BIG-IP® Cache Controller Administrator Guide

version 3.3

Service and Support Information

Product Version

This manual applies to version 3.3 of the BIG-IP® Cache Controller.

Obtaining Technical Support

Web	tech.f5.com
Phone	(206) 272-6888
Fax	(206) 272-6802
Email (support issues)	support@f5.com
Email (suggestions)	feedback@f5.com

Contacting F5 Networks

Web	www.f5.com
Toll-free phone	(888) 961-7242
Corporate phone	(206) 272-5555
Fax	(206) 272-5556
Email	sales@f5.com
Mailing Address	501 Elliott Avenue West 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.

Trademarks

F5, BIG-IP, and 3-DNS are registered trademarks of F5 Networks, Inc. SEE-IT, GLOBAL-SITE, EDGE-FX, and FireGuard 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® Cache Controller may include cryptographic software. Under the Export Administration Act, the United States government may consider it a criminal offense to export this BIG-IP® Cache 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.

FCC Compliance

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

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

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

Canadian Regulatory Compliance

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

Standards Compliance

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

Acknowledgments

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

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

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

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

This product includes software developed by Adam Glass.

This product includes software developed by Christian E. Hopps.

This product includes software developed by Dean Huxley.

This product includes software developed by John Kohl.

This product includes software developed by Paul Kranenburg.

This product includes software developed by Terrence R. Lambert.

This product includes software developed by Philip A. Nelson.

This product includes software developed by Herb Peyerl.

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

This product includes software developed by Chris Provenzano.

This product includes software developed by Theo de Raadt.

This product includes software developed by David Muir Sharnoff.

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

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

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

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

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

This product includes software developed by Christos Zoulas.

This product includes software developed by Charles Hannum.

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

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

In the following statement, "This software" refers to the Mitsumi CD-ROM driver: This software was developed by Holger Veit and Brian Moore for use with "386BSD" and similar operating systems. "Similar operating systems" includes mainly non-profit oriented systems for research and education, including but not restricted to "NetBSD," "FreeBSD," "Mach" (by CMU).

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

This product includes software developed by the Apache Group for use in the Apache HTTP server project

(http://www.apache.org/).

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

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

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

F5 Networks Limited Warranty

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

Limited Warranty

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

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

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

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

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

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

day period, Customer is entitled, at no charge, to updated versions of covered software such as bug fixes, and incremental enhancements as designated by minor revision increases. In addition, Customer will receive special pricing on upgraded versions of covered Products such as new clients, new modules, and major enhancements designated by major revision increases. Customer may purchase a Maintenance Agreement for enhanced maintenance and support services.

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

End-user Software License

IMPORTANT - READ BEFORE INSTALLING OR OPERATING THIS PRODUCT

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

1. Scope. This License applies to the software component ("Software") of the F5 product identified above ("Product") and any corrections, updates, new releases and new versions of such software. 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 with an unlimited number of servers. 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. The Software incorporates certain third party software, which is used subject to licenses from the respective owners. The protections given to F5 under this License also apply to the suppliers of this third party software, who are intended third party beneficiaries of this License.

3. Restrictions. The Software, documentation, and the associated copyrights and other intellectual

property rights are owned by F5 or its licensors, and are protected by law and international treaties. 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 US government 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 importing, exporting, or using cryptographic software.

5. Limited 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.

   a. 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 Software that fails during the warranty period at no cost to Licensee. Any Product 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 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.

   b. Restrictions. The foregoing limited warranties extend only to the original Licensee, and do not apply if the Software or the 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.

6. Infringement Indemnity. F5 will, at its expense, defend any suit brought against Licensee based upon a claim that the Software as delivered by F5 directly infringes a valid patent or copyright. F5 will pay costs and damages finally awarded against Licensee directly attributable to any such claim, but only on condition that (a) F5 is notified in writing of such claim within ten days following receipt by Licensee; (b) F5 has sole control of the defense and settlement negotiations, (c) Licensee provides F5 all information and communications received by Licensee concerning such claim, and (d) Licensee provides reasonable assistance to F5 when requested. F5 will have the right, at its option and expense, (i) to obtain for Licensee rights to use the Software, (ii) to replace or modify the Software so it becomes non-infringing , or (iii) to accept return of the Software in exchange or for a credit not to exceed the purchase price paid by Licensee for such Software. The foregoing, subject to the following restrictions, states the exclusive liability of F5 to Licensee concerning infringement.

   a. Restrictions. F5 will have no liability for any claim of infringement based on: (i) use of a superseded or altered release of the Software, (ii) use of the Software in combination with equipment or software

not supplied or specified by F5 in the Software documentation where the Software would not itself be infringing, (iii) use of the Software in an application or environment not described in the Software Documentation or (iv) Software that has been altered or modified in any way by anyone other than F5 or according to F5's instructions.

7. U.S. Government Restricted Rights. The Software was developed at private expense and is provided with "RESTRICTED RIGHTS." Use, duplication or disclosure by the government is subject to restrictions as set forth in FAR 52.227-14 and DFARS 252.227-7013 et. seq. or its successor. The use of this Software by the government constitutes acknowledgment of F5's and its licensors' rights in the Software.

8. DISCLAIMER; LIMITATION OF REMEDY. EXCEPT FOR THE WARRANTIES SPECIFICALLY DESCRIBED HEREIN, F5 AND ITS THIRD PARTY LICENSORS DO NOT MAKE ANY GUARANTEE OR WARRANTY, EXPRESS OR IMPLIED, WITH RESPECT TO THE SOFTWARE, SPECIFICATIONS, SUPPORT, SERVICE OR ANYTHING ELSE. NEITHER F5 NOR ITS THIRD PARTY LICENSORS HAVE AUTHORIZED ANYONE TO MAKE ANY REPRESENTATIONS OR WARRANTIES OTHER THAN AS PROVIDED ABOVE. F5 AND ITS THIRD PARTY LICENSORS DISCLAIM 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. THE COLLECTIVE LIABILITY OF F5 AND ITS THIRD PARTY LICENSORS UNDER THIS LICENSE WILL BE LIMITED TO THE AMOUNT PAID FOR THE PRODUCT. F5 AND ITS THIRD PARTY LICENSORS 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.

9. Termination. The license granted in Section 2 is effective until terminated, and will automatically terminate if Licensee fails to comply with any of its provisions. Upon termination, Licensee will destroy the Software and documentation and all copies or portions thereof.

10. 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

Introduction

Chapter 1

Configuring Local Server Acceleration

Chapter 2

Configuring Remote Server Acceleration

Chapter 3

Configuring Forward Proxy Caching

Chapter 4

Essential Configuration Tasks

Introduction

- Getting started
- Using the Administrator Kit
- What's new in version 3.3
- Learning more about the BIG-IP Controller product family

Getting started

Before you start installing the controller, we recommend that you browse the Administrator Guide and find the load balancing solution that most closely addresses your needs. Briefly review the basic configuration tasks and the few pieces of information you should gather in preparation for completing the tasks, such as IP addresses and host names.

Once you find your solution and gather the necessary network information, turn to the Installation Guide for hardware installation instructions, and then return to the Administrator Guide to follow the steps for setting up your chosen solution.

Choosing a solution

Most cache load balancing configurations are of one of three types, each of which is described below. In this guide, you can find a chapter devoted to each one of the common configurations that includes:

- An overview of the configuration
- A task summary, itemizing the tasks you must follow to implement the configuration
- A diagram depicting a sample implementation of the configuration, with sample IP addresses and device names
- Detailed instructions for the configuration tasks

Local server acceleration

This chapter describes how to use content-aware traffic direction to enhance the efficiency of an array of cache servers that cache content for a local web server.

Remote server acceleration

This chapter describes how to use content-aware traffic direction to enhance the efficiency of an array of cache servers that cache content for a remote web server.

Forward proxy caching

This chapter explains how to use content-aware traffic direction to enhance the efficiency of an array of cache servers storing Internet content for internal users.

Choosing a configuration tool

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

The First-Time Boot utility

All users will use the First-Time Boot utility, a wizard that walks you through the initial system set up. The First-Time Boot utility automatically starts the first time you turn the controller on, and it prompts you to enter basic system information including a root password and the IP addresses that will be assigned to the network interfaces. The Installation Guide provides detailed information about the specific pieces of information that the First-Time Boot utility prompts you to enter.

The Configuration utility

The Configuration utility is a web-based administrative application that you use to configure and monitor the load balancing setup on the BIG-IP Controller. In the Configuration utility, you can view, change, or add any setting supported by the BIG-IP Controller. You can also monitor current system performance, and download administrative tools such as the SNMP MIB or the SSH client. The Configuration utility requires Netscape Navigator version 4.7 or later, or Microsoft Internet Explorer version 4.1 or later.

The bigpipe and bigtop command line utilities

The bigpipe™ utility is the command line counter-part to the Configuration utility. Using **bigpipe** commands, you can configure virtual servers, open ports to network traffic, and configure a wide variety of features. To monitor the BIG-IP Controller, you can use

certain **bigpipe** commands, or you can use the bigtop™ utility, which provides real-time system monitoring. You can use the command line utilities directly on the BIG-IP Controller, or you can execute commands via a remote shell, such as the SSH client (included with the global release only), or a Telnet client (for countries restricted by cryptography export laws). The BIG-IP Controller Reference Guide provides detailed information about command line syntax.

Using the Administrator Kit

The *BIG-IP® Controller Administrator Kit* provides simple steps for quick, basic configuration, and also provides detailed information about more advanced features and tools, such as the **bigpipe** command line utility. The information is organized into the guides described below.

- ***Installation Guide***
  The Installation Guide walks you through the basic steps needed to get the hardware plugged in and the system connected to the network. Most users turn to this guide only the first time that they set up a BIG-IP Controller. The Installation Guide also covers general network administration issues, such as setting up common network administration tools including Sendmail.
- ***Administrator Guide***
  The Administrator Guide provides examples of common load balancing solutions supported by the particular type of BIG-IP Controller you purchased. For example, in the BIG-IP HA Controller Administrator Guide, you can find everything from a basic web server load balancing solution to a firewall load balancing solution.
- ***Reference Guide***
  The Reference Guide provides basic descriptions of individual BIG-IP objects, such as pools, nodes, and virtual servers. It also provides syntax information for **bigpipe** commands, configuration utilities, configuration files, and system utilities.

- ***F-Secure SSH User Guide***
  This guide is distributed only with BIG-IP Controllers that support the F-Secure SSH client (a tool used for remote command line access). It provides information about setting up and using the SSH client.

Stylistic conventions

To help you easily identify and understand certain types of information, all F5 Networks administrative documentation uses the stylistic conventions described below.

WARNING

All examples in F5 Networks documentation use only non-routable IP addresses. When you set up the solutions we describe, you must use IP addresses suitable to your own network in place of our sample addresses.

Identifying new terms

When we first define a new term, the term is shown in bold italic text. For example, a ***virtual server*** is a the combination of an IP address and port that maps to a set of back-end servers.

Identifying references to objects, names, and commands

We apply bold text to a variety of items to help you easily pick them out of a block of text. These items include web addresses, IP addresses, utility names, and portions of commands, such as variables and keywords. For example, the **bigpipe vip** command requires that you include at least one **<node>** variable.

Identifying references to other documents

We use italic text to denote a reference to another document. In references where we provide the name of a book as well as a specific chapter or section in the book, we show the book name in bold, italic text, and the chapter/section name in italic text to help

quickly differentiate the two. For example, you can find information about bigpipe commands in the *bigpipe Command Reference* section of the ***BIG-IP Controller Reference Guide***.

Identifying command syntax

We show actual, complete commands in bold Courier text. Note that we do not include the corresponding screen prompt, unless the command is shown in a figure that depicts an entire command line screen. For example, the following command sets the BIG-IP Controller load balancing mode to Round Robin:

```
bigpipe lb rr
```

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

Item in text	Description
****	Continue to the next line without typing a line break.
< >	You enter text for the enclosed item. For example, if the command has **<your name>**, type in your name.
|	Separates parts of a command.
[ ]	Syntax inside the square brackets is optional.
...	Indicates that you can type a series of items.

Table 1 *Command line syntax conventions*

Finding additional 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 from the web server on the BIG-IP Controller. 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 web server on the BIG-IP Controller has PDF versions of the guides included in the Administrator Kit. BIG-IP Controller upgrades replace these guides with updated versions as appropriate.
- The web-based Configuration utility has online help for each screen. Simply click the **Help** button in the toolbar.
- Individual **bigpipe** 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 web server on the BIG-IP Controller 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, updates for administrator guides (in PDF format), and the Ask F5 natural language question and answer engine. To access this site, you need to obtain a customer ID and a password from the F5 Help Desk.

What's new in version 3.3

The BIG-IP Controller offers the following major new features in version 3.3, in addition to many smaller enhancements.

BIG-IP e-Commerce Controller

The BIG-IP e-Commerce Controller is a new member of the BIG-IP product family. You can use the BIG-IP e-Commerce Controller to process SSL connections to your network. This controller contains a specific set of software and hardware features that accelerate SSL connections.

BIG-IP Cache Controller

This version of the BIG-IP Controller is available as the BIG-IP Cache Controller. The BIG-IP Cache Controller version contains a specific set of features from the BIG-IP Controller that maximizes the efficiency of caches in your network. In addition to the load balancing features available with this controller, this version of the controller has new rule syntax that provides the ability to redirect HTTP requests to caches in your network. These features include:

- **Cacheable content determination**
  This feature enables you to determine the type of content you cache on the basis of any combination of elements in the header of an HTTP request.
- **Content affinity**
  This feature assures that the same cache serves the same content subset even when caches become temporarily unavailable or when caches are added to or deleted from the cache pool.
- **Hot content load balancing**
  When configured, this feature identifies highly requested content and redirects these requests to a hot pool for load balancing.
- **Intelligent cache population**
  When configured, this feature allows caches to retrieve content from other caches in addition to the origin web server.

Performance enhancements

This version of the BIG-IP Controller includes internal performance enhancements. These enhancements improve the overall performance of the BIG-IP Controller.

Learning more about the BIG-IP Controller product family

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

- **The BIG-IP LB Controller**
  The BIG-IP LB Controller provides basic load balancing features.
- **The BIG-IP FireGuard Controller**
  The BIG-IP FireGuard Controller provides load balancing features that maximize the efficiency and performance of a group of firewalls.
- **The BIG-IP Cache Controller**
  The BIG-IP Cache Controller uses content-aware traffic direction to maximize the efficiency and performance of an group of cache servers.
- **The BIG-IP e-Commerce Controller**
  The BIG-IP e-Commerce Controller uses SSL acceleration technology to increase the speed and reliability of the secure connections that drive e-commerce sites.

❖ **The BIG-IP HA Controller**
The BIG-IP HA Controller provides all features from the basic BIG-IP LB Controller to the advanced BIG-IP FireGuard, BIG-IP Cache Controller, and BIG-IP e-Commerce Controller products.

◆ Note

BIG-IP Controllers distributed outside of the United States to a select few countries, 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.

1

Configuring Local Server Acceleration

- Introducing local server acceleration
- Configuration tasks
- Creating pools
- Creating a cache control rule
- Creating a virtual server
- Configuring for intelligent cache population

Introducing local server acceleration

This chapter explains how to set up a ***local server acceleration*** configuration, in which a BIG-IP Cache Controller redundant system uses content-aware traffic direction to enhance the efficiency of an array of cache servers that cache content for a local web server. This type of configuration is useful for any enterprise that wants to improve the speed with which it responds to content requests from users on the Internet.

The configuration detailed in this chapter uses the following BIG-IP Cache Controller features:

- **Cacheable content determination**
  Cacheable content determination enables you to determine the type of content you cache on the basis of any combination of elements in the header of an HTTP request.
- **Content affinity**
  Content affinity ensures that a given subset of content remains associated with a given cache to the maximum extent possible, even when cache servers become unavailable, or are added or removed. This feature also maximizes efficient use of cache memory.
- **Hot content load balancing**
  Hot content load balancing identifies ***hot***, or frequently requested, content on the basis of number of requests in a given time period for a given ***hot content subset***. A hot content subset is different from, and typically smaller than, the content subsets used for content striping. Requests for hot content are redirected to a cache server in the ***hot pool***, a designated group of cache servers. This feature maximizes the use of cache server processing power without significantly affecting the memory efficiency gained by content affinity.
- **Intelligent cache population**
  Intelligent cache population allows caches to retrieve content from other caches in addition to the origin web server. This feature is useful only when working with ***non-transparent*** cache servers, which can receive requests that are destined for the cache servers themselves, as opposed to ***transparent*** cache servers,

which can intercept requests destined for a web server. Intelligent cache population minimizes the load on the origin web server and speeds cache population.

Maximizing memory or processing power

From the time you implement a cache control rule until such time as a hot content subset becomes **hot**, the content is divided across your cache servers, so that no two cache servers contain the same content. In this way, efficient use of the cache servers' memory is maximized.

After a hot content subset becomes **hot**, requests for any content contained in that subset are load balanced, so that, ultimately, each cache server contains a copy of the hot content. The BIG-IP Cache Controller distributes requests for the hot content among the cache servers. In this way, efficient use of the cache servers' processing power is maximized.

Thus, for a particular content item, the BIG-IP Cache Controller maximizes either cache server memory (when the content is **cool**) or cache server processing power (when the content is **hot**), but not both at the same time. The fact that content is requested with greatly varying frequency enables the cache statement rule to evaluate and select the appropriate attribute to maximize for a given content subset.

Using the configuration diagram

Figure 1.1, following, illustrates a local server acceleration configuration, and provides an example configuration for this entire chapter. Remember that this is just a sample: when creating your own configuration, you must use IP addresses, host names, and so on, that are applicable to your own network.

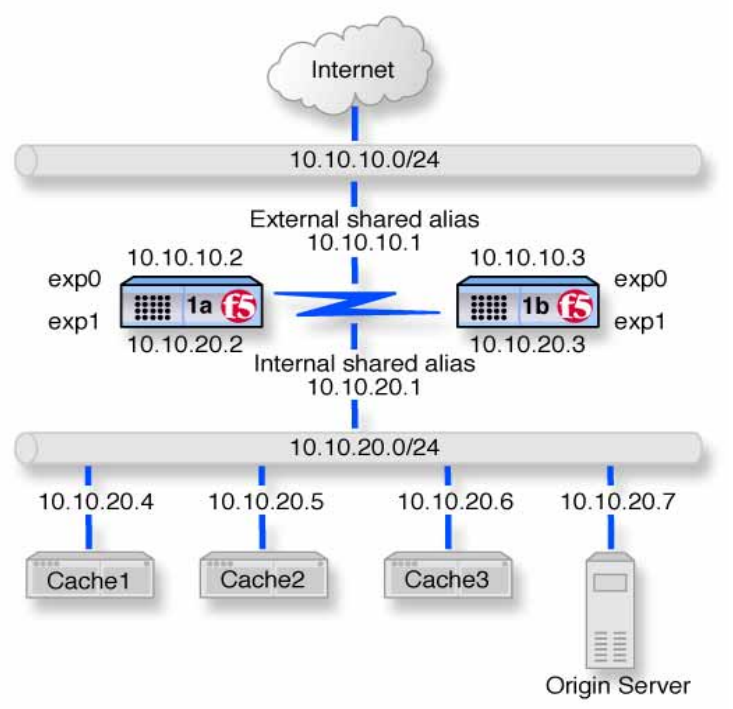

Figure 1.1 *Local server acceleration*

Configuration tasks

If you want to configure local server acceleration, you need to complete the following tasks in order:

- Create pools
- Create a cache control rule
- Create a virtual server
- Configure for intelligent cache population

Each of the following sections explains one of these tasks, and shows how you would perform the tasks in order to implement the configuration shown in Figure 1.1. Note that in this example, as in all examples in this guide, we use only non-routable IP addresses. In a real topology, the appropriate IP addresses would have to be routable on the Internet.

Creating pools

To use the local server acceleration configuration, you need to create three sets of load balancing ***pools***. You create pools for your ***origin server*** (the web server on which all your content resides), for your cache servers, and for your ***hot***, or frequently requested, content servers, which may or may not be cache servers. A pool is a group of devices to which you want the BIG-IP Cache Controller redundant system to direct traffic. For more information about pools, refer to .

You will create these pools:

- **Cache server pool**
  The BIG-IP Cache Controller directs all cacheable requests bound for your web server to this pool, unless a request is for hot content.
- **Origin server pool**
  This pool includes your origin web server. Requests are directed to this pool when:
  - The request is for ***non-cacheable*** content; that is, content that is not identified in the ***cacheable content expression*** part of a cache rule statement. For more information, see *Cacheable content expression*, on page 1-9.
  - The request is from a cache server that does not yet contain the requested content, and no other cache server yet contains the requested content.
  - No cache server in the cache pool is available.

❖ **Hot cache servers pool**
If a request is for frequently requested content, the BIG-IP Cache Controller redundant system directs the request to this pool.

◆ **Note**

While the configuration shown in Figure 1.1 implements a hot cache servers pool, this pool is not required if you want to use the content determination and content affinity features. However, you must implement this pool if you want to use the hot content load balancing or intelligent cache population features.

Creating a pool for the cache servers

First, create a pool for the cache servers. Use either the Configuration utility or the command line to create this pool.

To create a pool using the Configuration utility

1. In the navigation pane, click **Pools**.
   The Pools screen opens.
2. In the toolbar, click the **Add Pool** button.
   The Add Pool screen opens.
3. In the Add Pool screen, configure attributes required for the cache servers you want to add to the pool.
   For additional information about configuring a pool, click the **Help** button.

   Configuration notes
   To create the configuration shown in Figure 1.1:

   - Create a pool named **cache_servers**.
   - Add each cache server from the example, **10.10.20.4**, **10.10.20.5,** and **10.10.20.6**, to the pool.
     For each cache server you add to the pool, specify port **80**, which means this cache server accepts traffic for the HTTP service only.

To create a pool from the command line

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

```
bigpipe pool <pool_name> { lb_method <lb_method> member
   <member_definition> ... member <member_definition> }
```

For example, to implement the configuration shown in Figure 1.1, you use the command:

```
bigpipe pool cache_servers { lb_method round_robin member
   10.10.20.4:80 member 10.10.20.5:80 member 10.10.20.6:80 }
```

Creating a pool for the origin server

Next, create a pool for your origin server. Use either the Configuration utility or the **bigpipe pool** command, as you did to create the pool for the cache servers.

To create a pool using the Configuration utility

1. In the navigation pane, click **Pools**.
   The Pools screen opens.
2. In the toolbar, click the **Add Pool** button.
   The Add Pool screen opens.
3. In the Add Pool screen, configure attributes required for the cache servers you want to add to the pool.
   For additional information about configuring a pool, click the **Help** button.

   Configuration notes
   To create the configuration shown in Figure 1.1:

   - Create a pool named **origin_server**.
   - Add the origin server from the example (**10.10.20.7**) to the pool and specify port **80**, which means the server accepts traffic for the HTTP service only.

To create a pool from the command line

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

```
bigpipe pool <pool_name> { lb_method <lb_method> member
   <member_definition> ... member <member_definition> }
```

For example, to implement the configuration shown in Figure 1.1, you would use the command:

```
bigpipe pool origin_server { lb_method round_robin member
   10.10.20.7:80 }
```

Creating a pool for hot content

The last step in creating pools is to create a pool for hot content. Use either the Configuration utility or the command line to create this pool, as in the previous sections.

To create a pool using the Configuration utility

1. In the navigation pane, click **Pools**.
   The Pools screen opens.
2. In the toolbar, click the **Add Pool** button.
   The Add Pool screen opens.
3. In the Add Pool screen, configure the attributes required for the cache servers you want to add to the pool.
   For additional information about configuring a pool, click the **Help** button.

   Configuration notes
   To create the configuration shown in Figure 1.1:

   - Create a pool named **hot_cache_servers**.
   - Add each cache server from the example, **10.10.20.4**, **10.10.20.5,** and **10.10.20.6**, to the pool. For each cache server you add to the pool, specify port **80**, which means this cache server accepts traffic for the HTTP service only.

To create a pool from the command line

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

```
bigpipe pool <pool_name> { lb_method <lb_method> member
   <member_definition> ... member <member_definition> }
```

To implement the configuration shown in Figure 1.1, you would use the command:

```
bigpipe pool hot_cache_servers { lb_method round_robin member
   10.10.20.4:80 member 10.10.20.5:80 member 10.10.20.6:80 }
```

◆ Note

If you have the hot content pool and the cache servers pool reference the same nodes, it enables use of the intelligent cache population feature.

Creating a cache control rule

A cache control rule is a specific type of rule. A rule establishes criteria by which a BIG-IP Cache Controller directs traffic. A ***cache control rule*** determines where and how the BIG-IP Cache Controller redundant system directs content requests in order to maximize the efficiency of your cache server array and of your origin web server.

A cache control rule includes a ***cache statement***, which is composed of a cacheable content expression and two ***attributes***. An attribute is a variable that the cache statement uses to direct requests. It can also include several optional attributes.

A cache statement may be either the only statement in a rule, or it may be nested in a rule within an **if** statement.

Cacheable content expression

The cacheable content expression determines whether the BIG-IP Cache Controller redundant system directs a given request to the cache server or to the origin server, based on evaluating variables in the HTTP header of the request.

Any content that does not meet the criteria in the cacheable content expression is deemed non-cacheable.

For example, in the configuration illustrated in this chapter, the cacheable content expression includes content having the file extension **.html** or **.gif**. The BIG-IP Cache Controller redundant system considers any request for content having a file extension other than **.html** or **.gif** to be non-cacheable, and sends such requests directly to the origin server.

For your configuration, you may want to cache any content that is not dynamically generated.

Required attributes

The cache control rule must include the following attributes:

- **origin_pool**
  Specifies a pool of servers that contain original copies of all content. Requests are load balanced to this pool when any of the following are true:
  - The requested content does not meet the criteria in the cacheable content condition.
  - No cache server is available.
  - The BIG-IP Cache Controller redundant system is redirecting a request from a cache server that did not have the requested content.
- **cache_pool**
  Specifies a pool of cache servers to which requests are directed in a manner that optimizes cache performance.

Optional attributes

The attributes in this section apply only if you are using the hot content load balancing feature.

◆ Note

In order to use the intelligent cache population feature, the cache_pool and the hot_pool must either be the same pool, or different pools referencing the same nodes.

- **hot_pool**
  Specifies a pool of cache servers to which requests are load balanced when the requested content is **hot**.

The **hot_pool** attribute is required if any of the following attributes is specified:

- **hot_threshold**
  Specifies the minimum number of requests for content in a given hot content set that causes the content set to change from **cool** to **hot** at the end of the period.
  If you specify a value for **hot_pool**, but do not specify a value for this variable, the cache statement uses a default hot threshold of 100 requests.
- **cool_threshold**
  Specifies the maximum number of requests for content in a given hot content set that causes the content set to change from **hot** to **cool** at the end of the hit period.
  If you specify a variable for **hot_pool**, but do not specify a value for this variable, the cache statement uses a default cool threshold of 10 requests.
- **hit_period**
  Specifies the period in seconds over which to count requests for particular content before determining whether to change the content demand status (**hot** or **cool**) of the content.
  If you specify a value for **hot_pool**, but do not specify a value for this variable, the cache statement uses a default hit period of 60 seconds.

❖ **content_hash_size**
Specifies the number of units, or ***hot content subsets***, into which the content is divided when determining whether content demand status is **hot** or **cool**. The requests for all content in a given subset are summed, and a content demand status (**hot** or **cool**) is assigned to each subset. The **content_hash_size** should be within the same order of magnitude as the actual number of requests possible. For example, if the entire site is composed of 500,000 pieces of content, a **content_hash_size** of 100,000 would be typical.
If you specify a value for **hot_pool**, but do not specify a value for this variable, the cache statement uses a default hash size of 1028 subsets.

Content demand status

Content demand status is a measure of the frequency with which a given hot content subset is requested. Content demand status, which is either **hot** or **cool**, is applicable only when using the hot content load balancing feature. For a given hot content subset, content demand status is **cool** from the time the cache control rule is implemented until the number of requests for the subset exceeds the **hot_threshold** during a **hit_period**. At this point content demand status for the subset becomes **hot**, and requests for any item in the subset are load balanced to the **hot_pool**. Content demand status remains **hot** until the number of requests for the subset falls below the **cool_threshold** during a **hit_period**, at which point the content demand status becomes **cool**. The BIG-IP Cache Controller the directs requests for any item in the subset to the appropriate server in the **cache_pool** until such time as the subset becomes **hot** again.

To create a cache statement rule using the Configuration utility

1. In the navigation pane, click **Rules**.
The Rules screen opens.
2. In the toolbar, click the **Add Rule** button.
The Add Rule screen opens.

3. In the Add Rule screen, type the cache statement.
   For example, given the configuration shown in Figure 1.1, to cache all content having either the file extension **.html** or **.gif**, you would type:

```
rule cache_rule { cache ( http_uri ends_with "html" or http_uri
   ends_with "gif" ) { origin_pool origin_server cache_pool
   cache_servers hot_pool hot_cache_servers } }
```

4. Click the **Add** button.

To create a cache statement rule from the command line

To create a cache statement rule from the command line, use the following syntax:

```
bigpipe 'rule <rule_name> { cache  ( <condition>  ) { origin_pool
   <origin_pool_name> cache_pool <cache_pool_name> hot_pool
   <hot_pool_name> hot_threshold <hot_threshold_value>
   cool_threshold <cool_threshold_value> hit_period
   <hit_period_value> content_hash_size <content_hash_size_value>
   } }'
```

For example, given the configuration shown in Figure 1.1, to cache all content having the file extension **.html** or **.gif**, you would use the **bigpipe** command:

```
bigpipe 'rule cache_rule { cache ( http_uri ends_with "html" or
   http_uri ends_with "gif" ) { origin_pool origin_server
   cache_pool cache_servers hot_pool hot_cache_servers } }'
```

Creating a virtual server

Now that you have created pools and a cache control rule to determine how the BIG-IP Cache Controller redundant system will distribute traffic in the configuration, you need to create a virtual server to use this rule and these pools. For this virtual server, use the host name or IP address that Internet clients use to request content from your site.

To create a virtual server using the 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 Add Virtual Server screen, configure the attributes you want to use with the virtual server.
   For additional information about configuring a virtual server, click the **Help** button.

   Configuration notes
   To create the configuration shown in Figure 1.1:

   - Add a virtual server with address **`10.10.10.4`** and port **80** (this means the virtual server accepts traffic for the HTTP service only).
   - Add the rule **cache_rule**.

To create a virtual server from the command line

Use the **bigpipe vip** command to configure the virtual server to use the pool that contains the outside addresses of the firewalls:

```
bigpipe vip <virtual server>:<service> <interface> use rule <rule
   name>
```

In the command, replace the parameters with the appropriate information:

- **<virtual server>** is an IP address appropriate to your network.
- **<service>** is a service you want to configure, such as **HTTP**; **FTP**, or **Telnet**.
- **<interface>** is the interface on the BIG-IP on which you want to create this virtual server.
- **<rule name>** is the name of the rule you want this virtual server to use.

To implement the configuration shown in Figure 1.1, you use the command:

```
bigpipe vip 10.10.10.4:80 use rule cache_rule
```

Configuring for intelligent cache population

Your cache control rule routes a request to the appropriate cache server. However, the cache server will not have the requested content if the content has expired, or if the cache server is receiving a request for this content for the first time. If the cache does not have the requested content, the cache initiates a ***miss request*** (that is, a request resulting from a request for content a cache does not have) for this content. The miss request goes to the origin server specified in the configuration of the cache or to another cache server. If you want to allow intelligent cache population, you should configure the cache with its origin server set to be the virtual server on the BIG-IP Cache Controller, so that the cache sends miss requests to the internal shared interface of the BIG-IP Cache Controller. The BIG-IP Cache Controller translates the destination of the request, and sends the request to either the origin server or another cache server that already has the requested content.

To ensure that the origin server or cache server responds to the BIG-IP Cache Controller rather than to the original cache server that generated the miss request, the BIG-IP Cache Controller also translates the source of the miss request to the translated address and port of the associated Secure Network Address Translation (SNAT) connection.

In order to enable this scenario, you must:

- Create a SNAT on the BIG-IP Cache Controller.
- Enable destination processing on the internal interface of the BIG-IP Cache Controller.

Configuring a SNAT

A Secure Network Address Translation (SNAT) translates the address of a packet from the cache server to the address you specify. For more information about SNATs, see *Configuring SNAT address mappings*, on page 4-27.

To configure a SNAT mapping using the Configuration utility

1. In the navigation pane, click **Secure NATs**.
   The Secure Network Address Translations screen opens.
2. On the toolbar, click **Add SNAT**.
   The Add SNAT screen opens.
3. In the Add SNAT screen, configure the attributes required for the SNAT you want to add.
   For additional information about configuring a pool, click the **Help** button.

   Configuration notes
   To create the configuration shown in Figure 1.1, use the translation address **10.10.10.5**.

To configure a SNAT mapping on the command line

The **bigpipe snat** command defines one SNAT for one or more node addresses.

```
bigpipe snat map <node addr>... <node addr> to <SNAT addr>
```

For example, to implement the configuration shown in Figure 1.1, you use the command:

```
bigpipe snat map default to 10.10.10.5
```

Configuring interfaces

Typically, a BIG-IP Cache Controller has two interfaces:

- An external interface, typically set for ***destination processing***. Destination processing means that the interface can rewrite the destination address of an incoming packet, and allows initiation of virtual server connections.
- An internal interface, typically set for ***source processing***. Source processing means that the interface can rewrite the source of an incoming packet, and allows initiation of SNAT connections.

In this configuration, you must add destination processing to the internal interface. Adding destination processing to the internal interface enables the BIG-IP Cache Controller to direct a request from a cache server to either another cache server or to the origin web server.

To add destination processing to the internal interface using the 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. In the Network Interface Card Properties screen, configure the attributes required for the interface.
   For additional information about creating a pool, click the **Help** button.

   Configuration notes
   To create the configuration shown in Figure 1.1, make sure the **Enable Destination Processing** check box is checked for **exp1**.

To add destination processing to the internal interface from the command line

Use the **bigpipe interface** command with the **source** keyword to turn source processing on or off for an interface:

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

To implement the configuration shown in Figure 1.1 from the command line, you use the command:

```
bigpipe interface exp1 dest enable
```

2

Configuring Remote Server Acceleration

- Introducing remote server acceleration
- Configuration tasks
- Creating pools
- Creating a cache control rule
- Creating a virtual server
- Configuring for intelligent cache population

Introducing remote server acceleration

This chapter explains how to set up a ***remote server acceleration*** configuration, in which a BIG-IP Cache Controller redundant system uses content-aware traffic direction to enhance the efficiency of an array of cache servers that cache content for a remote web server. This configuration is similar to the configuration discussed in Chapter 1, *Configuring Local Server Acceleration*. The difference is that, in this configuration, the cache servers reside on an intranet network, while the origin web server resides on the Internet; in the local server acceleration configuration, the origin web server and the cache servers both reside on the intranet. The remote server alteration configuration is appropriate for any enterprise in which the cache server network and web server network are separated, and you want maximum speed and efficiency from both the cache servers and web server.

The configuration detailed in this chapter uses the following BIG-IP Cache Controller features:

- **Cacheable content determination**
  Cacheable content determination enables you to determine the type of content you cache on the basis of any combination of elements in the header of an HTTP request.
- **Content affinity**
  Content affinity ensures that a given subset of content remains associated with a given cache to the maximum extent possible, even when cache servers become unavailable, or are added or removed. This feature also maximizes efficient use of cache memory.
- **Hot content load balancing**
  Hot content load balancing identifies ***hot***, or frequently requested, content on the basis of number of requests in a given time period for a given ***hot content subset***. A hot content subset is different from, and typically smaller than, the content subsets used for content striping. Requests for hot content are redirected to a cache server in the ***hot pool***, a designated group of cache servers. This feature maximizes the use of cache server processing power without significantly affecting the memory efficiency gained by content affinity.

- **Intelligent cache population**
  Intelligent cache population allows caches to retrieve content from other caches in addition to the origin web server. This feature is useful only when working with ***non-transparent*** cache servers, which can receive requests that are destined for the cache servers themselves, as opposed to ***transparent*** cache servers, which can intercept requests destined for a web server. Intelligent cache population minimizes the load on the origin web server and speeds cache population.

Maximizing memory or processing power

From the time you implement a cache control rule until such time as a hot content subset becomes **hot**, the content is divided across your cache servers, so that no two cache servers contain the same content. In this way, efficient use of the cache servers' memory is maximized.

After a hot content subset becomes **hot**, requests for any content contained in that subset are load balanced, so that, ultimately, each cache server contains a copy of the hot content. The BIG-IP Cache Controller distributes requests for the hot content among the cache servers. In this way, efficient use of the cache servers' processing power is maximized.

Thus, for a particular content item, the BIG-IP Cache Controller maximizes either cache server memory (when the content is **cool**) or cache server processing power (when the content is **hot**), but not both at the same time. The fact that content is requested with greatly varying frequency enables the cache statement rule to evaluate and select the appropriate attribute to maximize for a given content subset.

Using the configuration diagram

Figure 2.1, following, illustrates a remote server acceleration configuration, and provides an example configuration for this entire chapter. Remember that this is just a sample: when creating your own configuration, you must use IP addresses, host names, and so on, that are applicable to your own network.

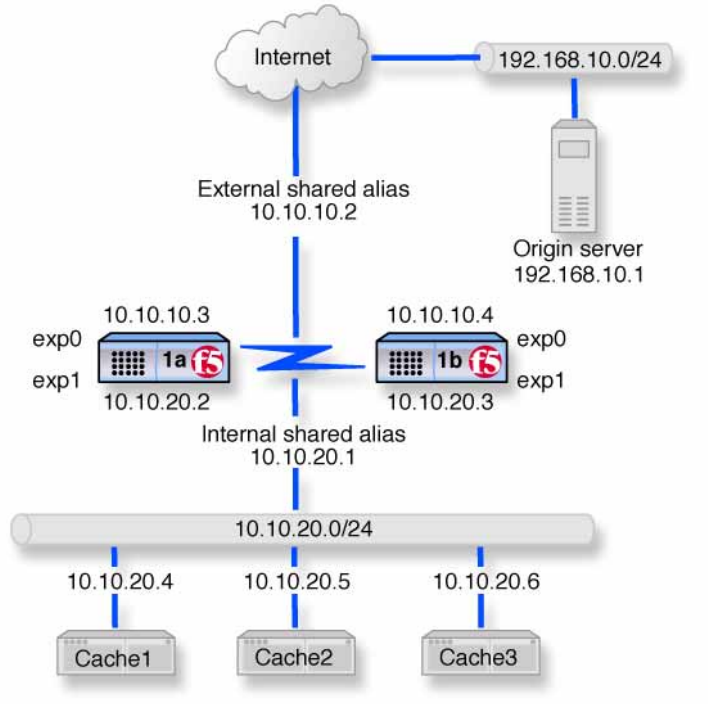

Figure 2.1 *Remote server acceleration*

Configuration tasks

To configure remote server acceleration, complete the following tasks in order:

- Create pools
- Create a cache control rule
- Create a virtual server
- Configure for intelligent cache population

Each of the following sections explains one of these tasks, and shows how you would perform the tasks in order to implement the configuration shown in Figure 2.1. Note that in this example, as in all examples in this guide, we use only non-routable IP addresses. In a real topology, the appropriate IP addresses would have to be routable on the Internet.

Creating pools

To use the remote server acceleration configuration, you create three sets of load balancing ***pools***. You create pools for your ***origin server*** (the web server on which all your content resides), for your cache servers, and for your ***hot***, or frequently requested, content servers, which may or may not be cache servers. A pool is a group of devices to which you want the BIG-IP Cache Controller redundant system to direct traffic. For more information about pools, refer to *Configuring a pool*, on page 4-5.

You will create these pools:

- **Cache server pool**
  The BIG-IP Cache Controller directs all cacheable requests bound for your web server to this pool, unless a request is for hot content.
- **Origin server pool**
  This pool includes your origin web server. Requests are directed to this pool when:

- The request is for ***non-cacheable*** content; that is, content that is not identified in the ***cacheable content expression*** part of a cache rule statement. For more information, see *Cacheable content expression*, on page 2-9.
- The request is from a cache server that does not yet contain the requested content, and no other cache server yet contains the requested content.
- No cache server in the cache pool is available.

❖ **Hot cache servers pool**
If a request is for frequently requested content, the BIG-IP Cache Controller redundant system directs the request to this pool.

◆ Note

While the configuration shown in Figure 2.1 implements a hot cache servers pool, this pool is not required if you want to use the content determination and content affinity features. However, you must implement this pool if you want to use the hot content load balancing or intelligent cache population features.

Creating a pool for the cache servers

First, create a pool for the cache servers. Use either the Configuration utility or the command line to create this pool.

To create a pool using the Configuration utility

1. In the navigation pane, click **Pools**.
   The Pools screen opens.
2. In the toolbar, click the **Add Pool** button.
   The Add Pool screen opens.
3. In the Add Pool screen, configure attributes required for the cache servers you want to add to the pool.
   For additional information about configuring a pool, click the **Help** button.

   Configuration notes
   To create the configuration shown in Figure 2.1:

- Create a pool named **cache_servers**.
- Add each cache server from the example, **10.10.20.4**, **10.10.20.5,** and **10.10.20.6**, to the pool. For each cache server you add to the pool, specify port **80**, which means this cache server accepts traffic for the HTTP service only.

To create a pool from the command line

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

```
bigpipe pool <pool_name> { lb_method <lb_method> member
   <member_definition> ... member <member_definition> }
```

To implement the configuration shown in Figure 2.1, you would use the command:

```
bigpipe pool cache_servers { lb_method round_robin member
   10.10.20.4:80 member 10.10.20.5:80 member 10.10.20.6:80 }
```

Creating a pool for the origin server

Next, create a pool for your origin server. Use either the Configuration utility or the **bigpipe pool** command, as you did to create the pool for the cache servers.

To create a pool using the Configuration utility

1. In the navigation pane, click **Pools**.
   The Pools screen opens.
2. In the toolbar, click the **Add Pool** button.
   The Add Pool screen opens.
3. In the Add Pool screen, configure attributes required for the cache servers you want to add to the pool.
   For additional information about configuring a pool, click the **Help** button.

   Configuration notes
   To create the configuration shown in Figure 2.1:

   - Create a pool named **origin_server**.

- Add the origin server from the example (**192.168.10.1**) to the pool and specify port **80**, which means the server accepts traffic for the HTTP service only.

To create a pool from the command line

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

```
bigpipe pool <pool_name> { lb_method <lb_method> member
   <member_definition> ... member <member_definition> }
```

For example, to implement the configuration shown in Figure 2.1, you would use the command:

```
bigpipe pool origin_server { lb_method round_robin member
   192.168.10.1:80 }
```

Creating a pool for hot content

Finally, create a pool for hot content. You can use either the Configuration utility or the command line to create this pool, as in the previous sections.

To create a pool using the Configuration utility

1. In the navigation pane, click **Pools**.
   The Pools screen opens.
2. In the toolbar, click the **Add Pool** button.
   The Add Pool screen opens.
3. In the Add Pool screen, configure the attributes required for the cache servers you want to add to the pool.
   For additional information about configuring a pool, click the **Help** button.

   Configuration notes
   To create the configuration shown in Figure 2.1:

   - Create a pool named **hot_cache_servers**.

- Add each cache server from the example, **10.10.20.4**, **10.10.20.5,** and **10.10.20.6**, to the pool. For each cache server you add to the pool, specify port **80**, which means this cache server accepts traffic for the HTTP service only.

To create a pool from the command line

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

```
bigpipe pool <pool_name> { lb_method <lb_method> member
   <member_definition> ... member <member_definition> }
```

To implement the configuration shown in Figure 2.1, you use the command:

```
bigpipe pool hot_cache_servers { lb_method round_robin member
   10.10.20.4:80 member 10.10.20.5:80 member 10.10.20.6:80 }
```

Note

If you have the hot content pool and the cache servers pool reference the same nodes, it enables use of the intelligent cache population feature.

Creating a cache control rule

A cache control rule is a specific type of rule. A rule establishes criteria by which a BIG-IP Cache Controller directs traffic. A ***cache control rule*** determines where and how the BIG-IP Cache Controller redundant system directs content requests in order to maximize the efficiency of your cache server array and of your origin web server.

A cache control rule includes a ***cache statement***, which is composed of a cacheable content expression and two ***attributes***. An attribute is a variable that the cache statement uses to direct requests. It can also include several optional attributes.

A cache statement may be either the only statement in a rule, or it may be nested in a rule within an **if** statement.

Cacheable content expression

The cacheable content expression determines whether the BIG-IP Cache Controller redundant system directs a given request to the cache server or to the origin server, based on evaluating variables in the HTTP header of the request.

Any content that does not meet the criteria in the cacheable content expression is deemed non-cacheable.

For example, in the configuration illustrated in this chapter, the cacheable content expression includes content having the file extension **.html** or **.gif**. The BIG-IP Cache Controller redundant system considers any request for content having a file extension other than **.html** or **.gif** to be non-cacheable, and sends such requests directly to the origin server.

For your configuration, you may want to cache any content that is not dynamically generated.

Required attributes

The cache control rule must include the following attributes:

- **origin_pool**
  Specifies a pool of servers that contain original copies of all content. Requests are load balanced to this pool when any of the following are true:
  - The requested content does not meet the criteria in the cacheable content condition.
  - No cache server is available.
  - The BIG-IP Cache Controller redundant system is redirecting a request from a cache server that did not have the requested content.
- **cache_pool**
  Specifies a pool of cache servers to which requests are directed in a manner that optimizes cache performance.

Optional attributes

The attributes in this section apply only if you are using the hot content load balancing feature.

◆ Note

In order to use the intelligent cache population feature, the cache_pool and the hot_pool must either be the same pool, or different pools referencing the same nodes.

- **hot_pool**
  Specifies a pool of cache servers to which requests are load balanced when the requested content is **hot**.

The **hot_pool** attribute is required if any of the following attributes is specified:

- **hot_threshold**
  Specifies the minimum number of requests for content in a given hot content set that causes the content set to change from **cool** to **hot** at the end of the period.
  If you specify a value for **hot_pool**, but do not specify a value for this variable, the cache statement uses a default hot threshold of 100 requests.
- **cool_threshold**
  Specifies the maximum number of requests for content in a given hot content set that causes the content set to change from **hot** to **cool** at the end of the hit period.
  If you specify a variable for **hot_pool**, but do not specify a value for this variable, the cache statement uses a default cool threshold of 10 requests.
- **hit_period**
  Specifies the period in seconds over which to count requests for particular content before determining whether to change the content demand status (**hot** or **cool**) of the content.
  If you specify a value for **hot_pool**, but do not specify a value for this variable, the cache statement uses a default hit period of 60 seconds.

❖ **content_hash_size**
Specifies the number of units, or ***hot content subsets***, into which the content is divided when determining whether content demand status is **hot** or **cool**. The requests for all content in a given subset are summed, and a content demand status (**hot** or **cool**) is assigned to each subset. The **content_hash_size** should be within the same order of magnitude as the actual number of requests possible. For example, if the entire site is composed of 500,000 pieces of content, a **content_hash_size** of 100,000 would be typical.
If you specify a value for **hot_pool**, but do not specify a value for this variable, the cache statement uses a default hash size of 1028 subsets.

Content demand status

Content demand status is a measure of the frequency with which a given hot content subset is requested. Content demand status, which is either **hot** or **cool**, is applicable only when using the hot content load balancing feature. For a given hot content subset, content demand status is **cool** from the time the cache control rule is implemented until the number of requests for the subset exceeds the **hot_threshold** during a **hit_period**. At this point content demand status for the subset becomes **hot**, and requests for any item in the subset are load balanced to the **hot_pool**. Content demand status remains **hot** until the number of requests for the subset falls below the **cool_threshold** during a **hit_period**, at which point the content demand status becomes **cool**. The BIG-IP Cache Controller the directs requests for any item in the subset to the appropriate server in the **cache_pool** until such time as the subset becomes **hot** again.

To create a cache statement rule using the Configuration utility

1. In the navigation pane, click **Rules**.
The Rules screen opens.
2. In the toolbar, click the **Add Rule** button.
The Add Rule screen opens.

3. In the Add Rule screen, type the cache statement.
   For the configuration shown in Figure 2.1, to cache all content having either the file extension **.html** or **.gif**, you would type:

```
rule cache_rule { cache ( http_uri ends_with "html" or http_uri
   ends_with "gif" ) { origin_pool origin_server cache_pool
   cache_servers hot_pool hot_cache_servers } }
```

4. Click the **Add** button.

To create a cache statement rule from the command line

To create a cache statement rule from the command line, use the following syntax:

```
bigpipe 'rule <rule_name> { cache  ( <condition>  ) { origin_pool
   <origin_pool_name> cache_pool <cache_pool_name> hot_pool
   <hot_pool_name> hot_threshold <hot_threshold_value>
   cool_threshold <cool_threshold_value> hit_period
   <hit_period_value> content_hash_size <content_hash_size_value>
   } }'
```

Given the configuration shown in Figure 2.1, to cache all content having the file extension **.html** or **.gif**, you would use the **bigpipe** command:

```
bigpipe 'rule cache_rule { cache ( http_uri ends_with "html" or
   http_uri ends_with "gif" ) { origin_pool origin_server
   cache_pool cache_servers hot_pool hot_cache_servers } }'
```

Creating a virtual server

Now that you have created pools and a cache statement rule to determine how the BIG-IP Cache Controller redundant system will distribute traffic in the configuration, you need to create a virtual server to use this rule and these pools. For this virtual server, use the host name or IP address that Internet clients use to request content from your site.

To create a virtual server using the 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 Add Virtual Server screen, configure the attributes you want to use with the virtual server.
   For additional information about configuring a virtual server, click the **Help** button.

   Configuration notes
   To create the configuration shown in Figure 2.1:

   - Add a virtual server with address **10.10.10.4** and port **80** (this means the virtual server will accept traffic for the HTTP service only).
   - Add the rule **cache_rule**.

To create a virtual server from the command line

Use the **bigpipe vip** command to configure the virtual server to use the pool that contains the outside addresses of the firewalls:

```
bigpipe vip <virtual server>:<service> <interface> use rule <rule
    name>
```

In the command, replace the parameters with the appropriate information:

- **<virtual server>** is an IP address appropriate to your network.
- **<service>** is a service you want to configure, such as **HTTP**; **FTP**, or **Telnet**.
- **<interface>** is the interface on the BIG-IP on which you want to create this virtual server.
- **<rule name>** is the name of the rule you want this virtual server to use.

To implement the configuration shown in Figure 2.1, you would use the command:

```
bigpipe vip 10.10.10.4:80 use rule cache_rule
```

Configuring for intelligent cache population

Your cache control rule routes requests to either the origin server or to the appropriate cache server.

When the cache control rule directs a request from a user to the origin server, the BIG-IP Cache Controller translates the destination of the request to the origin server and translates the source of the request to the translated address and port of the associated Secure Network Address Translation (SNAT) connection. This ensures that the request reaches the origin server and that the origin server responds to the BIG-IP Cache Controller and not directly to the user.

When the cache control rule directs a request from a user to the cache server, the cache will not contain the requested content if either it is the first time a cache has received a request for the content or the content has expired. In this case, the cache initiates a miss request (that is, a request resulting from a request for content a cache does not have) for this content to the origin server specified in the configuration of the cache or to another cache server. If you want to allow intelligent cache population, you should configure the cache with its origin server set to be the virtual server on the BIG-IP Cache Controller, so that the cache sends miss requests to the internal shared interface of the BIG-IP Cache Controller. The BIG-IP Cache Controller translates the destination of the request, and sends the request to either the origin server or another cache server that already has the requested content.

To ensure that the origin server or cache server responds to the BIG-IP Cache Controller rather than to the original cache server that generated the miss request, the BIG-IP Cache Controller also translates the source of the miss request to the translated address and port of the associated SNAT connection.

In order to enable these scenarios, you must:

- Create a SNAT on the BIG-IP Cache Controller.
- Enable source translation, but disable source processing on the external interface of the BIG-IP Cache Controller.
- Enable destination and source processing on the internal interface of the BIG-IP Cache Controller.

❖ Mark the origin server node as remote.

Configuring a SNAT

A Secure Network Address Translation (SNAT) translates the address of a packet from the cache server to the address you specify. For more information about SNATs, see *Configuring SNAT address mappings*, on page 4-27.

To configure a SNAT mapping using the Configuration utility

1. In the navigation pane, click **Secure NATs**.
   The Secure Network Address Translations screen opens.
2. On the toolbar, click **Add SNAT**.
   The Add SNAT screen opens.
3. In the Add SNAT screen, configure the attributes required for the SNAT you want to add.
   For additional information about configuring a pool, click the **Help** button.

   Configuration notes
   To create the configuration shown in Figure 2.1, use the translation address **10.10.10.5**.

To configure a SNAT mapping on the command line

The **bigpipe snat** command defines one SNAT for one or more node addresses.

```
bigpipe snat map <node addr>... <node addr> to <SNAT addr>
```

To implement the configuration shown in Figure 2.1, you use the command:

```
bigpipe snat map default to 10.10.10.5
```

Configuring interfaces

Typically, a BIG-IP Cache Controller has two interfaces:

- An external interface, typically set for ***destination processing***. Destination processing means that the interface can rewrite the destination address of an incoming packet, and allows initiation of virtual server connections.
- An internal interface, typically set for ***source processing***. Source processing means that the interface can rewrite the source of an incoming packet, and allows initiation of SNAT connections.

Configuring the external interface

In this configuration, both the origin web server and the content-requesting users are remote (that is, on the Internet, as opposed to on the intranet that includes the BIG-IP Cache Controller redundant system and the cache servers). Thus, when a BIG-IP Cache Controller directs a user content request to the origin web server, in order to prevent the origin web server from responding directly to the user, the BIG-IP Cache Controller must translate the source of the request.

To translate the source of requests to the origin server without translating the source of any other requests, you must enable source translation and confirm that source processing is disabled on the external interface.

To configure the external interface using the 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. In the Network Interface Card Properties screen, configure the attributes required for the interface.
   For additional information about creating a pool, click the **Help** button.

   Configuration notes
   To create the configuration shown in Figure 2.1, for **exp0**, make sure the **Enable Source Processing** is not checked and the **Enable Source Translation** box is checked.

To configure the external interface from the command line

Use the **bigpipe interface** command with the **source** keyword to enable or disable source processing for an interface:

```
bigpipe interface <interface> dest <enable><disable>
```

Use the **bigpipe interface** command with the **source** keyword to enable or disable source translation for an interface:

```
bigpipe interface <interface> source_translation <enable><disable>
```

To implement the configuration shown in Figure 2.1 from the command line, use the commands:

```
bigpipe interface exp0 source disable
bigpipe interface exp0 source_translation enable
```

Configuring the internal interface

In order to accommodate the case in which a cache server does not contain the requested content and sends a request for the content to the BIG-IP Cache Controller redundant system, you must add destination processing to the internal interface. Adding destination processing to the internal interface enables the BIG-IP Cache Controller to send a request for the content to either another cache server or to the origin web server.

To add destination processing to the internal interface using the 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. In the Network Interface Card Properties screen, configure the attributes required for the interface.
   For additional information about creating a pool, click the **Help** button.

   Configuration notes
   To create the configuration shown in Figure 2.1, for **exp1**, make sure the **Enable Destination Processing** check boxes is checked.

To add destination processing to the internal interface from the command line

Use the **bigpipe interface** command with the **source** keyword to turn destination processing on or off for an interface:

```
bigpipe interface <interface> dest <enable><disable>
```

To implement the configuration shown in Figure 2.1 from the command line, you use the command:

```
bigpipe interface exp1 dest enable
```

Marking the origin server node as remote

In order to initiate SNAT connection whenever the BIG-IP Cache Controller directs a connection to the remote web server, you must mark the web server node as remote.

To mark a node as remote using the Configuration utility

1. In the navigation pane, click **Nodes**.
   The Node list screen opens.
2. On the Node list, click the node you want to modify.
   The Node Properties screen opens.
3. On the Node Properties screen, configure the attributes required for the interface.
   For additional information about creating a pool, click the **Help** button.

   Configuration notes
   To create the configuration shown in Figure 2.1, for **192.168.10.1**, make sure the **Remote** box is checked.

To mark a node as remote from the command line

Use the **bigpipe node** command to configure the node.

```
bigpipe <node addr> remote
```

To implement the configuration shown in Figure 2.1, you use the command:

```
bigpipe 192.168.10.1 remote
```

3

Configuring Forward Proxy Caching

- Introducing forward proxy caching
- Configuration tasks
- Creating pools
- Creating a cache control rule
- Creating a virtual server

Introducing forward proxy caching

This chapter explains how to set up a ***forward proxy caching*** configuration, in which a BIG-IP Cache Controller redundant system uses content-aware traffic direction to enhance the efficiency of an array of cache servers storing Internet content for internal users. This type of configuration is useful for any enterprise that wants to increase the speed with which its users receive content requests from the Internet.

The configuration detailed in this chapter uses the following BIG-IP Cache Controller features:

- **Cacheable content determination**
  Cacheable content determination enables you to determine the type of content you cache on the basis of any combination of elements in the header of an HTTP request.
- **Content affinity**
  Content affinity ensures that a given subset of content remains associated with a given cache to the maximum extent possible, even when cache servers become unavailable, or are added or removed. This feature also maximizes efficient use of cache memory.
- **Hot content load balancing**
  Hot content load balancing identifies ***hot***, or frequently requested, content on the basis of number of requests in a given time period for a given ***hot content subset***. A hot content subset is different from, and typically smaller than, the content subsets used for content striping. Requests for hot content are redirected to a cache server in the ***hot pool***, a designated group of cache servers. This feature maximizes the use of cache server processing power without significantly affecting the memory efficiency gained by content affinity.

Maximizing memory or processing power

From the time you implement a cache control rule until such time as a hot content subset becomes **hot**, the content is divided across your cache servers, so that no two cache servers contain the same content. In this way, efficient use of the cache servers' memory is maximized.

After a hot content subset becomes **hot**, requests for any content contained in that subset are load balanced, so that, ultimately, each cache server contains a copy of the hot content. The BIG-IP Cache Controller distributes requests for the hot content among the cache servers. In this way, efficient use of the cache servers' processing power is maximized.

Thus, for a particular content item, the BIG-IP Cache Controller maximizes either cache server memory (when the content is **cool**) or cache server processing power (when the content is **hot**), but not both at the same time. The fact that content is requested with greatly varying frequency enables the cache statement rule to evaluate and select the appropriate attribute to maximize for a given content subset.

Using the configuration diagram

Figure 3.1, following, illustrates a forward proxy caching configuration, and provides an example configuration for this entire chapter. Remember that this is just a sample: when creating your own configuration, you must use IP addresses, host names, and so on, that are applicable to your own network.

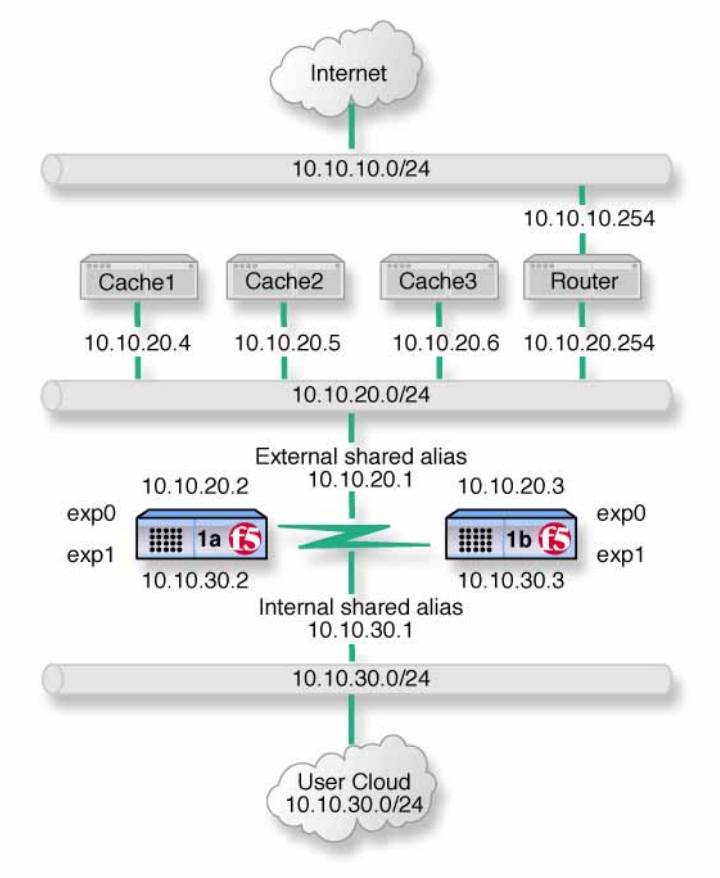

Figure 3.1 *Caching Internet content*

Configuration tasks

To configure forward proxy caching, complete the following tasks in order:

- Create pools

- Create a cache control rule
- Create a virtual server

Each of the following sections explains one of these tasks, and shows how you perform the tasks in order to implement the configuration shown in Figure 3.1. Note that in this example, as in all examples in this guide, we use only non-routable IP addresses. In a real topology, the appropriate IP addresses have to be routable on the Internet.

Creating pools

For this configuration, you create load balancing ***pools*** for your ***origin server*** (in this configuration, the origin server is the router that provides access to the Internet), for your cache servers, and for your ***hot***, or frequently requested, content servers, which may or may not be cache servers. A pool is a group of devices to which you want the BIG-IP Cache Controller redundant system to direct traffic. For more information about pools, refer to *Configuring a pool*, on page 4-5.

You create three pools:

- **Cache server pool**
  The BIG-IP Cache Controller directs all cacheable requests bound for your web server to this pool, unless a request is for hot content.
- **Origin server pool**
  This pool includes your origin web server. Requests are directed to this pool when:
  - The request is for ***non-cacheable*** content; that is, content that is not identified in the ***cacheable content expression*** part of a cache rule statement. For more information, see *Cacheable content expression*, on page 3-8.
  - The request is from a cache server that does not yet contain the requested content, and no other cache server yet contains the requested content.
  - No cache server in the cache pool is available.

❖ **Hot cache servers pool**
If a request is for frequently requested content, the BIG-IP Cache Controller redundant system directs the request to this pool.

◆ Note

While the configuration shown in Figure 3.1 implements a hot cache servers pool, this pool is not required if you want to use the content determination and content affinity features. However, you must implement this pool if you want to use the hot content load balancing feature.

Creating a pool for the cache servers

First, create a pool for the cache servers. Use either the Configuration utility or the command line to create this pool.

To create a pool using the Configuration utility

1. In the navigation pane, click **Pools**.
   The Pools screen opens.
2. In the toolbar, click the **Add Pool** button.
   The Add Pool screen opens.
3. In the Add Pool screen, configure attributes required for the cache servers you want to add to the pool.
   For additional information about configuring a pool, click the **Help** button.

   Configuration notes
   To create the configuration shown in Figure 3.1:

   - Create a pool named **cache_servers**.
   - Add each cache server from the example, **10.10.20.4**, **10.10.20.5,** and **10.10.20.6**, to the pool. For each cache server you add to the pool, specify port **80**, which means this cache server accepts traffic for the HTTP service only.

To create a pool from the command line

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

```
bigpipe pool <pool_name> { lb_method <lb_method> member
   <member_definition> ... member <member_definition> }
```

To implement the configuration shown in Figure 3.18, you use the command:

```
bigpipe pool cache_servers { lb_method round_robin member
   10.10.20.4:80 member 10.10.20.5:80 member 10.10.20.6:80 }
```

Creating a pool for the origin server

Next, create a pool for your origin server. In this configuration, the origin server is the router between the cache servers and the Internet. Use either the Configuration utility or the **bigpipe pool** command, as you did to create the pool for the cache servers.

To create a pool using the Configuration utility

1. In the navigation pane, click **Pools**.
   The Pools screen opens.
2. In the toolbar, click the **Add Pool** button.
   The Add Pool screen opens.
3. In the Add Pool screen, configure the attributes required for the cache servers you want to add to the pool.
   For additional information about configuring a pool, click the **Help** button.

   Configuration notes
   To create the configuration shown in Figure 3.1:

   - Create a pool named **origin_server**.
   - Add the origin server from the example, the router **10.10.20.254**, to the pool. Specify port **80**, which means the origin server accepts traffic for the HTTP service only.

To create a pool from the command line

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

```
bigpipe pool <pool_name> { lb_method <lb_method> member
   <member_definition> ... member <member_definition> }
```

To implement the configuration shown in Figure 3.1, you use the command:

```
bigpipe pool origin_server { lb_method round_robin member
   10.10.20.254:80 }
```

Creating a pool for hot content

Finally, create a pool for hot content. You can use either the Configuration utility or the command line to create this pool, as in the previous sections.

To create a pool using the Configuration utility

1. In the navigation pane, click **Pools**.
   The Pools screen opens.
2. In the toolbar, click the **Add Pool** button.
   The Add Pool screen opens.
3. In the Add Pool screen, configure the attributes required for the cache servers you want to add to the pool.
   For additional information about configuring a pool, click the **Help** button.

   Configuration notes
   To create the configuration shown in Figure 3.1:

   - Create a pool named **hot_cache_servers**.
   - Add each cache server from the example, **10.10.20.4**, **10.10.20.5,** and **10.10.20.6**, to the pool. For each cache server you add to the pool, specify port **80**, which means this cache server accepts traffic for the HTTP service only.

To create a pool from the command line

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

```
bigpipe pool <pool_name> { lb_method <lb_method> member
   <member_definition> ... member <member_definition> }
```

To implement the configuration shown in Figure 3.1, use the command:

```
bigpipe pool hot_cache_servers { lb_method round_robin member
   10.10.20.4:80 member 10.10.20.5:80 member 10.10.20.6:80 }
```

Creating a cache control rule

A cache control rule is a specific type of rule. A rule establishes criteria by which a BIG-IP Cache Controller directs traffic. A ***cache control rule*** determines where and how the BIG-IP Cache Controller redundant system directs content requests in order to maximize the efficiency of your cache server array and of your origin web server.

A cache control rule includes a ***cache statement***, which is composed of a cacheable content expression and two ***attributes***. An attribute is a variable that the cache statement uses to direct requests. It can also include several optional attributes.

A cache statement may be either the only statement in a rule, or it may be nested in a rule within an **if** statement.

Cacheable content expression

The cacheable content expression determines whether the BIG-IP Cache Controller redundant system directs a given request to the cache server or to the origin server, based on evaluating variables in the HTTP header of the request.

Any content that does not meet the criteria in the cacheable content expression is deemed non-cacheable.

For example, in the configuration illustrated in this chapter, the cacheable content expression includes content having the file extension **.html** or **.gif**. The BIG-IP Cache Controller redundant system considers any request for content having a file extension other than **.html** or **.gif** to be non-cacheable, and sends such requests directly to the origin server.

For your configuration, you may want to cache any content that is not dynamically generated.

Required attributes

The cache control rule must include the following attributes:

- **origin_pool**
  Specifies a pool of servers that contain original copies of all content. Requests are load balanced to this pool when any of the following are true:
  - The requested content does not meet the criteria in the cacheable content condition.
  - No cache server is available.
  - The BIG-IP Cache Controller redundant system is redirecting a request from a cache server that did not have the requested content.
- **cache_pool**
  Specifies a pool of cache servers to which requests are directed in a manner that optimizes cache performance.

Optional attributes

The attributes in this section apply only if you are using the hot content load balancing feature.

- **hot_pool**
  Specifies a pool of cache servers to which requests are load balanced when the requested content is **hot**.

  The **hot_pool** attribute is required if any of the following attributes is specified:

- **hot_threshold**
  Specifies the minimum number of requests for content in a given hot content set that causes the content set to change from **cool** to **hot** at the end of the period.
  If you specify a value for **hot_pool**, but do not specify a value for this variable, the cache statement uses a default hot threshold of 100 requests.
- **cool_threshold**
  Specifies the maximum number of requests for content in a given hot content set that causes the content set to change from **hot** to **cool** at the end of the hit period.
  If you specify a variable for **hot_pool**, but do not specify a value for this variable, the cache statement uses a default cool threshold of 10 requests.
- **hit_period**
  Specifies the period in seconds over which to count requests for particular content before determining whether to change the content demand status (**hot** or **cool**) of the content.
  If you specify a value for **hot_pool**, but do not specify a value for this variable, the cache statement uses a default hit period of 60 seconds.
- **content_hash_size**
  Specifies the number of units, or ***hot content subsets***, into which the content is divided when determining whether content demand status is **hot** or **cool**. The requests for all content in a given subset are summed, and a content demand status (**hot** or **cool**) is assigned to each subset. The **content_hash_size** should be within the same order of magnitude as the actual number of requests possible. For example, if the entire site is composed of 500,000 pieces of content, a **content_hash_size** of 100,000 would be typical.
  If you specify a value for **hot_pool**, but do not specify a value for this variable, the cache statement uses a default hash size of 1028 subsets.

Content demand status

Content demand status is a measure of the frequency with which a given hot content subset is requested. Content demand status, which is either **hot** or **cool**, is applicable only when using the hot content load balancing feature. For a given hot content subset, content demand status is **cool** from the time the cache control rule is implemented until the number of requests for the subset exceeds the **hot_threshold** during a **hit_period**. At this point content demand status for the subset becomes **hot**, and requests for any item in the subset are load balanced to the **hot_pool**. Content demand status remains **hot** until the number of requests for the subset falls below the **cool_threshold** during a **hit_period**, at which point the content demand status becomes **cool**. The BIG-IP Cache Controller directs requests for any item in the subset to the appropriate server in the **cache_poo**l until such time as the subset becomes **hot** again.

To create a cache statement rule using the Configuration utility

1. In the navigation pane, click **Rules**.
   The Rules screen opens.
2. In the toolbar, click the **Add Rule** button.
   The Add Rule screen opens.
3. In the Add Rule screen, type the cache statement.
   For example, given the configuration shown in Figure 3.1, to cache all content having either the file extension **.html** or **.gif**, you would type:

```
rule cache_rule { cache ( http_uri ends_with "html" or http_uri
   ends_with "gif" ) { origin_pool origin_server cache_pool
   cache_servers hot_pool hot_cache_servers } }
```

4. Click the **Add** button.

To create a cache control rule from the command line

To create a cache statement rule from the command line, use the following syntax:

```
bigpipe 'rule <rule_name> { cache  ( <condition>  ) { origin_pool
   <origin_pool_name> cache_pool <cache_pool_name> hot_pool
   <hot_pool_name> hot_threshold <hot_threshold_value>
```

```
   cool_threshold <cool_threshold_value> hit_period
   <hit_period_value> content_hash_size <content_hash_size_value>
   } }'
```

For example, given the configuration shown in Figure 3.1, to cache all content having the file extension **.html** or **.gif**, you would use the **bigpipe** command:

```
bigpipe 'rule cache_rule { cache ( http_uri ends_with "html" or
   http_uri ends_with "gif" ) { origin_pool origin_server
   cache_pool cache_servers hot_pool hot_cache_servers } }'
```

Creating a virtual server

Now that you have created pools and a cache control rule to determine how the BIG-IP Cache Controller will distribute outbound traffic, you need to create a wildcard virtual server to process traffic using this rule and these pools.

To create a wildcard virtual server using the 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 Add Virtual Server screen, configure the attributes you want to use with the virtual server.
   For additional information about configuring a virtual server, click the **Help** button.

   Configuration notes

   - Add a virtual server with address **0.0.0.0** and port **0** (this designates a wildcard virtual server).
   - Add the rule **cache_rule**.

To create a wildcard virtual server from the command line

Use the **bigpipe vip** command to configure the virtual server to use the pool that contains the outside addresses of the firewalls:

```
bigpipe vip 0.0.0.0:0 <interface> use rule <rule name>
```

In the command, replace the parameters with the appropriate information:

- **<interface>** is the interface on the BIG-IP on which you want to create this virtual server.
- **<rule name>** is the name of the rule you want this virtual server to use.

To implement the configuration shown in Figure 3.1, you would use the command:

```
bigpipe vip 0.0.0.0:0 use rule cache_rule
```

F5 Networks BIG-IP Local Traffic Controller

4

Essential Configuration Tasks

- Determining which configuration tasks to do
- Configuring a pool
- Configuring virtual servers
- Allowing access to ports and servers
- Configuring the timer settings
- Changing the global load balancing mode
- Configuring NATs and IP forwarding for nodes
- Configuring Extended Content Verification service checking
- Configuring and synchronizing redundant systems

Determing which configuration tasks to do

Before you follow the instructions in this chapter, you need to browse through the prior chapters to find the specific load balancing solution you want to set up, such as basic web server load balancing. Each load balancing chapter describes the configuration tasks you need to complete to set up the solution, but it points you to this chapter for the actual configuration steps.

This chapter covers the four essential configuration tasks that all users must complete, regardless of the chosen load balancing solution. The chapter also includes optional configuration tasks that most users find they want to do. In the individual load balancing solution chapters, you can find information about which optional configuration tasks and advanaced features may be right for you.

Basic configuration tasks

- **Configure pools**
  A pool contains a group of servers or other network equipment that the BIG-IP Controller load balances. The pool configuration includes key information such as the load balancing mode and the persistence mode. The nodes that you add to a pool are known as ***members***.
- **Configure virtual servers**
  The virtual servers reference pools of servers, or network devices, that you want to load balance.
- **Allow access to ports and services**
  The services and ports on a BIG-IP Controller are locked down and cannot accept connections until you specifically open them to network access. For each service that one or more of your virtual servers supports, you need to open the corresponding port number for network access. However, ports are automatically enabled when you use them in virtual server definition in the Configuration utility.

- **Configure the timer settings**
  The BIG-IP Controller supports several timer settings, but for a simple configuration, there are only two that you need to set. First you need to set the amount of time that idle connections are allowed to remain open. Second, you need to set the frequency at which the BIG-IP Controller checks nodes to make sure they are up and available to accept connections passed on by a virtual server.

Optional configuration tasks

This chapter also covers additional configuration options that users typically add to a simple configuration, including:

- **Change the load balancing mode**
  You can use an alternate load balancing mode.
- **Configure NATs or IP forwarding**
  You can set up network address translation (NAT) or IP forwarding to allow direct connections to and from nodes.
- **Configure extended content verification service checking**
  You can configure extended content verification (ECV) to allow the BIG-IP Controller to verify that a server is responding to requests.
- **Set up persistence**
  You can set up persistence to accommodate e-commerce and other dynamic content sites that require returning clients to bypass load balancing and return to the same node to which they last connected.
- **Configure and synchronize redundant systems**
  You can configure and set up redundant BIG-IP Controller systems.

WARNING

*When you set configuration options in the Configuration utility, they are immediately saved to the appropriate configuration file. However, when you set configuration options using the **bigpipe** command line utility, they are temporarily stored in system memory, and are not saved to a configuration file unless you*

*execute the **bigpipe -s** command. For more information about this command, see the **BIG-IP Controller Reference Guide**, bigpipe Command Reference.*

Table 4.1 describes the different types of connection configurations available on the BIG-IP Controller.

	NAT	SNAT	IP Forwarding	Virtual server	Forwarding virtual server
Security	Medium	High	Low (see following note)	High	High
Routable addresses required on the internal network	No	No	Yes	No	Yes
Protocols	TCP and UDP	TCP and UDP	Any IP protocol	TCP and UDP	TCP and UDP
NT Domain support	No	No	Yes	No	Yes
Active FTP support	No	Yes	Yes	Yes	Yes
Connection origination	Any direction	One direction	Any direction	One direction	One direction
Ports	Does not matter	Does not matter	Does not matter	Uses specific ports or wildcard	Uses specific ports or wildcard
Setup for specific nodes or hosts	Yes	Yes, but can use wildcards	No	Yes, but can use wildcard	Yes
Load balancing	No	No	No	Yes	No

Table 4.1 *Connection configuration options for the BIG-IP Controller*

◆ Note

Although IP forwarding does not require setup for specific hosts, the BIG-IP Controller supports IP filters that you can configure to restrict traffic.

Configuring a pool

The first step in a basic configuration is to configure a pool of network devices, such as servers, firewalls, or caches. A pool contains the load balancing and persistence attributes for the members of the pool.

You can define pools from the command line, or define one using the web-based Configuration utility. This section describes how to define a simple pool using each of these configuration methods.

To create a pool using the Configuration utility

1. In the navigation pane, click **Pools**.
   The Pools screen opens.
2. In the toolbar, click the **Add Pool** button.
   The Add Pool screen opens.
3. In the Add Pool screen, configure the load balancing method, persistence attributes, and members for the pool.

   Configuration note

   - For additional information about configuring a pool, click the **Help** button.

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> member
   <member_definition> ... member <member_definition>}
```

For example, if you want to create the pool **my_pool**, with two members using Round Robin **(rr)** load balancing, from the command line, you would type the following command:

```
bigpipe pool my_pool { lb_mode rr member 11.12.1.101:80 member
   11.12.1.100:80 }
```

Configuring virtual servers

The second step in a basic configuration is to configure virtual servers. This means that you have already configured a pool of servers that you can reference with a virtual server. Before you configure virtual servers, you need to know:

- If standard virtual servers or wildcard virtual servers meet the needs of your network
- Whether you need to activate optional virtual server properties

Once you know which virtual server options are useful in your network, you can:

- Define standard virtual servers
- Define wildcard virtual servers

Using standard or wildcard virtual servers

Virtual servers reference a pool you create that contains a group of content servers, firewalls, routers, or cache servers, and they are associated with one or more external interfaces on the BIG-IP Controller.

You can configure two different types of virtual servers:

- **Standard virtual servers**
  A standard virtual server represents a site, such as a web site or an FTP site, and it provides load balancing for a pool of content servers. The virtual server IP address should be the same IP address that you register with DNS for the site that the virtual server represents.
- **Wildcard virtual servers**
  A wildcard virtual server load balances a pool of transparent network devices such as firewalls, routers, or cache servers. Wildcard virtual servers are configured with an IP address of 0.0.0.0, and sometimes with a virtual port of 0.

Note that both the Configuration utility and the **bigpipe** command line utility accept host names in place of IP addresses, and also accept standard service names in place of port numbers.

Using additional features with virtual servers

After you create a pool and define a virtual server that references the pool, you can set up additional features, such as network address translation (NAT) or extended content verification (ECV). If you are planning on using any of these features, you may want to read the corresponding section before you actually begin the virtual server configuration process.

- Network address translations (see *Configuring NATs and IP forwarding for nodes*, on page 4-22)
- Extended Content Verification service checking (see *Configuring Extended Content Verification service checking*, on page 4-29)
- Persistence for connections that should return to the node to which they last connected (see *Configuring and synchronizing redundant systems*, on page 4-36)

Defining standard virtual servers

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

◆ Note

If you are using a 3-DNS Controller in conjunction with the BIG-IP Controller, the 3-DNS Controller uses the IP address associated with the registered domain name in its own configuration. For details, refer to the ***3-DNS Controller Administrator Guide****.*

To define a standard virtual server that references a pool using the Configuration utility

1. In the navigation pane, click **Virtual Servers**.
   The Virtual Servers screen opens.
2. On the toolbar, click **Add Virtual Server**.
   The Add Virtual Server screen opens.

3. In the Add Virtual Server screen, configure the attributes that you want to use with the virtual server.

Configuration notes

- For additional information about configuring a virtual server, click the **Help** button.

To define a standard virtual server mapping from the command line

Type the **bigpipe vip** command as shown below. If you have DNS configured, you can also use host names in place of IP addresses. You can use standard service names in place of port numbers.

```
bigpipe vip <virt IP>:<port> use pool <pool_name>
```

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

```
bigpipe vip 192.200.100.25:80 use pool my_pool
```

Defining wildcard virtual servers

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

However, when you are load balancing transparent nodes, a client's destination IP address is going to seem random. The client is connecting to an IP address on the other side of the firewall, router, or proxy server. In this situation, the BIG-IP Controller cannot match the client's destination IP address to a virtual server IP

address. Wildcard virtual servers resolve this problem by not translating the incoming IP address at the virtual server level on the BIG-IP Controller. For example, when the BIG-IP Controller does not find a specific virtual server match for a client's destination IP address, it matches the client's IP address to a wildcard virtual server. The BIG-IP Controller then forwards the client's packet to one of the firewalls or routers that the wildcard virtual server load balances, which in turn forwards the client's packet to the actual destination IP address.

A note about wildcard ports

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

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

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

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

◆ Note

*When you define a virtual server with port translation turned **off**, and you want to perform a service check on that node, you must configure service check intervals and timeouts using the port*

specified for the node. After the timeouts are configured, you can configure a service check. See Service checking for wildcard servers and ports, on page 4-19, for more details.

Defining the wildcard virtual server mappings

There are two procedures required to set up a wildcard virtual server. First, you must define the wildcard virtual server. Then you must turn off port translation for the virtual server.

The following sections describe the two procedures used to set up a wildcard virtual server using the configuration utility.

To define a wildcard virtual server mapping using the 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 Add Virtual Server screen, configure the attributes for the virtual server.
   For additional information about the options on this screen, click the **Help** button.

   Configuration notes

   - Note that port **0** defines a wildcard virtual server that handles all types of services. If you specify a port number, you create a port-specific wildcard virtual server. The wildcard virtual server only handles traffic for the port specified.

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

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

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

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

Configuration notes
For additional information about the options on this screen, click the **Help** button.

To define a wildcard virtual server mapping from the command line

There are three commands required to set up a wildcard virtual server. First, you must define a pool that contains the addresses of the transparent devices. Next, you must define the wildcard virtual server. Then you must turn port translation off for the virtual server. To define the pool of transparent devices, use the **bigpipe pool** command. For example, you can create a pool of transparent devices called **transparent_pool** that uses the Round Robin load balancing mode:

```
bigpipe pool transparent_pool { lb_mode rr member
   <member_definition>... member <member_definition> }
```

To define the virtual server, use the **bigpipe vip** command:

```
bigpipe vip <virtual IP>:<port> use pool <pool_name>
```

After you define the virtual server, you can enable or disable port translation using the following command:

```
bigpipe vip <virtual IP>:<port> translate port enable | disable
```

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

```
bigpipe pool transparent_pool { lb_mode rr member 10.10.10.101:80
   member 10.10.10.102:80 member 10.10.10.103:80 }
```

After you create the pool of transparent nodes, use the following command to create a wildcard virtual server that maps to the pool **transparent_pool**. Because the members are firewalls and need to handle a variety of services, the virtual server is defined using port

0 (or ***** or **any**). You can specify any valid non-zero port for the node port and then turn off port translation for that port. In this example, service checks ping port 80.

```
bigpipe vip 0.0.0.0:0 use pool transparent_pool
```

After you define the virtual server, turn off port translation for the port in the virtual server definition. In this example, port 80 is used for service checking. If you do not turn off port translation, all incoming traffic is translated to port 80.

```
bigpipe vip 0.0.0.0:0 translate port disable
```

Allowing access to ports and services

One of the security features of the BIG-IP Controller is that all ports on the controller are locked down and unavailable for service unless you specifically open them to network access. Before clients can use the virtual servers you have defined, you must allow access to each port that the virtual servers use.

This is the third task of the four essential tasks you must complete for a basic configuration. You must perform this task after you create a pool and a virtual server that references the pool, and before you configure the timer settings.

◆ Tip

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

To allow access to services using the Configuration utility

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

To allow access to services from the command line

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

```
bigpipe port <port>... <port> enable
```

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

```
bigpipe port 80 23 443 enable
```

WARNING

*In order for FTP to function properly, you must allow both ports 20 and 21 (or **ftp-data** and **ftp**).*

Configuring the timer settings

Configuring timer settings is the fourth task of the four essential tasks you must complete for a basic configuration. You must perform this task after you configure virtual servers and after you allow access to services and ports.

There are two essential timer settings that you need to configure:

- The node ping timer defines how often the BIG-IP Controller will ping node addresses to verify whether a node is **up** or **down**. It also defines how long the BIG-IP Controller waits for a response from a node before determining that the node is unresponsive and marking the node **down**.
- The idle connection timer defines how long an inactive connection is allowed to remain open before the BIG-IP Controller deletes the record of the connection, closing it and disconnecting the client.

The service check timer is optional, and you need to set it only if you want the BIG-IP Controller to check to see if a service, or even specific content, is available on a particular node.

Note

If you plan to use simple service checks, or ECV or EAV service checks, you need to set the service check timer.

Setting the node ping timer

The node ping timer is an essential setting on the BIG-IP Controller that determines how often the BIG-IP Controller checks node addresses to see whether they are **up** and available or **down** and unavailable. The node ping timer setting applies to all nodes configured for use by the BIG-IP Controller, and it is part of the BIG-IP Controller system properties.

Note

The ping interval should be set to occur about three times during every timeout period. For example, if you set the ping value to 5 seconds, we recommend that you set the timeout to 16 seconds.

To set the node ping timer using the Configuration utility

1. In the navigation pane, click the BIG-IP Controller icon. The BIG-IP System Properties screen opens.
2. In the Node Ping section of the table, in the **Ping** box, type the frequency (in seconds) at which you want the BIG-IP Controller to ping each node address it manages. A setting of 5 seconds is adequate for most configurations.

3. In the Node Ping section of the table, in the **Timeout** box, type the number of seconds you want the BIG-IP Controller to wait to receive a response to the ping.
   For additional information about the options on this screen, click the **Help** button.

   Configuration notes

   - If the BIG-IP Controller does not receive a response to the ping before the node ping timeout expires, the BIG-IP Controller marks the node **down** and does not use it for load balancing. A setting of 16 seconds is adequate for most configurations

To set the node ping timer from the command line

To define node ping settings, you use two commands. First, you set the node ping frequency using the **bigpipe tping_node** command, and then you set the node ping timer using the **bigpipe timeout_node** command.

```
bigpipe tping_node <seconds>
bigpipe timeout_node <seconds>
```

For example, the following commands sets the ping frequency at 5 seconds, and the timer to 16 seconds, which should be adequate for most configurations.

```
bigpipe tping_node 5
bigpipe timeout_node 16
```

Setting the timer for reaping idle connections

The BIG-IP Controller supports two timers for reaping idle connections, one for TCP traffic and one for UDP traffic. These timers are essential, and if they are set too high, or not at all, the BIG-IP Controller may run out of memory. Each individual port on the BIG-IP Controller has its own idle connection timer settings.

You can set the timers by using the Configuration utility or from the command line, however, please note that while it is a one-step process to set the timers using the Configuration utility, it is a two-step process from the command line.

WARNING

The BIG-IP Controller accepts UDP connections only if you set the UDP idle connection timer.

To set the inactive connection timer using the Configuration utility

1. In the navigation pane, click the expand button (+) next to **Virtual Servers**.
   The Virtual Server tree opens and displays the Ports option.
2. Click **Ports**.
   The Global Virtual Ports screen opens.
3. In the Global Virtual Ports screen, click the port number or service name for which you want to configure the idle connection timeout.
   For additional information about the options on this screen, click the **Help** button.

   Configuration notes

   - For the HTTP connections, we recommend setting the **Idle Connection Timeout TCP** to 60 seconds. For other services such as Telnet, higher settings may be necessary.
   - In the **Idle Connection Timeout UDP** box, type the number of seconds you want to elapse before the BIG-IP Controller drops UDP connections.

To set TCP idle connection timers from the command line

You need to set the timers for TCP and UDP using a two-step processes if you do it from the command line.

Use the **bigpipe treaper** to define a TCP idle connection timeout for one or more ports at a time. For HTTP connections we recommend only 60 seconds, but for other services such as Telnet we recommend higher settings. The default setting for this timer is 16 minutes (1005 seconds). Use the following syntax for this command:

```
bigpipe treaper <port>... <port> <seconds>
```

For example, the following command sets a 120 second time limit for idle connections on port 443:

```
bigpipe treaper 443 120
```

To set UDP idle connection timers from the command line

You can define a UDP idle connection timeout for one or more ports at a time using the **bigpipe udp** command.

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

For example, the following command sets a 120-second time limit for idle connections on port 53:

```
bigpipe udp 53 120
```

Setting the service check timer

The service check feature is similar to node ping, but instead of testing the availability of a server, it tests the availability of a particular service running on a server. The service check timer affects the three different types of service checks: simple service check, ECV service check, and EAV service check. To set up simple service check, you need only set the service check timer as described below. To set up ECV service check or EAV service check, however, you need to configure additional settings (see *Configuring Extended Content Verification service checking*, on page 4-29).

Note that each individual service managed by the BIG-IP Controller has its own service check timer settings.

To set the service check timer using the Configuration utility

1. In the navigation pane, click the expand button (+) next to **Nodes**.
   The Nodes tree opens and displays the Ports option.
2. Click **Ports**.
   The Global Node Ports screen opens.
3. In the Global Node Port Properties screen, click the port for which you want to configure the service check timer.
   For additional information about the options on this screen, click the **Help** button

   Configuration notes

   - For the **Frequency** setting, we recommend 5 seconds for most configurations.
   - For the **Timeout** setting, we recommend 16 seconds for most configurations.

To set the service check timer on the command line

To define service check settings, you actually use two commands. First, you set the service check frequency using the **bigpipe tping_svc** command, and then set the service check timer using the **bigpipe timeout_svc** command.

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

For example, the following sequence of commands sets the service check frequency at 5 seconds, and the timer to 16 seconds, which is adequate for most configurations.

```
bigpipe tping_svc 80 5
bigpipe timeout_svc 80 16
```

Service checking for wildcard servers and ports

When you configure a wildcard virtual server with a **0** port using nodes with standard ports, such as 80, with port translation turned **off**, the BIG-IP Controller uses the standard service check timeout values (port 80, for example) to service check the port. For more information about setting the service check timer, see *Setting the service check timer*, on page 4-17.

Using the simple keyword

The simple keyword is being phased out in future releases. This information is provided in order to support existing configurations.

The **simple** keyword is necessary only if you specified a node port of **0**. In previous versions of the BIG-IP Controller, this was the only way to set up a wildcard virtual server that handled connections for all services. However, we now recommend that you specify a node port and then turn off port translation for the virtual server.

To set up a simple service check for this type of virtual server, add the following entry to the **/etc/bigd.conf** file. Use the following syntax to set a check on a node where the check port is not the node port:

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

For example, a wildcard server is defined with a wildcard port, like this:

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

In this case, you must use the **simple** keyword to designate the wildcard <**server:**><**port**> and <**check port**> for the service check:

```
simple n1:0 80
```

Changing the global load balancing mode

Changing the global load balancing mode is one of the optional tasks you can perform after you have completed the four main tasks of a basic configuration. This means you already have:

- Configured pools
- Configured virtual servers
- Configured access to ports and services
- Configured the timer settings

After you complete the basic tasks, you can change the global load balancing mode. The default global load balancing mode on the BIG-IP Controller is Round Robin, and it simply passes each new connection request to the next server in line, eventually distributing connections evenly across the array of machines being load balanced. Round Robin mode works well in most configurations, especially if the equipment that you are load balancing is roughly equal in processing speed and memory. If you want to use the Round Robin load balancing mode, you can skip this section, and begin configuring features that you want to add to the basic configuration.

However, if you are working with servers that differ significantly in processing speed and memory, you may want to switch to Ratio load balancing mode. In Ratio mode, the BIG-IP Controller distributes connections among machines according to ratio weights

that you define, where the number of connections that each machine receives over time is proportionate to the ratio weight you define for each machine.

◆ Tip

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

◆ Note

*The BIG-IP Controller also supports more advanced dynamic load balancing modes that may be suitable for your site. These modes include specific member load balancing modes that you can assign to specific pools. Refer to the **BIG-IP Controller Reference Guide**, for more information about working with specialized load balancing modes.*

Using Ratio mode

If you want to switch the load balancing method used in a pool from Round Robin to Ratio you must modify the pool specification using the Configuration utility or from the command line. You change the load balancing mode to **ratio member**, and you must assign a ratio weight to each member of the pool.

Switching to Ratio mode

First, you should set the load balancing mode to Ratio. The load balancing mode is actually a property of the BIG-IP Controller system, and it applies to all virtual servers defined on the system.

To switch the system to Ratio mode using the Configuration utility

1. In the navigation pane, click **Pools**.
   The Pools screen opens.

2. In the toolbar, click the **Add Pool** button.
   The Add Pool screen opens.
3. Use the resources options to set the **Ratio** value for each member in the pool. In the **Ratio** box, type in a number to assign a ratio to this node within the pool. For example, if you are using the ratio load balancing mode 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**.
   For additional information about the options on this screen, click the **Help** button.

To switch the pool to Ratio mode on the command line

To switch the pool use the **modify** keyword with the **bigpipe pool** command. For example, if you want change the pool **my_pool**, to use the **ratio_member** load balancing mode and assign ratio values to members, you can type the following command:

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

Configuring NATs and IP forwarding for nodes

Configuring NATs and IP forwarding are optional tasks you can configure after you have completed the four main tasks of a basic configuration. This means you already have:

- Configured pools
- Configured virtual servers
- Configured access to ports and services
- Configured the timer settings

After you complete the basic tasks, you can configure network address translation and IP forwarding on the BIG-IP Controller.

The IP addresses that identify nodes on the BIG-IP Controller's internal network need not be routable on the external network. This protects nodes from illegal connection attempts, but it also prevents

nodes (and other hosts on the internal network) from receiving direct administrative connections, or from initiating connections to clients, such as mail servers or databases, on the BIG-IP Controller's external interface (destination processing).

Using network address translation resolves this problem. Network address translations (NATs) assign to a particular node a routable IP address that the node can use as its source IP address when connecting to servers on the BIG-IP Controller's external interface. You can use the NAT IP address to connect directly to the node through the BIG-IP Controller, rather than having the BIG-IP Controller send you to a random node according to the load balancing mode. IP forwarding provides functionality similar to a NAT. If your network does not support NATs, you may want to consider using IP forwarding.

◆ Note

In addition to these options, you can set up forwarding virtual servers which allow you to selectively forward traffic to specific addresses. The BIG-IP Controller maintains statistics for forwarding virtual servers. For more information about forwarding virtual servers, see the ***Reference Guide****.*

There are three configuration options on the BIG-IP Controller that you can use to control network access, and you need to identify which method is suitable for your needs:

- **Network Address Translation (NAT)**
  A network translation address provides a routable alias IP address that a node can use as its source IP address when making or receiving connections to clients on the external network. You can configure a unique NAT for each node address included in a virtual server mapping.

  NATs do not support port translation, and are not appropriate for FTP. You cannot define a NAT if you configure a default SNAT.

- **Secure Network Address Translation (SNAT)**
  A secure network address translation provides functionality similar to that of firewalls. A SNAT defines a routable alias IP address that one or more nodes can use as a source IP address

only when making connections to hosts on the external network. SNAT addresses support port translation, and they also prevent hosts on the external network from connecting directly to the node.

SNAT only supports TCP and UDP. SNAT also features support for both passive and active FTP. You cannot define a NAT if you define a default SNAT.

- **IP forwarding**
  IP forwarding does not translate node addresses. Instead, it simply exposes the node's IP address to the BIG-IP Controller's external network so that clients can use it as a standard routable address. When you turn IP forwarding on, the BIG-IP Controller acts as a router when it receives connection requests for node addresses. IP forwarding itself does not provide security features, but you can use the IP filter feature to implement a layer of security (see *Setting up IP forwarding*, on page 4-27) that can help protect your nodes.

WARNING

NATs and SNATs do not support the NT Domain or CORBA protocols. Instead of using NATs or SNATs, you need to configure IP forwarding (see Setting up IP forwarding, on page 4-27).

Defining a standard network address translation (NAT)

When you define standard network address translations (NATs), you need to create a separate NAT for each node that requires a NAT. You also need to use unique IP addresses for NAT addresses; a NAT IP address cannot match an IP address used by any virtual or physical servers in your network. You can configure a NAT with the Configuration utility or from the command line.

To configure a NAT using the Configuration utility

1. In the navigation pane, click **NATs**.
   The Network Address Translations screen opens.

2. On the toolbar, click **Add NAT**.
   The Add Nat screen opens.
3. Use the fields provided on the Add Nat screen to configure a NAT.
   For additional information about the options on this screen, click the **Help** button.

To configure a NAT from the command line

The **bigpipe nat** command defines one NAT for one node address.

```
bigppipe nat <node addr> to <NAT addr>
```

Defining a secure network address translation (SNAT)

When you define secure network address translations (SNATs), you can assign a single SNAT address to multiple nodes. Note that a SNAT address does not necessarily have to be unique; for example, it can match the IP address of a virtual server.

SNAT addresses have global properties that apply to all SNATs that you define in the BIG-IP Controller configuration as well as to the SNAT mappings you define. You can configure SNATs in the Configuration utility or from the command line.

Setting SNAT global properties

The SNAT feature supports three global properties that apply to all SNAT addresses:

- **Connection limits**
  The connection limit applies to each node that uses a SNAT, and each individual SNAT can have a maximum of 50,000 simultaneous connections.
- **TCP idle connection timeout**
  This timer defines the number of seconds that TCP connections initiated using a SNAT address are allowed to remain idle before being automatically disconnected.

❖ **UDP idle connection timeout**
This timer defines the number of seconds that UDP connections initiated using a SNAT address are allowed to remain idle before being automatically disconnected. This value should not be set to **0**.

To configure SNAT global properties from the Configuration utility

1. In the navigation pane, click **Secure NATs**.
   The Secure Network Address Translations screen opens.
2. In the Secure Network Address Translation screen, configure a SNAT.
   For additional information about the options on this screen, click the **Help** button.

   Configuration notes

   - To turn connection limits off, type **0** in the **Connection Limit** box to turn connection limits off. If you turn connection limits on, keep in mind that each SNAT can support only 50,000 simultaneous connections.

   - The **UDP Idle Connections** value should not be set to **0**.

To configure SNAT global properties from the command line

Configuring global properties for a SNAT requires that you enter three **bigpipe** commands. The following command sets the maximum number of connections you want to allow for each node using a SNAT.

```
bigpipe snat limit <value>
```

The following commands set the TCP and UDP idle connection timeouts:

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

Configuring SNAT address mappings

Once you have configured the SNAT global properties, you can configure SNAT address mappings. The SNAT address mappings define each SNAT address, and also define the node or group of nodes that uses the SNAT address. Note that a SNAT address does not necessarily have to be unique; for example, it can match the IP address of a virtual server. A SNAT address cannot match an address already in use by a NAT or another SNAT address.

To configure a SNAT mapping using the Configuration utility

1. In the navigation pane, click **Secure NATs**.
   The Secure Network Address Translations screen opens.
2. On the toolbar, click **Add SNAT**.
   The Add SNAT screen opens.
3. To Configure the SNAT, fill in the fields on the screen.
   For additional information about the options on this screen, click the **Help** button.

To configure a SNAT mapping from the command line

The **bigpipe snat** command defines one SNAT for one or more node addresses.

```
bigpipe snat map <node addr>... <node addr> to <SNAT addr>
```

For example, the command below defines a secure network address translation for two nodes:

```
bigpipe snat map 192.168.75.50 192.168.75.51 to 192.168.100.10
```

Setting up IP forwarding

If you do not want to translate addresses with a NAT or SNAT, you can use the IP forwarding configuration option. IP forwarding is an alternate way of allowing nodes to initiate or receive direct connections from the BIG-IP Controller's external network. IP forwarding exposes all of the node IP addresses to the external

network, making them routable on that network. If your network uses the NT Domain or CORBA protocols, IP forwarding is an option for direct access to nodes.

To set up IP forwarding, you need to complete two tasks:

- **Turn IP forwarding on**
  The BIG-IP Controller uses a system control variable to control IP forwarding, and its default setting is **off**.
- **Verify the routing configuration**
  You probably have to change the routing table for the router on the BIG-IP Controller's external network. The router needs to direct packets for nodes to the BIG-IP Controller, which in turn directs the packets to the nodes themselves.

Turning on IP forwarding

IP forwarding is a property of the BIG-IP Controller system, and it is controlled by the system control variable **net.inet.ip.forwarding**.

To set the IP forwarding system control variable using the Configuration utility

1. In the navigation pane, click the BIG-IP Controller icon.
   The BIG-IP System Properties screen opens.
2. On the toolbar, click **Advanced Properties**.
   The BIG-IP System Control Variables screen opens.
3. Check the **Allow IP Forwarding** box.
   For additional information about the options on this screen, click the **Help** button.

To set the IP forwarding system control variable from the command line

Use the standard **sysctl** command to set the variable. The default setting for the variable is **0**, which is **off**. You want to change the setting to **1**, which is **on**:

```
sysctl -w net.inet.ip.forwarding=1
```

To permanently set this value, you can use a text editor, such as **vi** or **pico**, to manually edit the **/etc/rc.sysctl** file. For additional information about editing this file, see the ***BIG-IP Controller Reference Guide***, *BIG-IP Controller System Control Variables.*

Addressing routing issues for IP forwarding

Once you turn on IP forwarding, you probably need to change the routing table on the default router. Packets for the node addresses need to be routed through the BIG-IP Controller. For details about changing the routing table, refer to your router's documentation.

Configuring Extended Content Verification service checking

Extended content verification service checking is another feature you can configure after you have performed the three basic configuration tasks. Extended content verification service check is a special type of service check that actually retrieves content from a server. If the content matches the expected result, the BIG-IP Controller marks the node **up** and uses it for load balancing. If the content does not match, or if the server does not return content, the BIG-IP Controller marks the node **down**, and does not use it for load balancing.

You can set up ECV service check using the Configuration utility, or you can use a text editor, such as **vi** or **pico**, to manually create the **/etc/bigd.conf** file, which stores ECV information.

ECV service check is most frequently used to verify content on web servers, although you can use it for more advanced applications, such as verifying firewalls or mail servers. This section focuses on setting up ECV for web servers. For details about using advanced ECV service check options, see the ***Reference Guide***.

◆ Note

It is important to note that the intervals and timeouts for service checks apply to EAV and ECV service checks. These timeouts are configured by setting the service check timers. For more information about setting these timers, see Configuring the timer settings, on page 4-13.

ECV service check properties

ECV service check is a property of both a node port and a node. If you define ECV service check settings for a node port, all nodes that use the port inherit the ECV service check settings. You can override these settings by defining ECV service check settings for the node itself.

There are three different types of ECV service check settings that you can define:

- **ECV normal**
  An ***ECV normal*** service check requires that the BIG-IP Controller mark a node **up** (available for load balancing) when the retrieved content matches the expected result. For example, if the home page for your web site included the words **Welcome home**, you could set up an ECV service check to look for the string **"Welcome home"**. A match for this string would mean that the web server is up and available.
- **ECV SSL**
  An ***ECV SSL*** service check performs the same function as an ECV normal service check, but it is designed to work with secure servers that use the SSL protocol, rather than standard servers using HTTP. The BIG-IP Controller uses SSL version 3, as do popular web browsers, but it is backward-compatible for web servers that support only version 2.

❖ **ECV reverse**
In contrast to **ECV normal**, an ***ECV reverse*** service check requires that the BIG-IP Controller mark a node **down** (not available for load balancing) when the retrieved content matches the expected result. For example, if the content on your web site home page is dynamic and changes frequently, you may prefer to set up a reverse ECV service check that looks for the string **"Error"**. A match for this string would mean that the web server was down.

WARNING

When the BIG-IP Controller checks content looking for a match, it reads through the content until the service check times out, or until the read reaches 5,000 bytes, whichever comes first. When you choose text, an HTML tag, or an image name to search for, be sure to pick one that appears in the first 5,000 bytes of the web page.

Writing regular expressions for ECV service checks

When you set up an ECV service check for a web server, you need to define a send string and a receive expression. A ***send string*** is the request that the BIG-IP Controller sends to the web server. Send strings typically request that the server return a specific web page, such as the default page for a web site. For example, the most common send string is **"GET /"** which simply retrieves the default HTML page for a web site. The ***receive expression*** is the text string that the BIG-IP Controller looks for in the returned web page.

Receive expressions use regular expression syntax, and they are not case-sensitive. Although regular expressions can be complex, you will find that simple regular expressions are adequate for most ECV service checks.

The corresponding receive expression could be any simple text string included in your home page, such as text, HTML tags, or image names.

Sample send strings

The send string below is probably the most common send string, and it retrieves the default HTML page for a web site. Note that all send strings are enclosed by quotation marks (" ") inside the **/etc/bigd.conf** file.

```
"GET /"
```

To retrieve a specific page from a web site, simply enter a fully qualified path name:

```
"GET /www/support/customer_info_form.html"
```

Sample receive expressions

The most common receive expressions contain a text string that would be included in a particular HTML page on your site. The text string can be regular text, HTML tags, or image names. Note that all receive expressions are enclosed by quotation marks (" ").

For example, the following receive expression attempts to match the text **Welcome**, and it is useful for ECV reverse service checks:

```
"welcome"
```

The sample receive expression below searches for a standard HTML tag. Note that even though you are searching for an HTML tag, you still need to enclose the regular expression with quotation marks (" ").

```
"<HEAD>"
```

You can also use null receive expressions, formatted as the one shown below. When you use a null receive expression, the BIG-IP Controller considers any content retrieved to be a match.

```
""
```

Null receive expressions are suitable only for ECV normal and ECV SSL. Note, however, that if you use them you run the risk of the BIG-IP Controller considering an HTML error page to be a successful service check.

◆ Note

*The regular expression syntax discussed here is not the same as the **wildcard syntax** that is commonly used in command shells. For more information about regular expression, see the man page for **re_format**. To view the man page for **re_format**, type **man re_format** at the command line.*

Setting up ECV service checks using the Configuration utility

Using the Configuration utility, you can set ECV service check options in the Global Node Port Properties screen, and also in individual Node Properties screens. Regardless of which screen you use to configure the options, the steps are the same.

To set up ECV service check using the Configuration utility

1. In the navigation pane, click **Nodes**.
   The Nodes screen opens.
2. Select a node from the list.
   The Node Properties screen opens.
3. If you want to configure ECV service check options, stay in this screen.

   If you want to configure ECV service check options for the port that the node uses, click the port number listed next to the IP address of the node.
4. Click the **ECV** button.
5. In the **Type** box, choose the type of ECV service check you want to set up: normal, reverse, or SSL.

6. In the **Send String** box, type the send string that requests the web page. Note that the Configuration utility automatically places quotation marks around the string itself. For example, the following string retrieves the default HTML page for the site:

   ```
   GET /
   ```

7. In the **Receive Rule** box, type the receive expression that the BIG-IP Controller should look for in the returned web page. For example, the following receive expression looks for a text string in a web page:

   ```
   Welcome home!
   ```

8. Click the **Apply** button.
   For additional information about the options on this screen, click the **Help** button.

Manually configuring and testing the /etc/bigd.conf file

You can set up ECV service check on the command line by creating an **/etc/bigd.conf** file in a text editor such as **vi** or **pico**. Each line in the **/etc/bigd.conf** file defines a send string and a receive expression for one node, or for one port. Remember that when you define a ECV service check for a port, all nodes that use the port inherit the service check settings.

Changes to the **/etc/bigd.conf** do not take effect until the system is rebooted, or **bigd** is restarted. To restart **bigd**, simply run the command **bigd**.

The BIG-IP Controller provides a command line tool that allows you to verify the syntax of the **/etc/bigd.conf** file before the system begins using it. Once you set up the file, we recommend that you test it before you reboot the system or restart the **bigd** daemon and begin using the file.

Setting up the /etc/bigd.conf file

The **/etc/bigd.conf** file uses three different types of syntax for lines in the file that correspond to the three different types of service check that you can configure: ECV normal, ECV SSL, and ECV reverse. The following sections describe the syntax for each type, and provide some useful examples.

To set up an ECV normal service check

The line for a normal ECV service check begins with the keyword **active**. The <**node IP**> parameter is optional, and you need to include it only if you are defining an ECV service check for a specific node.

```
active [<node IP>:]<port> "<send_string>" "<recv_expr>"
```

For example, the following line sets up a normal ECV service check for a node, where the BIG-IP Controller looks for the text **Welcome** in the default page for the site.

```
active 192.168.100.10:80 "GET /" "welcome"
```

To set up ECV SSL service check

The line for an SSL ECV service check begins with the keyword **ssl**. The <**node IP**> parameter is optional, and you need to include it only if you are defining ECV service check for a specific node.

```
ssl [<node IP>:]<port> "<send_string>" "<recv_expr>"
```

For example, the following line sets up an SSL ECV service check for a node port. Note that the receive expression is null. When you use a null receive expression, the BIG-IP Controller considers any retrieved content to be a match.

```
ssl 443 "GET /www/orders/order_form.html" ""
```

To set up ECV reverse service check

The line for a reverse ECV service check begins with the keyword **reverse**. The <**node IP**> parameter is optional, and you need to include it only if you are defining ECV service check for a specific node.

```
reverse [<node IP>:]<port> "<send_string>" "<recv_expr>"
```

For example, the following line sets up a reverse ECV service check for a node port. Note that the receive expression is null. When you use a null receive expression, the BIG-IP Controller considers any retrieved content to be a match.

```
reverse 80 "GET /" ""
```

Testing /etc/bigd.conf syntax

To test /etc/bigd.conf syntax

You can test your ECV syntax in the **bigd.conf** file using the following **bigd** command:

```
/sbin/bigd -d
```

This command parses the file, checks ECV syntax, reports any errors, and then exits.

◆ Note

*The Big-IP Controller reads the **/etc/bigd.conf** file once at startup. If you change the file from the command line, you must reboot or restart **bigd** for the changes to take effect. If you make changes in the Configuration utility, clicking the **Apply** button makes changes and restarts **bigd**. For more information about **bigd**, see the **BIG-IP Controller Reference Guide**, System Utilities.*

Configuring and synchronizing redundant systems

Another optional feature you can configure after you have performed the four basic configuration tasks is configuration synchronization.

Redundant BIG-IP Controller systems have special settings that you need to configure, such as interface fail-safe settings. One convenient aspect of configuring a redundant system is that once you have configured one of the controllers, you can simply copy the configuration to the other controller in the system using the configuration synchronization feature in the **bigpipe** command line tool or in the Configuration utility.

There are two basic aspects to working with redundant systems:

- Synchronizing configurations between two controllers
- Configuring fail-safe settings for the interfaces

Preparing to use the synchronization command

Before you can use the **bigpipe configsync** command or the Configuration utility to synchronize domestic HA redundant BIG-IP Controllers, you must first run the **config_failover** command. This command performs the following tasks:

- Checks for a fail-over IP address for the other controller in BIG/db.
- Verifies that the **AllowHosts** entry in the **/etc/sshd_config** file includes the IP address of the other controller in the redundant configuration.
- Runs the **ssh-keygen** command which creates the security keys for the controller.
- Shares the security keys with the other controller in the redundant system.

To run the **config_failover** command, type the following command from the command line:

```
config_failover
```

The **config_failover** utility prompts you for the root password of the other controller in the redundant system before it generates the security keys for the BIG-IP Controller.

Synchronizing configurations between controllers

Once you complete the initial configuration on the first controller in the system, you can synchronize the configurations between the active unit and the standby unit. When you synchronize a configuration, the following configuration files are copied to the other BIG-IP Controller:

- **The common keys in BIG/db**

- **/etc/bigip.conf**
  The **/etc/bigip.conf** file stores virtual server and node definitions and settings, including node ping settings, the load balancing mode, and NAT and SNAT settings.
- **/etc/bigd.conf**
  The **/etc/bigd.conf** file stores service check settings.
- **/etc/hosts.allow**
  The **/etc/hosts.allow** file stores the IP addresses that are allowed to make administrative shell connections to the BIG-IP Controller.
- **/etc/hosts.deny**
  The **/etc/hosts.deny** file stores the IP addresses that are not allowed to make administrative shell connections to the BIG-IP Controller.
- **User account files**
- **/etc/ipfw.conf** and **/etc/ipfw.filt**
  The **/etc/ipfw.conf** and **/etc/ipfw.filt** files store IP filter settings.
- **rc.sysctl**
  The **rc. sysctl** file contains system control variable settings.
- **/etc/rateclass.conf**
  The **/etc/rateclass.conf** file stores rate class definitions.
- **/etc/ipfwrate.conf** and **/etc/ipfwrate.filt**
  The **/etc/ipfwrate.conf** and **/etc/ipfwrate.filt** files store IP filter settings for filters that also use rate classes.
- **/etc/snmpd.conf**
  The **/etc/snmpd.conf** file stores SNMP configuration settings.

If you use command line utilities to set configuration options, be sure to save the current configuration to the file before you use the configuration synchronization feature. Use the following **bigpipe** command to save the current configuration:

```
bigpipe -s
```

WARNING

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

To synchronize the configuration using the Configuration utility

1. In the navigation pane, click the BIG-IP logo.
   The BIG-IP System Properties screen opens.
2. On the toolbar, click the **Sync Configuration** button.
   The Sync Configuration screen opens.
3. Click the **Synchronize** button.
   For additional information about the options on this screen, click the **Help** button.

To synchronize the configuration from the command line

You use the **bigpipe configsync** command to synchronize configurations. When you include the **all** option in the command, all the configuration files are synchronized between machines.

```
bigpipe configsync all
```

If you want to synchronize only the **/etc/bigip.conf** file, you can use the same command without any options:

```
bigpipe configsync
```

Configuring fail-safe settings

For maximum reliability, the BIG-IP Controller supports failure detection on both internal and external interface cards. When you arm the fail-safe option on an interface card, the BIG-IP Controller monitors network traffic going through the interface. If the BIG-IP Controller detects a loss of traffic on an interface when half of the fail-safe timeout has elapsed, it attempts to generate traffic. An interface attempts to generate network traffic by issuing ARP requests to nodes accessible through the interface. Also, an ARP request is generated for the default route if the default router is accessible from the interface. Any traffic through the interface, including a response to the ARP requests, averts a fail-over.

If the BIG-IP Controller does not receive traffic on the interface before the timer expires, it initiates a fail-over, switches control to the standby unit, and reboots.

WARNING

You should arm the fail-safe option on an interface only after the BIG-IP Controller is in a stable production environment. Otherwise, routine network changes may cause fail-over unnecessarily.

Arming fail-safe on an interface

Each interface card installed on the BIG-IP Controller has a unique name, which you need to know when you set the fail-safe option on a particular interface card. You can view interface card names in the Configuration utility, or you can use the **bigpipe interface** command to display interface names on the command line.

To arm fail-safe on an interface using the Configuration utility

1. In the navigation pane, click **NICs** (network interface cards).
   The Network Interface Cards list opens and displays each installed NIC.
2. Select an interface name.
   The Network Interface Card Properties screen opens.
3. Check **Arm Failsafe** to turn on the fail-safe option for the selected interface.
4. In the **Timeout** box, type the maximum time allowed for a loss of network traffic before a fail-over occurs.
5. Click the **Apply** button.
   For additional information about the options on this screen, click the **Help** button.

To arm fail-safe on an interface from the command line

One of the required parameters for the **bigpipe interface** command is the name of the interface itself. If you need to look up the names of the installed interface cards, use the **bigpipe interface** command with the **show** keyword:

```
bigpipe interface show
```

To arm fail-safe on a particular interface, use the **bigpipe interface** command with the **failsafe arm** keyword and interface name parameter:

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

For example, you have an external interface named **exp0** and an internal interface named **exp1**. To arm the fail-safe option on both cards with a timeout of 30 seconds, you need to issue the following commands:

```
bigpipe interface timeout 30
bigpipe interface exp0 failsafe arm
bigpipe interface exp1 failsafe arm
```

Glossary

attributes

An attribute is a variable that the cache statement uses to direct requests. Attributes can be either required or optional.

BIG-IP 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-IP web server

The web server that runs on a BIG-IP Controller and hosts the Configuration utility.

bigpipe

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 (Berkeley Internet Name Domain)

The most common implementation of DNS, which provides a system for matching domain names to IP addresses.

cacheable content determination

Determines the type of content you cache on the basis of any combination of elements in the HTTP header.

cacheable content expression

An expression that determines, based on evaluating variables in the HTTP header of the request, whether or not a BIG-IP Cache Controller directs a given request to a cache server or to an origin server. Any content that does not meet the criteria in the cacheable content expression is deemed non-cacheable.

cache_pool

Specifies a pool of cache servers to which requests are directed in a manner that optimizes cache performance. The BIG-IP Cache Controller directs all requests bound for your origin server to this pool, unless you have configured the hot content load balancing feature and the request is for hot (frequently requested) content. See also *hot* and *origin server.*

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.

content affinity

Ensures that a given subset of content remains associated with a given cache server to the maximum extent possible, even when cache servers become unavailable, or are added or removed. This feature also maximizes efficient use of cache memory.

content demand status

A measure of the frequency with which content in a given hot content subset is requested over a given hit_period. Content demand status is either hot, in which case the number of requests for content in the hot

content subset during the most recent hit_period has exceeded the hot_threshold, or cool, in which case the number of requests during the most recent hit period is less than the cool_threshold. See also *cool, cool_threshold, hit_period, hot, hot content subset,* and *hot_threshold.*

content_hash_size

Specifies the number of units, or hot content subsets, into which the content is divided when determining whether content is hot or cool. The requests for all content in a given subset are summed, and a state (hot or cool) is assigned to each subset. The content_hash_size should be within the same order of magnitude as the actual number of requests possible. For example, if the entire site is composed of 500,000 pieces of content, a content_hash_size of 100,000 would be typical.

If you specify a value for hot_pool, but do not specify a value for this variable, the cache statement uses a default hash size of 10 subsets. See also *content_hash_size, cool, hot, hot content subset.*

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.

cool

A term used to describe content demand status when using hot content load balancing. See also *content demand status, hot,* and *hot content load balancing.*

cool_threshold

Specifies the maximum number of requests for given content that will cause that content to change from hot to cool at the end of the hit period.

If you specify a variable for hot_pool, but do not specify a value for this variable, the cache statement uses a default cool threshold of 10 requests. See also *cool, hit_period,* and *hot.*

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.

destination processing

Destination processing means that the interface rewrites the destination address of an incoming packet.

destination translation

Included in destination processing, destination translation means that the interface rewrites the destination address of an incoming packet. See also *destination processing*.

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.

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.

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.

forward proxy caching

A configuration, in which a BIG-IP Cache Controller redundant system uses content-aware traffic direction to enhance the efficiency of an array of cache servers storing Internet content for internal users.

hit_period

Specifies the period in seconds over which to count requests for particular content before determining whether to change the state (hot or cool) of the content.

If you specify a value for hot_pool, but do not specify a value for this variable, the cache statement uses a default hit period of 10 seconds. See also *cool*, *hot*, and *hot_pool*.

host

A network server which manages one or more virtual servers that the 3DNS Controller uses for load balancing.

hot

A term used to define frequently requested content based on the number of requests in a given time period for a given hot content subset. See also *hot content subset*.

hot_pool

A designated group of cache servers to which requests are load balanced when the requested content is hot. If a request is for hot content, the BIG-IP Cache Controller redundant system directs the request to this pool.

hot content load balancing

Identifies hot, or frequently requested, content on the basis of number of requests in a given time period for a given hot content subset. A hot content subset is different from, and typically smaller than, the content subsets used for content striping. Requests for hot content are redirected to a cache server in the hot pool, a designated group of cache servers. This feature maximizes the use of cache server processing power without significantly affecting the memory efficiency gained by cacheable content determination. See also *hot*, *hot content subset*, and *hot_pool*.

hot content subset

A hot content subset is different from, and typically smaller than, the content subsets used for cacheable content determination. This is

created once content has been determined to be hot and is taken or created from the content subset. See also *cacheable content determination.*

hot_threshold

Specifies the minimum number of requests for content in a given hot content subset that will cause that content to change from cool to hot at the end of the period.

If you specify a value for hot_pool, but do not specify a value for this variable, the cache statement uses a default hot threshold of 100 requests. See also *cool*, *hot*, *hot content subset*, and *hot_pool*.

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.

intelligent cache population

Allows caches to retrieve content from other caches in addition to the origin web server. This feature is useful only when working with non-transparent cache servers, which can receive requests that are destined for the cache servers themselves, as opposed to transparent cache servers, which can intercept requests destined for a web server but cannot themselves receive requests. Intelligent cache population minimizes the load on the origin web server and speeds cache population. See also *non-transparent cache server* and *transparent cache server.*

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.

miss request

A miss request results from a request for content a cache does not have.

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.

non-cacheable content

Content that is not identified in the cacheable content condition part of a cache rule statement. See also *cacheable content condition*.

non-transparent cache servers

Cache servers that can receive requests that are destined for the cache servers themselves.

origin server

The web server on which all original copies of your content reside

origin_pool

Specifies a pool of servers that contain original copies of all content. Requests are load balanced to this pool when any of the following are

true: the requested content is not cacheable, no cache server is available or the BIG-IP Cache Controller redundant system is redirecting a request from a cache server that did not have the requested content.

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.

pool

A pool is a group of devices that you want the Big_IP Cache Controller redundant system to load balance.

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.

remote server acceleration

A configuration in which a BIG-IP Cache Controller redundant system uses content-aware traffic direction to enhance the efficiency of an array of cache servers that cache content for a remote web server.

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.

source processing

Source processing means that the interface rewrites the source of an incoming packet.

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.

stripes
Cacheable content subsets distributed among your cache servers.

transparent cache servers
Cache servers that can intercept requests destined for a web server but cannot themselves receive requests.

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.

F5 Networks

BIG·IP®

Local Traffic Controller

Index

A

B

C

D

E

F

H

I

L

M

N

O

P

R

S

T

U

V

W