller

There are seven basic tasks you must complete in order to use SNMP on the BIG/ip Controller.

❖ **Download the MIBs**
Download the BIG/ip MIBs and load them into your network management station.

❖ **Setup administrative access**
Configure **/etc/hosts.allow** to allow administrative access to the SNMP agent.

- **Configure snmpd.conf**
  This file configures the SNMP agent. You can configure this file with the F5 Configuration utility, or by editing it directly with a text editor.
- **Configure rc.local**
  Configure **/etc/rc.local** to automatically start the SNMP agent.
- **Configure snmptrap.conf**
  The configuration in **/etc/snmptrap.conf** determines which messages generate traps and what those traps are.
- **Configure syslog.conf**
  Configure **/etc/syslog.conf** to pipe specified message types through **checktrap.pl**.
- **Configure checktrap.pl**
  The BIG/ip Controller generates SNMP traps by piping **syslog** messages through the **/sbin/checktrap.pl**.

Downloading the MIBs

The BIG/ip platform includes a private BIG/ip SNMP MIB. This MIB is specifically designed for use with the BIG/ip Controller. You can configure the SNMP settings in the F5 Configuration utility, or on the command line.

SNMP management software requires that you use the MIB files associated with the device. You may obtain two MIB files from the BIG/ip directory **/usr/contrib/f5/mibs**, or you can download the files from the **Additional Software Downloads** section of the F5 Configuration utility home page.

- **LOAD-BAL-SYSTEM-MIB.txt** This is a vendor MIB that contains specific information for properties associated with specific F5 functionality (load balancing, NATs, and SNATs)
- **UCD-SNMP-MIB.txt** This is a MIB-II (RFC 1213) that provides standard management information.

For information about the objects defined in the **LOAD-BAL-SYSTEM-MIB.txt**, refer to the descriptions in the object identifier (OID) section of the MIB file. For information about the objects defined in **UCD-SNMP-MIB.txt**, refer to RFC 1213.

Understanding configuration file requirements

You need to make changes to several configuration files on the BIG/ip Controller before you use the SNMP agent. Once you change these configuration files, you need to restart the SNMP agent.

/etc/hosts.deny

This file must be present to deny by default all UDP connections to the SNMP agent. The contents of this file are as follows:

```
ALL : ALL
```

/etc/hosts.allow

The **/etc/hosts.allow** file is used to specify which hosts are allowed to access the SNMP agent. There are two ways to configure access to the SNMP agent with the **/etc/host.allow** file. You can type in an IP address, or list of IP addresses, that are allowed to access the SNMP agent, or you can type in an IP address and mask to allow a range of addresses in a subnetwork to access the SNMP agent.

For a specific list of addresses, type in the list of addresses you want to allow to access the SNMP agent. Addresses in the list must be separated by blank space or by commas. The basic syntax is as follows:

```
daemon: <IP address> <IP address> <IP address>
```

For example, you can type the following line which sets the SNMP agent to accept connections from the IP addresses specified:

```
bigsnmpd: 128.95.46.5 128.95.46.6 128.95.46.7
```

For a range of addresses, the basic syntax is as follows, where **daemon** is the name of the daemon, and **IP/MASK** specifies the network that is allowed access. The **IP** must be a network address:

```
daemon: IP/MASK
```

For example, you might use the following line which sets the **bigsnmpd** daemon to allow connections from the **128.95.46.0/255.255.255.0** address:

```
bigsnmpd: 128.95.46.0/255.255.255.0
```

The example above allows the 254 possible hosts from the network address **`128.95.46.0`** to access the SNMP daemon. Additionally, you may use the keyword **ALL** to allow access for all hosts or all daemons.

To allow access to the SNMP agent in the F5 Configuration utility

1. Click **SNMP** in the navigation pane.
   The SNMP Configuration screen opens.
2. In the BIG/ip SNMP Configuration screen, check **Enabled** to allow access to the BIG/ip SNMP agent.
3. In the Client Access Allow list section, type the following information:
   - **IP Address or Network Address**
     Type in an IP address or network address from which the SNMP agent can accept requests. Click the add (>>) button to add the address to the Current List. For a network address, type in a netmask.
   - **Netmask**
     If you type a network address in the IP Address or Network Address box, type the netmask for the network address in this box. Click the add (>>) button to add the network address to the Current List.
4. Click the **Apply** button.

/etc/snmpd.conf

The **/etc/snmpd.conf** file controls most of the SNMP agent. This file is used to set up and configure certain traps, passwords, and general SNMP variable names. A few of the necessary variables are listed below:

❖ **System Contact Name**
The System Contact is a MIB-II simple string variable defined by almost all SNMP boxes. It usually contains a user name, as well as an email address. This is set by the **syscontact** key.

- **Machine Location (string)**
  The Machine Location is a MIB-II variable that almost all boxes support. It is a simple string that defines the location of the box. This is set by the **syslocation** key.
- **Community String**
  The community string clear text password is used for basic SNMP security. This also maps to VACM groups, but for initial read/only access, it is limited to only one group.
- **Trap Configuration**
  Trap configuration is controlled by these entries in the **/etc/snmpd.conf** file:
  - **trapsink <host>**
    This sets the host to receive trap information. The **<host>** is an IP address.
  - **trapport <port>**
    This sets the port on which traps are sent. There must be one **trapport** line for each **trapsink** host.
  - **trapcommunity <community string>**
    This sets the community string (password) to use for sending traps. If set, it also sends a trap upon startup: **coldStart(0)**.
  - **authtrapenable <integer>**
    Setting this variable to **1** enables traps to be sent for authentication warnings. Setting it to **2** disables it.
  - **data_cache_duration <seconds>**
    This is the time in seconds data is cached. The default value for this setting is one second.

◆ Note

*To change the trap port, the **trapport** line must precede the **trapsink** line. If you use more then one **trapport** line, then there must be one **trapport** line before each trapsink line. The same follows for **trapcommunity**. If you use more then one **trapcommunity** line, then there must be one **trapcommunity** line before each **trapsink** line.*

To set SNMP properties in the F5 Configuration utility

1. Click **SNMP** in the navigation pane.
   The SNMP Configuration screen opens.
2. To enable the SNMP agent, click the **Enable** box.
3. In the Client Access Allow List, type an IP address or network address from which the SNMP agent can accept requests. Click the add (>>) button to add the address to the Current List. For a network address, type in a netmask. To remove an IP address or network address from the list, click the address, and click the remove (<<) button.
4. In the System Information section, type the following information:
   - In the **System Contact** box, enter the contact name and email address for the person who should be contacted if this BIG/ip Controller generates a trap.
   - In the **Machine Location** box, enter a machine location, such as *First Floor*, or *Building 1*, that describes the physical location of the BIG/ip Controller.
   - In the **Community String** box, enter a community name. The community name is a clear text password used for basic SNMP security and for grouping machines that you manage.
5. In the Trap Configuration section, type the following information:
   - Check **Auth Trap Enabled** to allow traps to be sent for authentication warnings.
   - In the **Community** box, enter the community name to which this BIG/ip controller belongs. Traps sent from this box are sent to the management system managing this community.
   - In the **Port** box, enter the community name to which this BIG/ip controller belongs. Traps sent from this box are sent to the management system managing this community.

- In the **Trap** box, enter the host that should be notified when a trap is sent by the BIG/ip SNMP agent. After you type the **Community**, **Port**, and **Trap** for the trap sink, click the add (>>) button to add it to the Current List.
  To remove a trap sink from the list, click the trap sink you want to remove, and click the remove (<<) button.

6. Click the **Apply** button.

/etc/rc.local

The following entry in the **/etc/rc.local** automatically starts up the SNMP agent when the system boots up (Figure 9.1).

```
# BIG/ip SNMP Agent
if [ -f /etc/snmpd.conf ]; then
   /sbin/bigsnmpd -c /etc/snmpd.conf
fi
```

Figure 9.1 *Starting the SNMP agent in the* ***/etc/rc.local*** *file.*

If the **/etc/snmpd.conf** is present on your system, the SNMP agent is automatically started.

/etc/snmptrap.conf

This configuration file includes OID, trap, and regular expression mappings. The configuration file specifies whether to send a specific trap based on a regular expression. An excerpt of the configuration file is shown in Figure 9.2.

```
# Default traps.
.1.3.6.1.4.1.3375.1.1.110.2.6 (ROOT LOGIN) ROOT LOGIN
.1.3.6.1.4.1.3375.1.1.110.2.5 (denial) REQUEST DENIAL
.1.3.6.1.4.1.3375.1.1.110.2.4 (BIG/ip Reset) SYSTEM RESET
.1.3.6.1.4.1.3375.1.1.110.2.3 (Service detected UP) SERVICE UP
.1.3.6.1.4.1.3375.1.1.110.2.2 (Service detected DOWN) SERVICE DOWN
#.1.3.6.1.4.1.3375.1.1.110.2.1 (error) Unknown Error
#.1.3.6.1.4.1.3375.1.1.110.2.1 (failure) Unknown Failure
```

Figure 9.2 *Excerpt from the* ***/etc/snmptrap.conf*** *file*

Some of the OIDs have been permanently mapped to BIG/ip specific events. The OIDs that are permanently mapped for the BIG/ip Controller include:

- Root login
- Request denial
- System reset
- Service up
- Service down

You may, however, insert your own regular expressions and map them to the 110.1 OID. The **/etc/snmptrap.conf** file contains two examples for mapping your own OIDs:

- Unknown error
- Unknown failure

By default, the lines for these files are commented out. Use these OIDs for miscellaneous events. When lines match your expression, they are sent to your management software with the 110.2.1 OID.

Syslog

In order to generate traps, you must configure **syslog** to send syslog lines to **checktrap.pl**. If the syslog lines make a match to the specified configuration in the **snmptrap.conf** file, a valid SNMP trap is generated. The following lines in the **/etc/syslog.conf** file require the **syslog** look at information logged, scan the **snmptrap.conf** file, and determine if a trap should be generated:

```
local0.* | exec /sbin/checktrap.pl.
local1.* | exec /sbin/checktrap.pl.
auth.* | exec /sbin/checktrap.pl.
```

◆ Note

*If you uncomment these lines, make sure you restart **syslogd**. For more information about working with the **Syslog** utility, see the **BIG/ip Controller Administrator Guide**, Monitoring and Administration.*

Configuring options for the checktrap script

The **checktrap.pl** script reads a set of lines piped in from **syslog**. The script checks each line against a set of regular expressions in the specified configuration file. If a line matches the regular expression, an SNMP trap is sent.

Options for checktrap

snmpd_conf_file=<snmp configuration file>

This is the file that contains the SNMP variables. The **checktrap.pl** gets trap configuration information from this file. The default is **/etc/snmpd.conf**.

trapd_conf_file=<snmp trap configuration file>

This is the file that contains the regular expression to SNMP trap OID mappings. It also contains a description string that is added to the trap message. The default is **/etc/snmptrap.conf**.

trap_program=<snmp trap program>

This is the program that sends the trap. This program should be the **snmptrap** program included with the BIG/ip Controller. The default is **/sbin/snmptrap**.

`no_date_strip`

This turns off automatic date stripping. Normally, each input line is expected to begin with a date. Typically, this date is stripped off before the trap is sent. This option keeps the date information in the trap. By default, the date is stripped from the trap.

`usage`

Prints a usage string.

Glossary

active unit

In a redundant system, the controller which currently load balances connections. If the active unit in the redundant system fails, the standby unit assumes control and begins to load balance connections.

BIG/pipe

A utility that provides command line access to the BIG/ip Controller.

BIG/stat

A statistical monitoring utility that ships on the BIG/ip Controller. This utility provides a snap-shot of statistical information.

BIG/top

A statistical monitoring utility that ships on the BIG/ip Controller. This utility provides real-time information.

big3d

A monitoring utility that collects metrics information about paths between a BIG/ip Controller and a specific local DNS server. The big3d utility runs on BIG/ip Controllers and it forwards metrics information to a 3DNS Controller.

BIND (Berkley Internet Name Domain)

The most common implementation of DNS, which provides a system for matching domain names to IP addresses.

chain

A series of filtering criteria used to restrict access to an IP address. The order of the criteria in the chain determines how the filter is applied, from the general criteria first, to the more detailed criteria at the end of the chain.

cookie persistence

Cookie persistence is a mode of persistence you can configure on the BIG/ip Controller where the controller stores persistent connection information in a cookie.

default wildcard virtual server

A virtual server that has an IP address and port number of **0.0.0.0:0**. This virtual server accepts all traffic which does not match any other virtual server defined in the configuration.

dynamic load balancing modes

Dynamic load balancing modes base connection distribution on live data, such as current server performance and current connection load.

dynamic site content

A type of site content that is automatically generated each time a user accesses the site. Examples are current stock quotes or weather satellite images.

EAV service check

A service check feature that uses an external program to determine if a node is **up** or **down** based on whether the node returns specific content. EAV service check is only one of the three types of service checks available on a BIG/ip Controller. See also service check, and external service checker program.

ECV service check

A service check feature that allows you to determine if a node is up or down based on whether the node returns specific content. ECV service check is only one of the three types of service checks available on a BIG/ip Controller. See also service check.

Extended Application Verification (EAV)

A service check feature that uses an external program to determine if a node is up or down based on whether the node returns specific content.

Extended Content Verification (ECV)

A service check feature that allows you to determine if a node is up or down based on whether the node returns specific content.

external interface

A network interface on the BIG/ip Controller configured to process destination requests. In a basic configuration, this interface has the administration ports locked down. In a normal configuration, this is typically a network interface on which external clients request connections to internal servers.

external service checker program

A custom program that performs a service check on behalf of the BIG/ip Controller. See also, EAV service check.

F-Secure SSH

An encryption utility that allows secure shell connections to a remote system.

BIG/ip web server

The web server that runs on a BIG/ip Controller and hosts the F5 Configuration utility.

fail-over

The process whereby a standby unit in a redundant system takes over when a software failure or a hardware failure is detected on the active unit.

fail-over cable

The cable that directly connects the two controller units together in a redundant system.

Fastest mode

A dynamic load balancing mode that bases connection distribution on which server currently exhibits the fastest response time to node pings.

FDDI (Fiber Distributed Data Interface)

A multi-mode protocol for transmitting data on optical-fiber cables up to 100 Mbps.

First-Time Boot utility

A utility that walks you through the initial system configuration process. The First-Time Boot utility runs automatically when you turn on a controller for the first time.

host

A network server which manages one or more virtual servers that the 3DNS Controller uses for load balancing.

ICMP (Internet Control Message Protocol)

An Internet communications protocol used to determine information about routes to destination addresses, such as virtual servers managed by BIG/ip Controllers and 3DNS Controllers.

internal interface

A network interface on the BIG/ip Controller configured to process source requests. In a basic configuration, this interface has the administration ports open. In a normal configuration, this is typically a network interface which handles connections from internal servers.

iQuery

A UDP based protocol used to exchange information between BIG/ip Controllers and 3DNS Controllers. The iQuery protocol is officially registered for port 4353.

last hop

A last hop is the last hop a connection took to get to the BIG/ip Controller. You can configure the BIG/ip Controller to send packets back to the device from which they originated when that device is part of a last hop pool.

Least Connections mode

A dynamic load balancing mode that bases connection distribution on which server currently manages the fewest open connections.

load balancing mode

A particular method of determining how to distribute connections across an array.

loopback adapter

A loopback adapter is a software interface that is not associated with an actual network card. The nPath routing configuration requires you to configure loopback adapters on servers.

MAC (Media Access Control)

A protocol that defines the way workstations gain access to transmission media, most widely used in reference to LANs. For IEEE LANs, the MAC layer is the lower sublayer of the data link layer protocol.

MAC Address

An address used to represent hardware devices on an Ethernet network.

member

A reference to a node when it is included in a particular virtual server mapping. Virtual server mappings typically include multiple member nodes.

mirroring

A feature on the BIG/ip Controller that preserves connection and persistence information in a BIG/ip Controller redundant system.

named

The name server daemon, which manages domain name server software.

NAT (Network Address Translation)

An alias IP address that identifies a specific node managed by the BIG/ip Controller to the external network.

node

A specific combination of an IP address and port number associated with a server in the array managed by the BIG/ip Controller.

node address

The IP address associated with one or more nodes. This IP address can be the real IP address of a network server, or it can be an alias IP address on a network server.

node alias

A node address that the BIG/ip Controller uses to verify the status of multiple nodes. When the BIG/ip Controller uses a node alias to check node status, it pings the node alias. If the BIG/ip Controller receives a response to the ping, it marks all nodes associated with the node alias as up, and if it does not receive a response to the ping, the BIG/ip Controller marks all nodes associated with the node alias as **down**.

node ping

A feature that the BIG/ip Controller uses to determine whether nodes are **up** or **down**. Node ping sends standard echo pings to servers and transparent devices. If the server or device responds to the ping, it marks the related nodes **up**. If the server or device does not respond to the ping, it marks the related nodes **down**.

node port

The port number or service name hosted by a specific node.

node status

Whether a node is up and available to receive connections, or **down** and unavailable. The BIG/ip Controller uses the node ping and service check features to determine node status.

Observed mode

A dynamic load balancing mode that bases connection distribution on a combination of two factors: the server that currently hosts the fewest connections, and also has the fastest response time.

persistence

A series of related connections received from the same client, having the same session ID. When persistence is turned on, a controller sends all connections having the same session ID to the same node instead of load balancing the connections.

port

A number that is associated with a specific service supported by a host. Refer to the Services and Port Index for a list of port numbers and corresponding services.

port-specific wildcard virtual server

A wildcard virtual server address that uses a port number other than **0**.

Predictive mode

A dynamic load balancing mode that bases connection distribution on a combination of two factors: the server that currently hosts the fewest connections, but also has the fastest response time. Predictive mode also ranks server performance over time, and passes connections to servers which exhibit an improvement in performance rather than a decline.

Priority mode

A static load balancing mode that bases connection distribution on server priority levels. The BIG/ip Controller distributes connections in a round robin fashion to all nodes in the highest priority group. If all the nodes in the highest priority group become unavailable, the BIG/ip Controller begins to pass connections to nodes in the next lower priority group.

rate class

A rate class determines the volume of traffic allowed through a rate filter.

ratio

A parameter that assigns a weight to a virtual server for load balancing purposes.

Ratio mode

The Ratio load balancing mode distributes connections across an array of virtual servers in proportion to the ratio weights assigned to each individual virtual server.

receive expression

A receive expression is the text string that the BIG/ip Controller looks for in the web page returned by a web server during an extended content verification (ECV) service check.

redundant system

A pair of controllers that are configured for fail-over. In a redundant system, there are two controller units, one running as the active unit and one running as the standby unit. If the active unit fails, the standby unit takes over and manages connection requests.

remote administrative IP address

An IP address from which a controller allows shell connections, such as Telnet or SSH.

Round Robin mode

A static load balancing mode that bases connection distribution on a set server order. Round Robin mode sends a connection request to the next available server in the order.

send string

A send string is the request that the BIG/ip Controller sends to the web server during an extended content verification (ECV) service check.

service check

A BIG/ip Controller feature that determines whether a node is up or down. When a BIG/ip Controller issues a service check, it attempts to connect to the service hosted by the node. If the connection is successful, the node is up. If the connection fails, the node is down. See also ECV service check, and EAV service check.

SNAT (Secure Network Address Translation)

A SNAT is a feature you can configure on the BIG/ip Controller. A SNAT defines a routable alias IP address that one or more nodes can use as a source IP address when making connections to hosts on the external network.

SNMP (Simple Network Management Protocol)

The Internet standard protocol, defined in STD 15, RFC 1157, developed to manage nodes on an IP network.

sod (switch over daemon)

A daemon that controls the fail-over process in a redundant system.

standby unit

A controller in a redundant system that is always prepared to become the active unit if the active unit fails.

stateful site content

Content that maintains dynamic information for clients on an individual basis and is commonly found on e-commerce sites. For example, a site that allows a user to fill a shopping cart, leave the site, and then return and purchase the items in the shopping cart at a later time has stateful site content which retains the information for that client's particular shopping cart.

static load balancing modes

Static load balancing modes base connection distribution on a pre-defined list of criteria; it does not take current server performance or current connection load into account.

static site content

A type of site content that is stored in HTML pages, and changes only when an administrator edits the HTML document itself.

sticky mask

A sticky mask is a special IP mask that you can configure on the BIG/ip Controller. This mask optimizes sticky persistence entries by grouping more of them together.

transparent node

A node that appears as a router to other network devices, including the BIG/ip Controller.

virtual address

An IP address associated with one or more virtual servers managed by the BIG/ip Controller.

virtual port

The port number or service name associated with one or more virtual servers managed by the BIG/ip Controller. A virtual port number should be the same TCP or UDP port number to which client programs expect to connect.

virtual server

A specific combination of a virtual address and virtual port, associated with a content site that is managed by a BIG/ip Controller or other type of host server.

virtual server mapping

The group of nodes across which a virtual server load balances connections for a given site.

watchdog timer card

A hardware device that monitors the BIG/ip Controller for hardware failure.

wildcard virtual server

A virtual server that uses an IP address of **0.0.0.0**. A wildcard virtual server accepts connection requests for destinations outside of the local network. Wildcard virtual servers are included only in Transparent Node Mode configurations.

Index

/etc/bigd.conf file 4-2, 4-8
/etc/bigip.conf file 3-4
/etc/hosts.allow file 9-3
/etc/snmpd.conf file 9-4
/etc/snmptrap.conf file 9-8
/etc/syslog.conf file 8-17

802.1q VLAN trunk mode. See VLAN trunk mode

A

active-active mode
- additional BIG/db configuration parameters 6-18
- configuring 6-10
- configuring servers 6-12
- configuring shared IP alias 6-10
- disabling automatic fail back 6-17
- enabling BIG/db configuration parameters 6-13
- placing a controller back in service 6-17
- returning to active/standby mode 6-21
- running mixed versions of BIG/ip 6-20
- synchronizing the configuration 6-15
- system fail-over 6-16
- taking a controller out of service 6-17
- transitioning from active/standby 6-15
- updating the fail-over daemon 6-15
- virtual address configuration 6-13

advanced service checks. See service checks

B

BIG/db
- bigdba 8-25
- bigdba commands 8-25

BIG/db configuration keys 8-26
BIG/db database 7-34
BIG/ip Controllers
- domestic 1-8
- international 1-8
- setting preferred active unit 6-8
- SNMP 9-1–9-10

BIG/ip HA Controller 1-8
BIG/ip HA+ Controller 1-8
BIG/ip LB Controller 1-8
BIG/pipe
- monitoring 8-1

BIG/pipe utility 1-5
BIG/stat 8-14
BIG/stat command line options 8-14
BIG/stat utility 8-1
BIG/top 8-1
- runtime commands 8-16

BIG/top command line options 8-16
BIG/top utility 1-5
bigdnode daemon 4-5
bigpipe commands
- active-active 6-20

bigpipe conn command 8-22
bigpipe global command 8-9
bigpipe interface command 7-35, 8-10
bigpipe nat command 8-8
bigpipe node command 8-7
bigpipe port command 8-8
bigpipe snat command 8-9
bigpipe summary command 8-2
bigpipe vip command 8-6

C

checktrap script
- configuring options for 9-9, 9-10

client IP address 3-9
configurations
- optimization 2-18
- scalability 1-4

connection table 8-22
connections
- mirroring. See mirroring 6-2

constant operands 3-10
controller
- active 6-1
- standby 6-1

controllers
BIG/ip HA 1-8
BIG/ip HA+ 1-8
LB 1-8
cookie persistence 1-7

D

default.txt file 8-26
destination address affinity 5-5–5-7
destination processing 2-26

E

EAV
pingers 4-9
setting up 4-5–4-15
ECV
expressions 3-10
Extended Application Verification. See EAV 4-1
Extended Content Verification. See ECV 4-1
external service checker 4-4
installing 4-7
programs 4-4, 4-5
requirements 4-6

F

F5 Configuration utility
configuring a pool 3-6
configuring a virtual server with a pool 3-14
configuring gateway fail-safe 6-4
configuring network fail-over 6-7
enhancements 1-14
FTP EAV service check 4-10
POP3 EAV service check 4-11
resetting statistics 8-9
setting SNMP properties 9-6
setting up ECV 4-3
using VLAN tags 7-36
Fastest mode 2-2
filters 2-7
IP filters 2-7
Rate filters and rate classes 2-9
firewall load balancing 7-12–7-16
First-Time Boot utility 1-4
forwarding virtual servers 2-31
F-Secure SSH client 1-8

G

gateway fail-safe 6-4, 6-6
messages 6-6
global mirroring 6-2

H

hard-wired fail-over 6-7
Hash mode 5-2

I

ifconfig command 7-35
illegal connection attempts 8-21
Insert mode 5-3
intelligent traffic control (ITC) 1-10, 3-1
Interface cards 2-11
configuring additional interfaces 2-11
routing with multiple NICs 2-16
interface cards
status 8-10
interface security 2-28
IP addresses 1-7
client 5-7
IP filters 1-6
destination IP addresses 2-8
illegal connection attempts 8-21
in the F5 Configuration utility 2-8
source IP addresses 2-8
IP network topology
single interface 7-28
two interfaces 7-30
ISP load balancing 7-9–7-12
ITC. See intelligent traffic control

L

last hop pool 7-8

last hop pools 2-30
Least Connections mode 2-2
least connections mode 3-4
least_conn_member. See load balancing methods 3-4
load balancing
    dynamic modes 1-6
    Fastest mode 1-6, 2-2
    Least Connections mode 1-6, 2-2
    members 3-1
    Observed mode 1-6, 2-2
    pools 3-1
    Predictive mode 1-6, 2-3
    Priority mode 1-6, 2-3
    priority number 2-3
    Ratio mode 1-5
    Round Robin mode 1-5
    static modes 1-5
load balancing methods 3-3
    least_conn_member 3-4
    priority_member 3-3
    ratio_member 3-3
load balancing pools 3-7
    described 3-1
log files 8-22
    viewing 8-22
log messages 8-17

M

member list 3-3
members
    described 3-3
mirroring 6-1–6-4
    commands 6-2, 6-3
    connections 6-2
    persistence 6-2
    SNAT connections 6-3
monitoring
    NATs 8-13
    services 8-10
    SNATs 8-13
    virtual addresses 8-10
    virtual servers 8-10

N

NATs
    statistics 8-21
network address translations. See NATs
network configurations
    802.1q VLAN trunk mode 7-33
    firewall load balancing 7-12–7-16
    IP network topology 7-28, 7-30
    ISP load balancing 7-9–7-12
    nPath routing 7-1–7-7
    per-connection routing 7-7
    SNAT and virtual servers 7-25
    per-connection routing 7-7–7-9
    VPN and router load
        balancing 7-20–7-25
    VPN load balancing 7-17–7-20
network interface cards. See interface cards
network-based fail-over 6-7
node addresses 8-2
    monitoring 8-12
    node aliases 2-22
    node ping 2-21
    removing from service 8-20
    statistics 8-21
node aliases 2-22
node lists
    dependent 3-2
node ping 2-21
node ping log file 8-22
node pings 4-5
    reducing node pings 2-21
nodes 8-2
    monitoring 8-12
    service check 2-21
    statistics 8-21
    viewing on command line 8-21
nPath routing 7-1–7-7
    defining virtual servers 7-2–7-4
    setting idle connection timeout 7-5–7-7
    setting routes 7-4

O

Observed mode 2-2

operands 3-10
operators 3-10, 3-11

P

Passive mode 5-4
passwords
    BIG/ip Controller 8-23
    BIG/ip web server 8-23
per-connection routing 2-29
persistence 1-7, 5-8–5-10
    advanced options 5-1–5-10
    cookie 5-1
    HTTP cookie 5-1–5-5
    maintaining across virtual servers 5-8–5-10
    mirroring.See mirroring 6-2
    sticky. See destination address affinity
    per-connection routing 7-7–7-9
    using persist mask 5-7
pools 3-1, 3-14
    defining 3-4–3-7, 3-14
    elements 3-6
    example entry 3-5
    independent 3-2
ports 8-8
Predictive mode 2-3
priority levels 2-5
Priority mode 2-3
priority mode 3-3
priority_member. See load balancing methods 3-3

R

rate classes 1-7, 2-9
    F5 Configuration utility 2-9
rate filters
    F5 Configuration utility 2-10
ratio weights 2-5, 3-3
ratio_member. See load balancing methods 3-3
redundant systems 6-1
    active/standby 6-10
    active-active controllers 6-9–6-21
    active-active mode 1-10, 6-10
    default mode 6-10
    gateway fail-safe 6-4–6-6
    mirroring 6-1
    network-based failover 6-7–6-8
removing from service
    nodes 8-20
    ports 8-19
    virtual address 8-19
    virtual servers 8-19
reset connections on service down 2-34
Rewrite mode 5-3
router configurations 2-20
routing
    per-connection 7-7
rules
    configuring 3-13, 3-13–3-15
    elements 3-15
    selecting pools 3-7–3-12

S

secure network address translations. See SNATs
security
    changing passwords 8-23
    illegal connection attempts 8-21
service check 2-21
service checks 4-1–4-15
    EAV 4-1
    ECV 4-1
SNATs 7-25
SNAT connections 6-3
SNMP
    /etc/hosts.allow file 9-3
    /etc/hosts.deny file 9-3
    client access 9-4
    configuration file requirements 9-3–9-9
    configuring 9-1–9-10
    downloading MIBs 9-2
    F5 Configuration utility 9-4
    MIB 1-15
    OIDs 9-8
    syslog 9-9

trap configuration 9-5
source translation 2-26
special features
introduction 2-1
SQL-based services 4-13–4-15
statistics
NATs 8-21
node addresses 8-21
nodes 8-21
resetting 8-6
resetting for NATs 8-8
resetting for virtual ports 8-8
resetting global 8-9
resetting node addresses 8-7
resetting node servers 8-7
resetting SNATs 8-9
resetting virtual addresses 8-6
resetting virtual servers 8-6
viewing system 8-21
virtual addresses 8-21
virtual ports 8-21
virtual servers 8-21
string variables
http_cookie 3-12
http_host 3-12
http_method 3-12
http_uri 3-12
http_version 3-12
syslog 8-1, 8-16, 9-9

T

transparent virtual servers 2-32

U

users
creating new 8-23
utilities
BIG/pipe 1-5, 8-2
BIG/stat 8-14
BIG/top 1-5, 8-15
First-Time Boot 1-4
syslog 8-16

V

variable operands 3-10, 3-11
versatile interfaces
destination route and translation processing 2-26
source and destination processing 2-24
source translation processing 2-26
virtual addresses 8-2, 8-11
statistics 8-21
virtual ports 8-2, 8-11
allowing 8-20
statistics 8-21
virtual server mappings 5-8, 5-10, 8-21
virtual server port translation 2-33
virtual servers 3-2, 5-8–5-10, 7-25, 8-2, 8-10
statistics 8-21
transparent 1-13
viewing on command line 8-20
VLAN trunk mode
adding tag definitions 7-34
configuring multiple VLANS 7-34
disabling tags 7-35
enabling tags 7-35
setting up 7-33–7-36
using F5 configuration utility 7-36
using ifconfig 7-35
using netstat 7-35
VPN and router load balancing 7-20–7-25
VPN load balancing 7-17–7-20